CLOUD ACCOUNTABILITY PROJECT

D:D-2.4b Cloud Accountability Toolkit Architecture

Deliverable Number	D42.4b
Work Package	WP 42
Version	Final
Deliverable Lead Organisation	ATC
Dissemination Level	PU
Contractual Date of Delivery (release)	31/03/2016
Date of Delivery	31/03/2016

Editors

Vasilis Tountopoulos (ATC), Giorgos Giotis (ATC)

Contributors

Frederic Gittler (HPE), Theo Koulouris (HPE), Siani Pearson (HPE), Rehab Alnemr (HPE), Jean-Claude Royer (EMN), Michela D' Errico (HPE), Anderson Santana De Oliveira (SAP), Thomas Rubsamen (HFU), Philipp Ruf (HFU), Christoph Reich (HFU), Mehdi Haddad (EMN), Ali Kassem (EMN), Mario Sudholt (EMN), Tobias Pulls (KaU), Carmen Gago (UMA), Christian Froystad (SINTEF), Martin Jaatun (SINTEF), Lorenzo Dalla Corte (TiU), Massimo Felici (HPE), Walid Benghabrit (EMN)

Reviewers

All tool owners

A4Cloud
www.a4cloud.eu
Accountability For Cloud and Other Future Internet Services
FP7-ICT-2011-8-317550-A4CLOUD

SEVENTH FRAMEWORK PROGRAMME

Executive Summary

This document is an appendix to the Cloud Accountability Reference Architecture document (deliverable D42.4), which describes the A4Cloud toolkit, showing the way that the A4Cloud reference architecture can be instantiated through a reference implementation of the accountability support services and the relevant artefacts. The A4Cloud toolkit is presented, based on the phases of the accountability mechanisms, namely tools that facilitate preventive, detective and/or corrective mechanisms. On top of that, the tools are grouped into categories, according to a functional classification of the intended tool usage. This categorisation is presented in Section 1. The subsequent sections are dedicated in the specification of the tools comprising the A4Cloud toolkit, considering the functional tool categorisation, while Section 7 summarises the way that the tools interact between each other to implement the accountability support services and deliver the respective artefacts.

The document presents the different tools with the aim to assist the reader in understanding:

- The scope of the tool in the Cloud Accountability Reference Architecture;
- The target cloud stakeholders that the tool functionalities aim to address;
- The position of each tool with respect to the other tools of the A4Cloud toolkit.

Thus, for each tool, we present an overview of the tool, describing its main functions and the envisaged target stakeholders, the high level architecture of the tool, identifying its major functional components and the description of both the user and machine/application programming interfaces (UI and API) provided, indicating the input and output for each interface and the expected consumers, as well as the main required machine interfaces from other A4Cloud or external tools.

Compared to previous releases of the tools documentation, this annex introduces two additional tools, namely the Accountability Monitor and the Security and Privacy Assurance Case Environment. These tools have been introduced to fill in the gap identified, especially in the process for provision of an account, between the definition of the cloud accountability reference architecture and the reference implementation of accountability support services and relevant mechanisms.

As said above, the document is concluded with a summary of the tools interactions in Section 7. This section describes how the tools interact with each other in order to implement the accountability support services and which accountability artefacts are involved in the tools interactions to accomplish a specific service. In detail, the interactions of the tools occur as a result of the accomplishment of the functions of accountability, which are identified in the relevant lifecycle for accountability for an accountable organisation, and the respective accountability support services. For these interactions, we demonstrate the involvement of the accountability artefacts, as an important input or output for each A4Cloud tool.

As explained above, this document is an annex to Deliverable D42.4, which explains the implementation of the accountability support services. The relevant tools of the A4Cloud toolkit comprise a reference implementation of these services and by no means aim to prevail as the sole and only implementation of the cloud accountability reference architecture. In this sense, the document is exploited for the instantiation of the cloud accountability reference architecture in WP47 and the demonstration of how accountability can be implemented in a real life business case, which involves the collaboration of multiple cloud service providers for the collection and processing of personal data from the business end users in an accountable way.

Table of Contents

1 Introduction

The tools constituting the A4Cloud toolkit have been designed to support accountability for data governance in the cloud by specifying certain functions identified during the development of the A4Cloud accountability framework. The tool design process was motivated by the goal to provide stakeholders with tools supporting those elements of accountability for which little or no support was found to exist, while complementing existing privacy and security mechanisms. Each A4Cloud tool addresses different elements of accountability, accountability support services and respective artefacts, and may operate over different timescales.

As explained in Deliverable D:D-2.4: Cloud Accountability Reference Architecture, the accountability mechanisms address six key groups of practices, as shown in Figure 1, which are mapped onto an Accountability Lifecycle that should be operated by an accountable organisation.

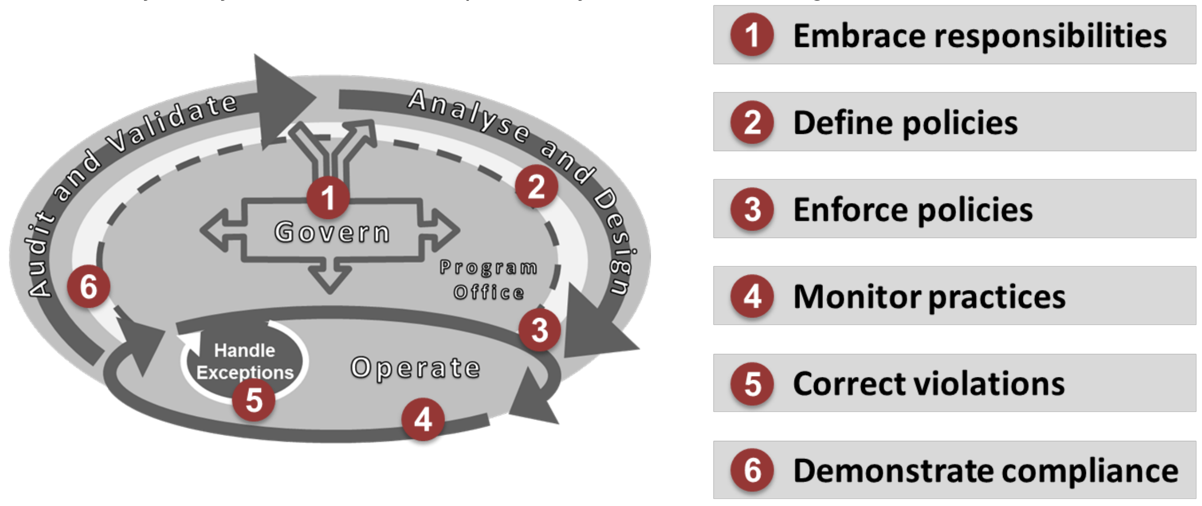


Figure 1: Accountability Lifecycle and Practices.

The A4Cloud toolkit supports these accountable organisations in running these practices, by implementing the accountability functions to be executed by the cloud and data protection roles. At a high level, accountability functions can be classified as being preventive, detective and corrective. Preventive functions focus on mitigating the occurrence of an unauthorized action. In the RA, preventive functions include assessing a risk, identifying and expressing appropriate policies to mitigate it, and enforcing the latter via mechanisms and procedures put in place. Detective functions are used to identify the occurrence of an incident or risk that goes against the policies and procedures in place. In the RA, these centre on monitoring and identifying policy violations via detection and traceability measures such as audit, tracking, reporting, and monitoring. Finally, corrective functions are those that are used to fix an undesired result that has already occurred. In the RA, these focus on managing incidents, providing notifications and facilitating redress.

Using this categorization, we can further classify the A4Cloud tools into five functional areas:

- Contract and Risk Management: the A4Cloud tools in this area aim to address the need for supporting the management of risks and the selection of suitable cloud service contracts in the context of accountability for data governance in the cloud. All tools in this category implement preventive accountability functions.
- Policy Definition and Enforcement: the A4Cloud tools in this area aim to address the functionalities needed for defining and enforcing accountability policies, as well as their maintenance within the data lifecycle of a cloud service provision chain. The tools in this category implement preventive accountability functions.
- Evidence and Validation: the A4Cloud tools in this area deal with the collection and provision of evidence and the validation of the proper execution of the accountability tools in a specific setting. The tools in this category implement both preventive and detective accountability functions.

- Data Subject Controls: the A4Cloud tools in this area target the needs of data subjects by providing controls for the proper management and protection of their personal data in a cloud service ecosystem. The tools in this category implement detective accountability functions.
- Incident Management and Remediation: the A4Cloud tools in this area provide corrective accountability functions facilitating remediation and redress.

This functional categorisation of the tools enables the allocation of the accountability functions and the respective support services into tools that we have implemented in the course of the project and they are introduced in the following Table 1.

Functional area	Accountability Support Service	Name of the tool	Main tool usage scenario
Contract and Risk Management	Policy Definition and Validation	Data Protection Impact Assessment Tool	To assess the impact of the cloud provider selection on the data protection aspects, and get the requirements to follow specific privacy, security and functional steps
		Cloud Offerings Advisory Tool	To get a guided selection of a cloud provider, according to functional, security and privacy requirements
Policy Definition and Enforcement	Policy Definition and Validation	Data Protection Policies Tool	To create accountability policies
		Accountability Lab	To perform policy matching between abstract policy statements and preferences
	Policy Management and Enforcement	Accountable Primelife Policy Engine	To enforce accountability policies for the management of personal data
Evidence and Validation	Monitoring and Environment State Collection	Accountable Primelife Policy Engine	To generate logs on the data handling processes with respect to data access and retention properties for monitoring and auditing purposes
		Audit Agent System	To analyse logs on the data handling processes with respect to data retention and integrity properties for monitoring and auditing purposes
		Data Transfer Monitoring Tool	To generate logs on the data handling processes with respect to data transfer

Functional area	Accountability Support Service	Name of the tool	Main tool usage scenario
			properties for monitoring and auditing purposes
		Accountability Monitor	To provide means to monitor accountability policies in the context of a real system
	Collection and Management of Evidence	Audit Agent System	To collect and create evidence
		Transparency Log	To securely store logs and evidence records
	Validation	Audit Agent System	To perform internal and external auditing
		Assertion Tool	To validate the proper implementation of the accountability support services
		Security and Privacy Assurance Case Environment	To provide assurance of how cloud providers comply with relevant policies and support them in operational environments
Data Subject Controls	Validation	Data Track	To control the disclosure of personal data in the cloud
		Transparency Log	To enable secure communication with Data Controller
	Notification	Plug-in for Assessment of Policy Violation	To allow the collection of violations from the cloud, against agreed data handling processes, and managing their severity level
Incident Management and Remediation	Incident Management	Incident Management Tool	To assess the type of the perceived and/or reported incidents
		Audit Agent System	To raise incidents on an abnormal behaviour of the environment
		Data Transfer Monitoring Tool	To raise incidents on an abnormal behaviour of the environment
	Notification	Incident Management Tool	To generate notification alerts

Functional area	Accountability Support Service	Name of the tool	Main tool usage scenario
		Accountable Primelife Policy Engine	To implement notifications, based on accountability policies
	Remediation	Remediation and Redress Tool	To suggest remediation, such as to complete and submit complaints form to a DPA, enforce the selected remediation/ redress actions

Table 1: The tools of the A4Cloud toolkit.

In more details, the A4Cloud toolkit is composed of the following fourteen (14) tools and one plug-in:

- The Data Protection Impact Assessment Tool (DPIAT): This tool is used by Small-Medium Enterprises (SMEs) to identify the risks in a given configuration and environment of carrying out a certain business transaction, which involves the processing of personal or confidential data.
- The Cloud Offerings Advisory Tool (COAT): This tool is designed to assist potential cloud customers (SME organizations and individuals) in assessing and selecting cloud offerings, with respect to certain security and privacy requirements.
- The Accountability Lab (AccLab): This tool is used to check the consistency between human readable accountability obligations expressed in the Abstract Accountability Language (AAL) and the compliance of machine-readable accountability policy language called Accountable Primelife Policy Language (A-PPL) with such AAL statements.
- The Data Protection Policies Tool (DPPT): This tool is used by Cloud Providers to create machine readable representation of accountability policy statements in A-PPL enforceable language. The tool is used to configure the component in charge of enforcing the accountability related policies, by translating the policies into an A-PPL file and sending it to the responsible policy enforcement component.
- The Accountable Primelife Policy Engine (A-PPL Engine or A-PPLE): This tool enforces data handling policies expressed in A-PPL and generates logs with respect to the actions enforced in compliance to these policies.
- The Audit Agent System (AAS): This tool enables the automated audit of multi-tenant and multi-layer cloud applications and cloud infrastructures for compliance with custom-defined policies, using software agents.
- The Data Transfer Monitoring Tool (DTMT): This tool automates the collection of evidence describing how data transfers within a cloud infrastructure comply with data handling policies.
- Data Track (DT): This tool is used by data subjects to get a user-friendly visualization of all personal data they have disclosed to cloud services, with the additional capability to rectify data if necessary.
- The Assertion Tool (AT): This tool ensures the validation of the A4Cloud tools through a test case-based validation methodology, during the development and deployment phases.
- The Accountability Monitor (AccMon): This tool provides the means for cloud service providers to monitor the implementation of accountability policies.
- The Security and Privacy Assurance Case Environment (SPACE): This tool addresses the problem of providing assurance on how the cloud service providers can demonstrate their compliance with relevant policies and support them in operational environments towards increasing trustworthiness and enhancing transparency in a cloud environment.

- The Transparency Log (TL): This cryptographic tool provides a secure and privacy-preserving unidirectional asynchronous communication channel, typically between a cloud subject and a cloud provider. Messages can be stored on untrusted system in the form of buffers, which are called piles, and can still be securely retrieved asynchronously by recipients registered to these piles.
- The Remediation and Redress Tool (RRT): This tool assists cloud customers (individuals or SMEs) in responding to real or perceived data handling incidents and their redress.
- The Incident Management Tool (IMT): This tool is the entry point for managing anomalies and violations that occur in cloud services and should be notified to the cloud subjects, such as privacy violations or security breaches. The tool enables cloud subjects receive incident notifications and takes the initial steps to respond to these incidents, by gathering comprehensive and structured information related to these incidents.
- The Plug-in for Assessment of Policy Violation (PAPV): This is a plug-in component to DT that provides an assessment on the criticality of previously detected policy violations. By using it, the cloud actors can check which policy violations are the most relevant ones to their data processed in the cloud service supply chain.

Figure 2 illustrates the A4Cloud tools according to this classification.

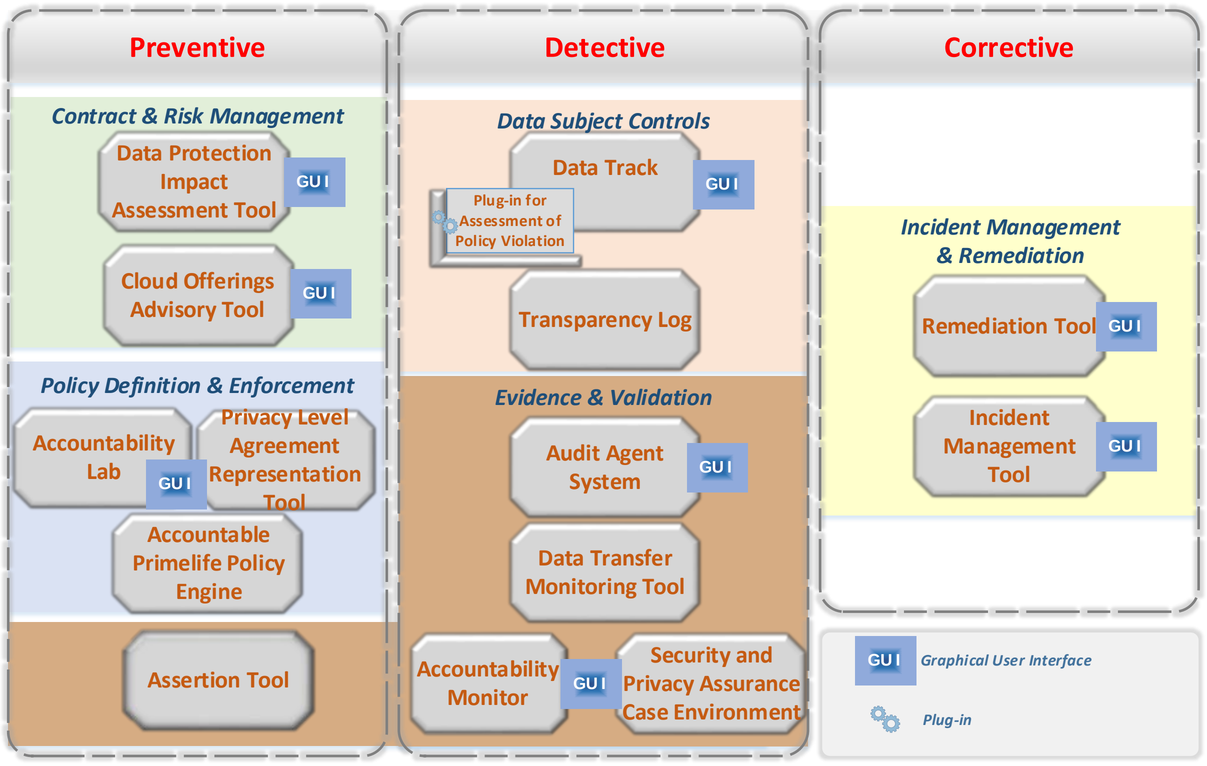

Preventive
Detective
Corrective
Contract & Risk Management
Data Protection Impact Assessment Tool
GUI
Cloud Offerings Advisory Tool
GUI
Policy Definition & Enforcement
Accountability Lab
GUI
Privacy Level Agreement Representation Tool
Accountable Primelife Policy Engine
Assertion Tool
Data Subject Controls
Data Track
GUI
Plug-in for Assessment of Policy Violation
Transparency Log
Evidence & Validation
Audit Agent System
GUI
Data Transfer Monitoring Tool
Accountability Monitor
GUI
Security and Privacy Assurance Case Environment
Incident Management & Remediation
Remediation Tool
GUI
Incident Management Tool
GUI
GUI Graphical User Interface
Plug-in

Figure 2: High level view of the A4Cloud Toolkit.

The rest of this chapter presents a detailed description of each A4Cloud tool, examining these functional areas in succession. The description of the tools is structured, so that, for each tool, the following information is provided:

- An overview of the tool, describing its main functions and the envisaged target stakeholders;
- The high level architecture of the tool, identifying its major functional components;
- The description of both the user and machine/application programming interfaces (UI and API) provided, indicating the input and output for each interface and the expected consumers, as well as the main required machine interfaces from other A4Cloud or external tools.

2 Contract and Risk Management

The contract and risk management area of the A4Cloud toolkit architecture contains tools which address the need for providing support in managing risk and cloud service contract selection in the context of accountability for data stewardship in the cloud. As a result, tools in this area serve a *preventive* role. Prevention is facilitated via two separate but complementary mechanisms, namely:

i. Evaluation of cloud offerings and contract terms with the goal of enabling a more educated decision making on which service and service provider to select.
ii. Assessment of the risks associated with various facets of the cloud service consumption process, involving personal and/or confidential data and elicitation of actionable information and guidance on how to mitigate them;

For the instantiation of this part of RA, these two mechanisms are being developed as distinct A4Cloud software tools: the Data Protection Impact Assessment Tool (DPIAT) and the Cloud Offerings Advisory Tool (COAT).

2.1 Data Protection Impact Assessment Tool

2.1.1 Tool Overview and Target Stakeholders

DPIAT is a Web-based tool for individuals working in SMEs. The tool considers a number of information sources, from which cloud specific risks and existing countermeasures can be collected and evaluated, in the process of supporting impact assessments for projects assuming to process personal data in the cloud. The scope of this tool is to provide guidance in the process of first determining the need for a fully-fledged DPIA, and then, of conducting the assessment in a friendly, yet didactic, manner. The assessment is performed through the provision of information about the project under evaluation and its organisational practices, combined with the selection of a cloud service provider (CSP). During the process, DPIAT proposes up-to-date DPIA questionnaires with respect to existing standards and recommendations, building on the expertise of experts from different disciplines, ranging from law sciences, information security and risk management, to user experience design.

DPIAT employs the Cloud Adoption Risk Assessment Model (CARAM) [1] to evaluate the risks resulting from the adoption of specific cloud services. The model is designed to assist (potential) cloud customers in assessing all kinds of risks—not only privacy-related—that they face by selecting a specific CSP. In principle, it is a qualitative deductive risk assessment model based on ENISA's cloud risk assessment model and the CSA's Cloud Assessment Initiative Questionnaire (CAIQ) [2]. It complements ENISA's approach [3][4] to take into account cloud customers' assets (modelled based on the list of assets from the ENISA report) and the implementation status of security controls in CSA STAR public registry [5] to perform a relative risk assessment of (potential) cloud solutions. This can help cloud consumers to determine which CSPs have acceptable risk profiles for security, privacy, and quality of service. CARAM extends the DPIA process to account for the security controls implemented by the CSP. The results of CARAM risk assessment constitute a part of the DPIAT results.

2.1.2 High Level Architecture

The high level architecture of DPIAT is shown in Figure 3. DPIAT consists of the following components, implementing respective process level mechanisms:

- The User Questionnaire, which offers interaction with the users to collect information about the project (s)he will be involved in the intended cloud service (or the type of data that are going to be used in the cloud service).
- The Rule-based Engine, which sets up and processes the rules to the questionnaire path taken when the user responds with specific answers.
- The Risk Assessment Plugin, which assesses the risks associated with the project based on the information collected from the User Questionnaire.
- The Reporting Component, which is responsible for presenting the user with a complete assessment report in a comprehensible and user-friendly format.

- The Information and Explanation Component, which provides support on the meaning of each question and the implications of the responses.
- The Logging Component, which logs the responses to the questionnaire and the final report given to the user.

These components are fed with user related information, including data location, the roles involved in the project, contextual information on the environment setting, the respective trust and risk modelling (through the CARAM model), external certification systems (with emphasis for our case the CSA STAR Certification program, which is to manually be adopted, since there is no automatic integration with this program), the organisational policies for the protection of the classified data and the knowledge base (KB) of the threats and their associated mitigation control actions.

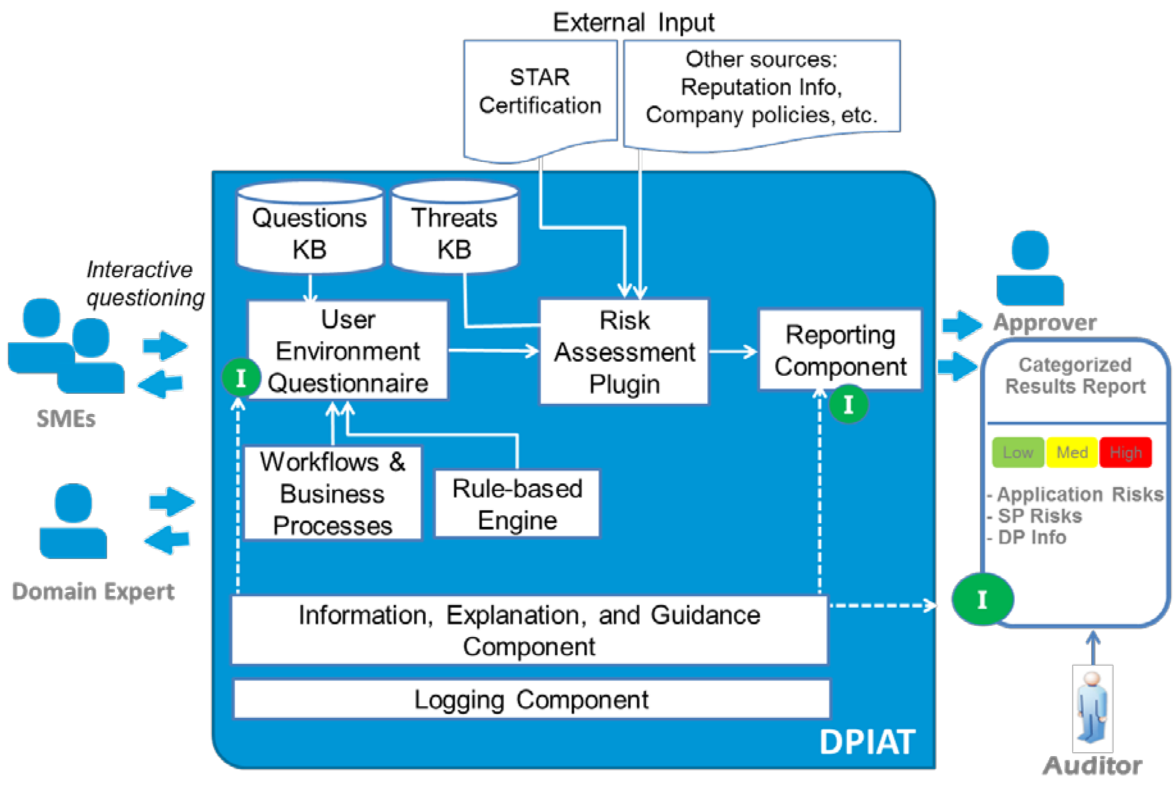


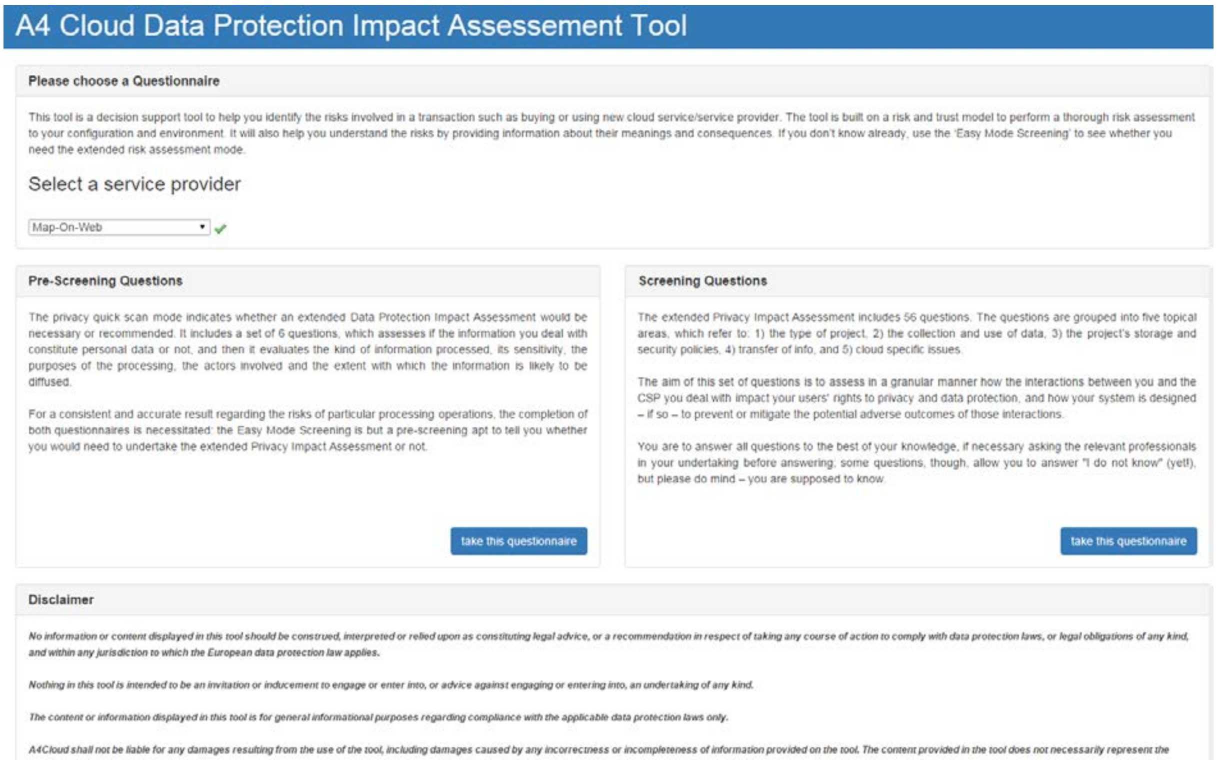
Figure 3: The high level architecture of the Data Protection Impact Assessment Tool.

Through this architecture, the DPIAT is capable of assessing a set of use cases, including the support to decision making on competing cloud service offerings and the action on moving data into the cloud.

2.1.3 User Interface

The Web interface of DPIAT enables easy and user-friendly access and experience.

The home page shown in Figure 4 asks the users whether they would like to start with an easy-mode questionnaire to assess if they need to answer the full expert-mode questionnaire. The easy-mode questionnaire consists of six (6) preliminary screening questions to make a quick assessment of the project being carried by the user. Certain answers from the user will lead the tool to direct him/her to the expert-mode questionnaire to carry on with a full data protection risk assessment, e.g. if the project contains sensitive data to be stored in the cloud. The expert-mode questionnaire contains a set of fifty (50) questions.



A4 Cloud Data Protection Impact Assessement Tool

Please choose a Questionnaire

This tool is a decision support tool to help you identify the risks involved in a transaction such as buying or using new cloud service/service provider. The tool is built on a risk and trust model to perform a thorough risk assessment to your configuration and environment. It will also help you understand the risks by providing information about their meanings and consequences. If you don't know already, use the 'Easy Mode Screening' to see whether you need the extended risk assessment mode.

Select a service provider

Map-On-Web ✓

Pre-Screening Questions

The privacy quick scan mode indicates whether an extended Data Protection Impact Assessment would be necessary or recommended. It includes a set of 6 questions, which assesses if the information you deal with constitute personal data or not, and then it evaluates the kind of information processed, its sensitivity, the purposes of the processing, the actors involved and the extent with which the information is likely to be diffused.

For a consistent and accurate result regarding the risks of particular processing operations, the completion of both questionnaires is necessitated: the Easy Mode Screening is but a pre-screening apt to tell you whether you would need to undertake the extended Privacy Impact Assessment or not.

Screening Questions

The extended Privacy Impact Assessment includes 56 questions. The questions are grouped into five topical areas, which refer to: 1) the type of project, 2) the collection and use of data, 3) the project's storage and security policies, 4) transfer of info, and 5) cloud specific issues.

The aim of this set of questions is to assess in a granular manner how the interactions between you and the CSP you deal with impact your users' rights to privacy and data protection, and how your system is designed – if so – to prevent or mitigate the potential adverse outcomes of those interactions.

You are to answer all questions to the best of your knowledge, if necessary asking the relevant professionals in your undertaking before answering; some questions, though, allow you to answer "I do not know" (yet!), but please do mind – you are supposed to know.

Disclaimer

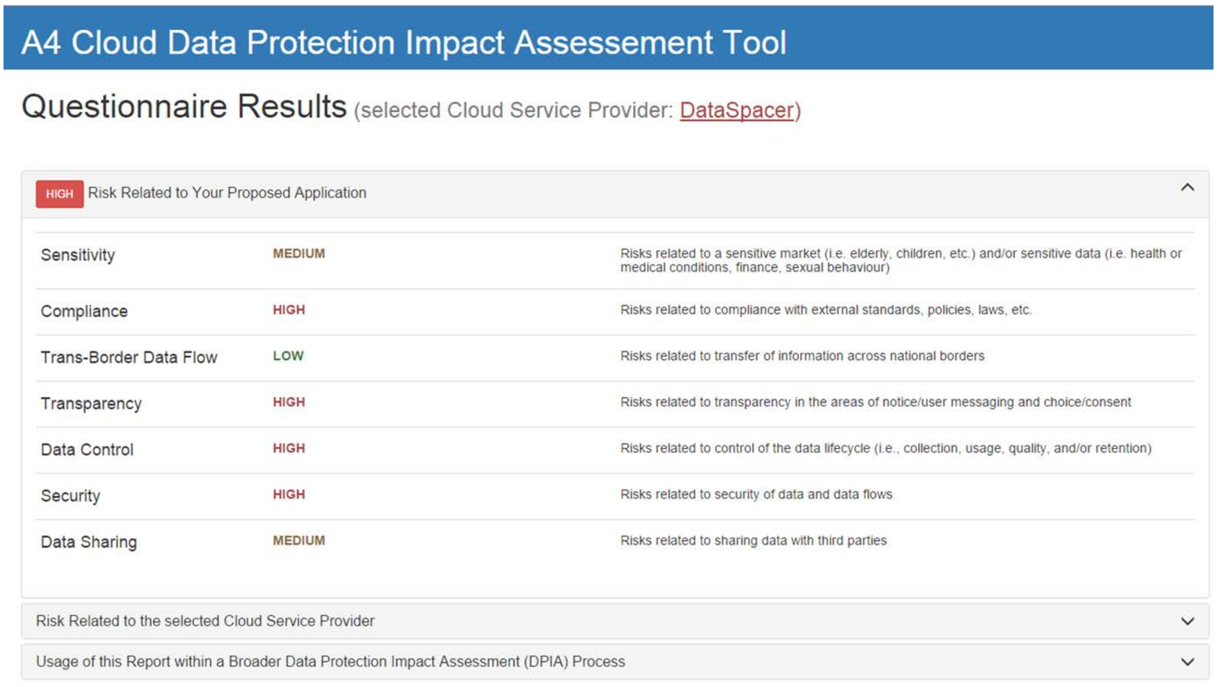
No information or content displayed in this tool should be construed, interpreted or relied upon as constituting legal advice, or a recommendation in respect of taking any course of action to comply with data protection laws, or legal obligations of any kind, and within any jurisdiction to which the European data protection law applies.

Nothing in this tool is intended to be an invitation or inducement to engage or enter into, or advice against engaging or entering into, an undertaking of any kind.

The content or information displayed in this tool is for general informational purposes regarding compliance with the applicable data protection laws only.

A4Cloud shall not be liable for any damages resulting from the use of the tool, including damages caused by any incorrectness or incompleteness of information provided on the tool. The content provided in the tool does not necessarily represent the

Figure 4: The DPIAT initial screen.



A4 Cloud Data Protection Impact Assessement Tool

Questionnaire Results (selected Cloud Service Provider: DataSpacer)

HIGH Risk Related to Your Proposed Application

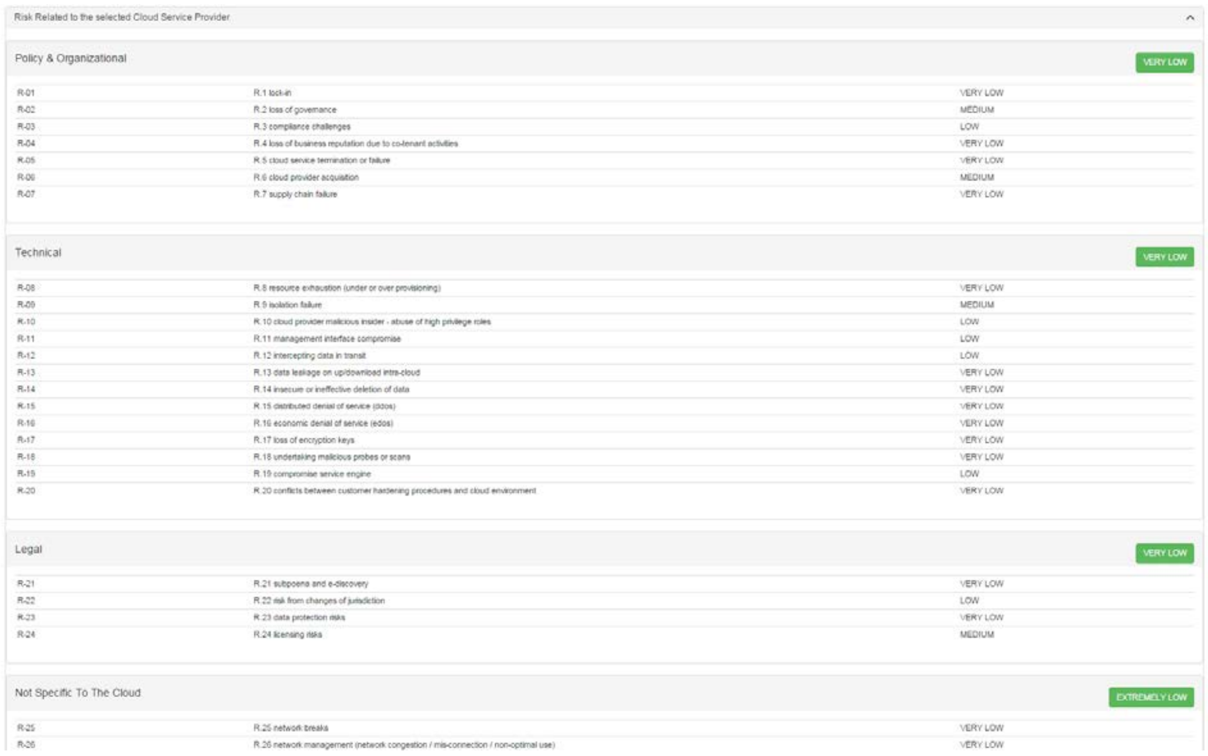
Sensitivity	MEDIUM	Risks related to a sensitive market (i.e. elderly, children, etc.) and/or sensitive data (i.e. health or medical conditions, finance, sexual behaviour)
Compliance	HIGH	Risks related to compliance with external standards, policies, laws, etc.
Trans-Border Data Flow	LOW	Risks related to transfer of information across national borders
Transparency	HIGH	Risks related to transparency in the areas of notice/user messaging and choice/consent
Data Control	HIGH	Risks related to control of the data lifecycle (i.e., collection, usage, quality, and/or retention)
Security	HIGH	Risks related to security of data and data flows
Data Sharing	MEDIUM	Risks related to sharing data with third parties

Risk Related to the selected Cloud Service Provider

Usage of this Report within a Broader Data Protection Impact Assessment (DPIA) Process

Figure 5: An example of the DPIAT output report –First Section: Risks related to the project.

The output of the DPIAT process is a report that includes the following information (an example of the DPIAT report is shown in Figure 5 and Figure 5): i) the data protection risk profile, ii) assistance in deciding whether to proceed or not, and iii) suggested mitigations. The report contains three (3) sections. The first, project-based risk assessment, is based on the answers to the questionnaire and contains the overall privacy impact score and several privacy indicator scores. The second part displays the risks based on the security controls used by the CSP. It contains the thirty five (35) ENISA risk scenarios with their associated quantitative and qualitative assessments. The last section provides additional information that the user needs to know about the DPIA process and information related to the GDPR article 33.



Risk Related to the selected Cloud Service Provider

Policy & Organizational		VERY LOW
R-01	R.1 lock-in	VERY LOW
R-02	R.2 loss of governance	MEDIUM
R-03	R.3 compliance challenges	LOW
R-04	R.4 loss of business reputation due to co-tenant activities	VERY LOW
R-05	R.5 cloud service termination or failure	VERY LOW
R-06	R.6 cloud provider acquisition	MEDIUM
R-07	R.7 supply chain failure	VERY LOW

Technical		VERY LOW
R-08	R.8 resource exhaustion (under or over provisioning)	VERY LOW
R-09	R.9 isolation failure	MEDIUM
R-10	R.10 cloud provider malicious insider - abuse of high privilege roles	LOW
R-11	R.11 management interface compromise	LOW
R-12	R.12 intercepting data in transit	LOW
R-13	R.13 data leakage on up/download intra-cloud	VERY LOW
R-14	R.14 insecure or ineffective deletion of data	VERY LOW
R-15	R.15 distributed denial of service (ddos)	VERY LOW
R-16	R.16 economic denial of service (edos)	VERY LOW
R-17	R.17 loss of encryption keys	VERY LOW
R-18	R.18 undertaking malicious probes or scans	VERY LOW
R-19	R.19 compromise service engine	LOW
R-20	R.20 conflicts between customer hardening procedures and cloud environment	VERY LOW

Legal		VERY LOW
R-21	R.21 subpoena and e-discovery	VERY LOW
R-22	R.22 risk from changes of jurisdiction	LOW
R-23	R.23 data protection risks	VERY LOW
R-24	R.24 licensing risks	MEDIUM

Not Specific To The Cloud		EXTREMELY LOW
R-25	R.25 network breaks	VERY LOW
R-26	R.26 network management (network congestion / mis-connection / non-optimal use)	VERY LOW

Figure 6: An example of the DPIAT output report – Second Section: Risks related to the selected Cloud Service Provider.

2.1.4 Tool Application Programming Interfaces

DPIAT, although being a standalone tool, offers a single interface, named *Idpiat*, which has the following two methods, as shown in Table 2.

Name of the API/method	Purpose of use	Consumed by	Data format	API format
Idpiat / Retrieve Questionnaire	Returns a single instance of a questionnaire	The UI of DPIAT (target all envisaged users of the tool)	JSON	RESTful
Idpiat / List Questionnaires	Returns a list of all available questionnaires (currently, there are two questionnaires available, the pre-screening and the screening ones)	The UI of DPIAT (target all envisaged users of the tool)	JSON	RESTful

Table 2: Interfaces and respective API methods offered by the Data Protection Impact Assessment Tool.

Although these interfaces are currently used internally by the UI of DPIAT, we are considering whether they can be public so that they will be consumed by other tools as well in the future, through a web service messaging and transport layer (such as a RESTful service).

DPIAT consumes the APIs provided by other tools and environments, as shown in Table 3.

Name of the API/method	Purpose of use	Provided by	Data format	API format
Questionnaire List	Retrieve a list of Questionnaires that can be chosen by the user to complete.	Questionnaire Provider, Certification Authorities	JSON	RESTful

Table 3: Interfaces and methods needed by the Data Protection Impact Assessment Tool.

2.2 Cloud Offerings Advisory Tool

2.2.1 Tool Overview and Target Stakeholders

The Cloud Offerings Advisory Tool (COAT) targets cloud customers, both SMEs and individuals, as well as SME cloud providers in assessing and selecting cloud offerings with respect to security and privacy requirements by providing information and guidance on:

- how to understand and assess what a cloud service provider is offering from a privacy and security perspective;
- how to compare offerings (from a data protection compliance and provider accountability point of view);
- the meaning of the comparison attributes, used to compare the offerings.

In more details, COAT assists the stakeholders in assessing the offerings of a cloud service provider, from a privacy and security perspective and compares offerings from various providers, from a data protection compliance and provider accountability point of view. It, also, provides guidance on the meaning of the comparison attributes and the education of users on security aspects, while it implements mechanisms, so that the offered advice and the user's decision are logged for accountability purposes.

2.2.2 High Level Architecture

Figure 7 shows the high level architecture of COAT, which is decomposed in the following main components, implementing respective process level mechanisms:

- The User Requirements Component, which gathers all the explicit (such as user requirements, based on criteria) and implicit (such as user location and contextual information) input for this tool.
- The Logging component, which logs the values of the user selections and the final report with the cloud offerings provided to him/her.
- The Information, Explanation and Guidance Component, which supports the guidance of the users on the explanation to each term or criterion used in the comparison process.
- The Matchmaker, which is further split into the Service Matchmaker module (matching the user requirements to service functional features) and the Contract Analysis Component (matching the user requirements to contract offerings).
- The Comparison Component, which produces a report with the best option for each criterion, by facilitating grouping and ranking of the offerings.

Through enabling the users expressing their requirements, COAT navigates them to the appropriate CSP to be chosen. The selection process strongly depends on the user's own requirements, so that the decision best addresses data protection issues and security needs by performing a comparative analysis of the Cloud service providers. Through this architecture, COAT will be able to implement a set of use cases, including the support for the decision making about cloud service offerings and the understanding of the contract terms of these service offerings.

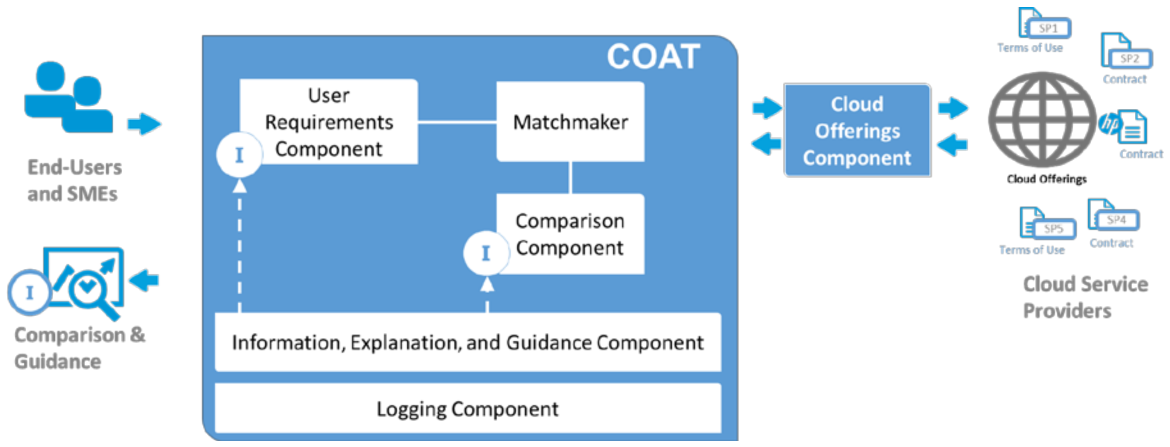
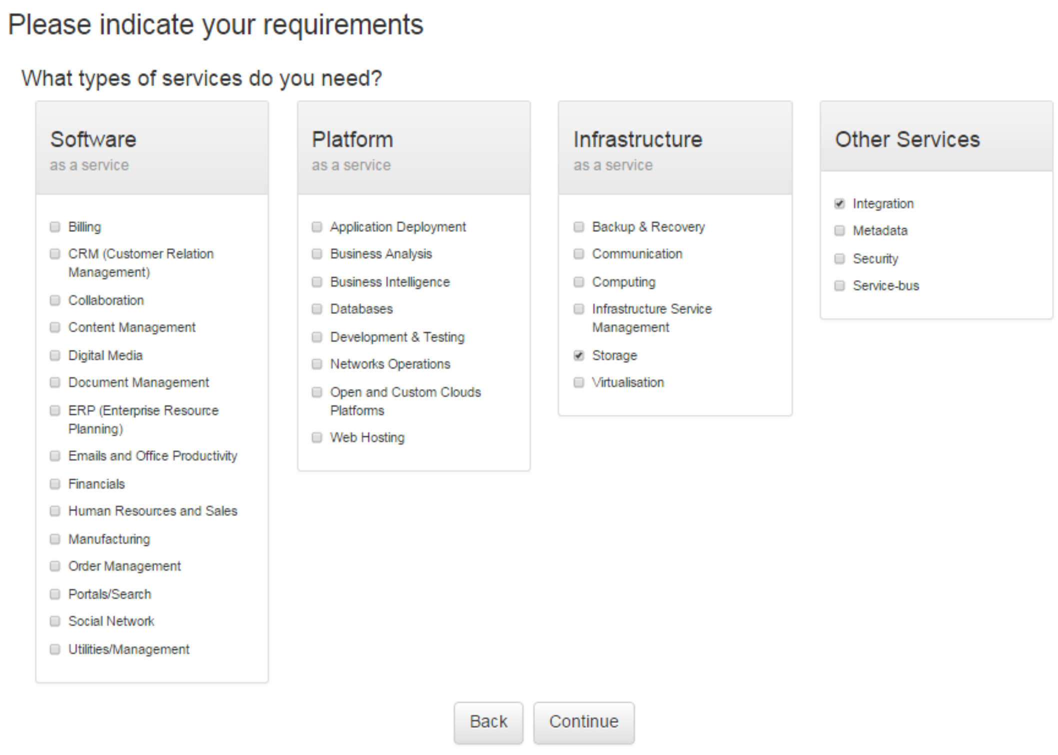


Figure 7: The high level architecture of the Cloud Offerings Advisory Tool.

2.2.3 User Interface

The tool offers a Web-based store interface, which allows users to indicate their requirements, through simply selecting the answers to a set of questions. To initiate a comparison of any Cloud Service Provider, the tool first asks for the types of cloud services that one wants to search for, as shown in Figure 8. Then, COAT provides guidance, through various questions designed to help identify the data privacy and security issues that are important for one's business or personal life, when selecting a Cloud Service Provider.

The result of this question-based navigation to the appropriate CSP is a shortlist of Cloud Offerings and a recommendation on which Cloud Service Providers' offer corresponds best to one's needs. This is shown in Figure 9.



Please indicate your requirements

What types of services do you need?

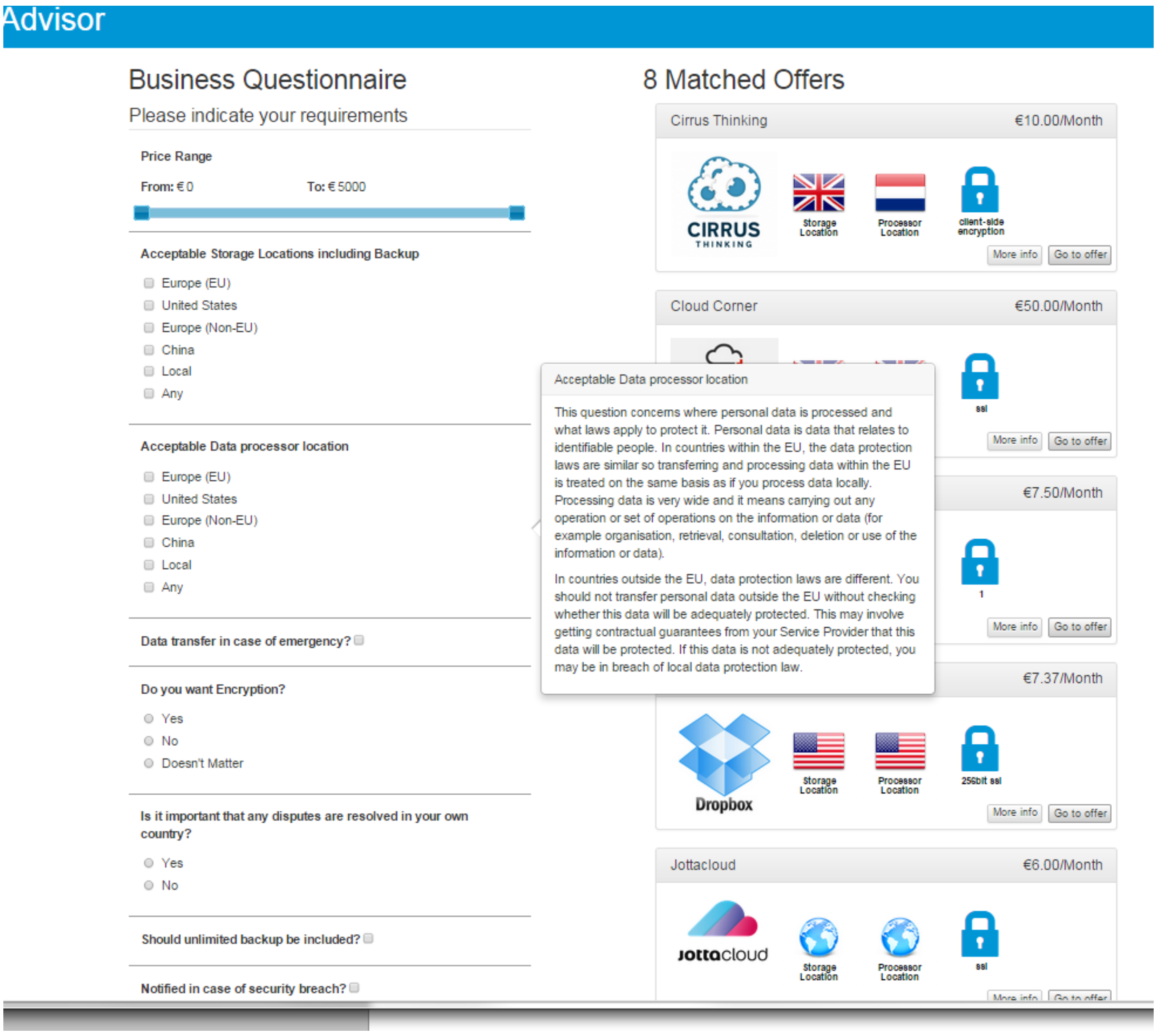
Software as a service
- Billing
- CRM (Customer Relation Management)
- Collaboration
- Content Management
- Digital Media
- Document Management
- ERP (Enterprise Resource Planning)
- Emails and Office Productivity
- Financials
- Human Resources and Sales
- Manufacturing
- Order Management
- Portals/Search
- Social Network
- Utilities/Management

Platform as a service
- Application Deployment
- Business Analysis
- Business Intelligence
- Databases
- Development & Testing
- Networks Operations
- Open and Custom Clouds Platforms
- Web Hosting

Infrastructure as a service
- Backup & Recovery
- Communication
- Computing
- Infrastructure Service Management
- [✓ Storage]
- Virtualisation

Other Services
- [✓ Integration]
- Metadata
- Security
- Service-bus

Figure 8: The use of COAT to define requirements.


Advisor

Business Questionnaire

Please indicate your requirements

Price Range

From: € 0 To: € 5000

Acceptable Storage Locations including Backup

- ☐ Europe (EU)
- ☐ United States
- ☐ Europe (Non-EU)
- ☐ China
- ☐ Local
- ☐ Any

Acceptable Data processor location

- ☐ Europe (EU)
- ☐ United States
- ☐ Europe (Non-EU)
- ☐ China
- ☐ Local
- ☐ Any

Data transfer in case of emergency? ☐

Do you want Encryption?

- ○ Yes
- ○ No
- ○ Doesn't Matter

Is it important that any disputes are resolved in your own country?

- ○ Yes
- ○ No

Should unlimited backup be included? ☐

Notified in case of security breach? ☐

Acceptable Data processor location

This question concerns where personal data is processed and what laws apply to protect it. Personal data is data that relates to identifiable people. In countries within the EU, the data protection laws are similar so transferring and processing data within the EU is treated on the same basis as if you process data locally. Processing data is very wide and it means carrying out any operation or set of operations on the information or data (for example organisation, retrieval, consultation, deletion or use of the information or data).

In countries outside the EU, data protection laws are different. You should not transfer personal data outside the EU without checking whether this data will be adequately protected. This may involve getting contractual guarantees from your Service Provider that this data will be protected. If this data is not adequately protected, you may be in breach of local data protection law.

8 Matched Offers

Cirrus Thinking	€10.00/Month
Cloud Corner	€50.00/Month
	€7.50/Month
	€7.37/Month
Jottacloud	€6.00/Month

Figure 9: Example of the resulting shortlist of cloud offerings provided by COAT.

2.2.4 Tool Application Programming Interfaces

COAT is a standalone tool. However, it offers a single interface named *Icoat,* which implements the API methods shown in Table 4.

Name of the API/method	Purpose of use	Consumed by	Data format	API format
Icoat/Retrieve Questionnaire	Returns a single instance of a questionnaire. In this case it is the requirements list shown for the user.	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/Manage Questionnaire	Stores, updates or deletes a single instance of a questionnaire for future reference	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/List Questionnaires	Returns a list of all available questionnaires in the system	The UI of COAT	JSON	RESTful

Name of the API/method	Purpose of use	Consumed by	Data format	API format
		(target all envisaged users of the tool)		
Icoat/Matches	Accepts a JSON representation of the MatchCriteria object and returns a list of matching Service offerings	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/SOLRClient	Search for and retrieve records from a search engine implemented in SOLR [6], based on the match criteria created.	The UI of COAT (target all envisaged users of the tool)	XML /JSON implemented through SOLR structure	RESTful

Table 4: Interfaces and respective methods offered by the Cloud Offerings Advisory Tool.

Although these interface methods are currently used internally by the UI of COAT, we are considering whether they can be public so that they will be consumed by other tools as well in the future, through a web service messaging and transport layer (such as a RESTful service).

COAT may consume interface methods that could by potentially provided by cloud providers. Such APIs provided by other tools and environments are shown in Table 5.

Name of the API/methods	Purpose of use	Should be offered by	Data format	API format
Countries List	Retrieve a list of countries that are available to Service Providers when adding their services. Used as options for the end user to choose from where to select their location	Cloud Providers	JSON	RESTful
Service Types List	Retrieve a list of service types that are available to Service Providers when adding their services. Allows the end user to filter offers by service type when selecting their questionnaire	Cloud Providers	JSON	RESTful
Questionnaire List	Retrieve a list of Questionnaires that can be chosen by the user to complete	Questionnaire Provider, Certification Authorities	JSON	RESTful
Offer Detail	Shows a detailed view of an offer	Cloud Providers	JSON	RESTful

Table 5: Interfaces and methods needed by the Cloud Offerings Advisory Tool.

3 Policy Definition and Enforcement

This functional area supplements the previous one of Section 2 in the preventive role of the A4Cloud tools to support accountability. The set of tools belonging to this category facilitate prevention of the loss of data governance in complex cloud service provision chains through the implementation of mechanisms for the provision and support of enforceable accountability policies. Such mechanisms facilitate the definition, validation, storage, enforcement and overall management of accountability policies.

For the instantiation of the Cloud Accountability Reference Architecture in this functional area, three software tools are provided: AccLab (Accountability Lab tool), Data Protection Policies Tool (DPPT) and A-PPLE (Accountable Primelife Policy Engine).

3.1 Accountability Lab

3.1.1 Tool Overview and Target Stakeholders

The purpose of the Accountability Lab (AccLab) tool is to bridge the gap between the human-readable, but abstract, accountability obligations expressed in a policy specification language, which in our case is the Abstract Accountability Language (AAL) and the concrete, machine-readable, accountability policies expressed in Accountable Primelife Policy Language (A-PPL). More specifically, AccLab reasons about accountability at design time, makes a link between legal and technical aspects and guides users to write abstract accountability policies. Thus, it provides facilities to write abstract accountability obligations in AAL via a “smart wizard” user interface, which can then be (semi-) automatically translated to A-PPL policies, through DPPT, and be enforced by a policy enforcement engine, like A-PPLE.

This tool is used by the privacy officers of the cloud providers, being either data controllers or processors, to express the accountability policies in an abstract form, and the policy consumers (such as other cloud providers or cloud customers) that need to agree on a policy to express their data-handling preferences, attached to this specific policy. The tool gets as input the abstract textual definition of the policy in AAL format, expressing obligations (for the privacy officer of the provider issuing the policy) or the preferences (for the privacy officer of the actor accepting the policy). The tool internally processes the obligations and identifies the mapping between these obligations to policy terms and rules. Along this process, AccLab checks the AAL statements for consistency and correctness, while compliance checking is applied to determine whether one obligation is stronger than another.

3.1.2 High Level Architecture

Figure 10 shows the high-level architecture view of the AccLab. The natural obligations, i.e. textual sentences about laws, regulations and norms for data privacy preferences, are integrated with the components of a system that is designed using an accountability components diagram. The kernel of the AccLab is the AAL language (AAL code) and its semantics (Temporal logic). AAL expressions are interpreted, checked, and mapped into A-PPL rules. The second part of the kernel is a connector, which can interact with a set of other tools to enable monitoring, logging, a policy engine connection, or potentially any external auditing. Through this architecture, the AccLab provides compliance checking between AAL statements and A-PPL policy rules, so that cloud providers can reason on the consistency of the machine-readable A-PPL accountability policies with the expressed AAL-based accountability obligations of the involved providers.

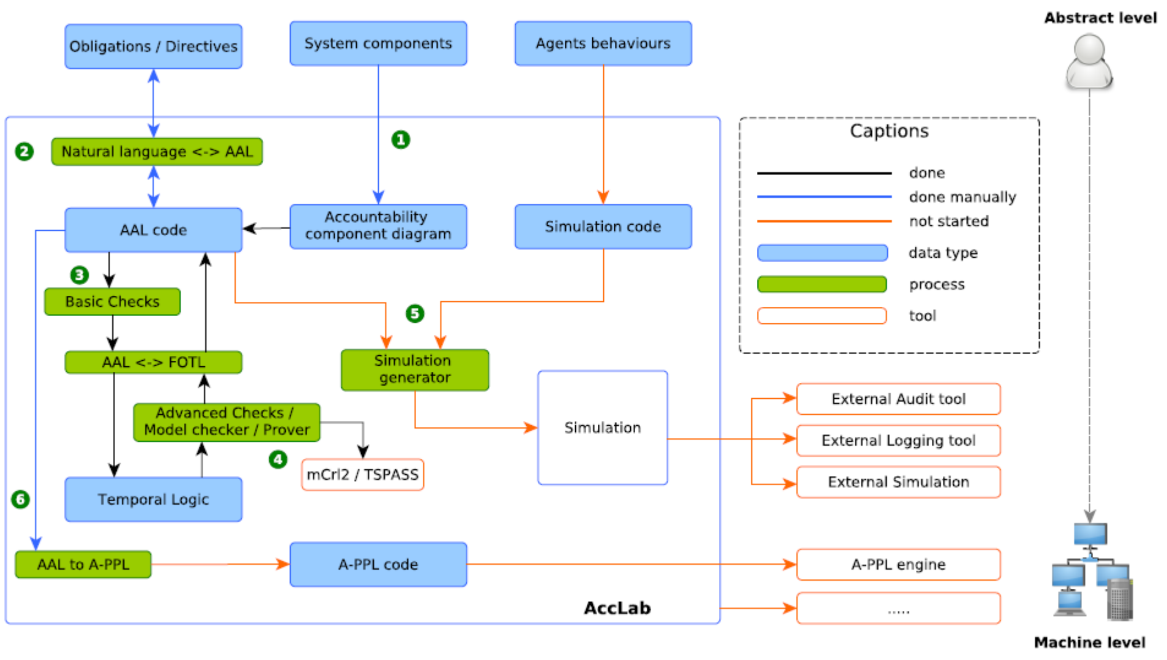
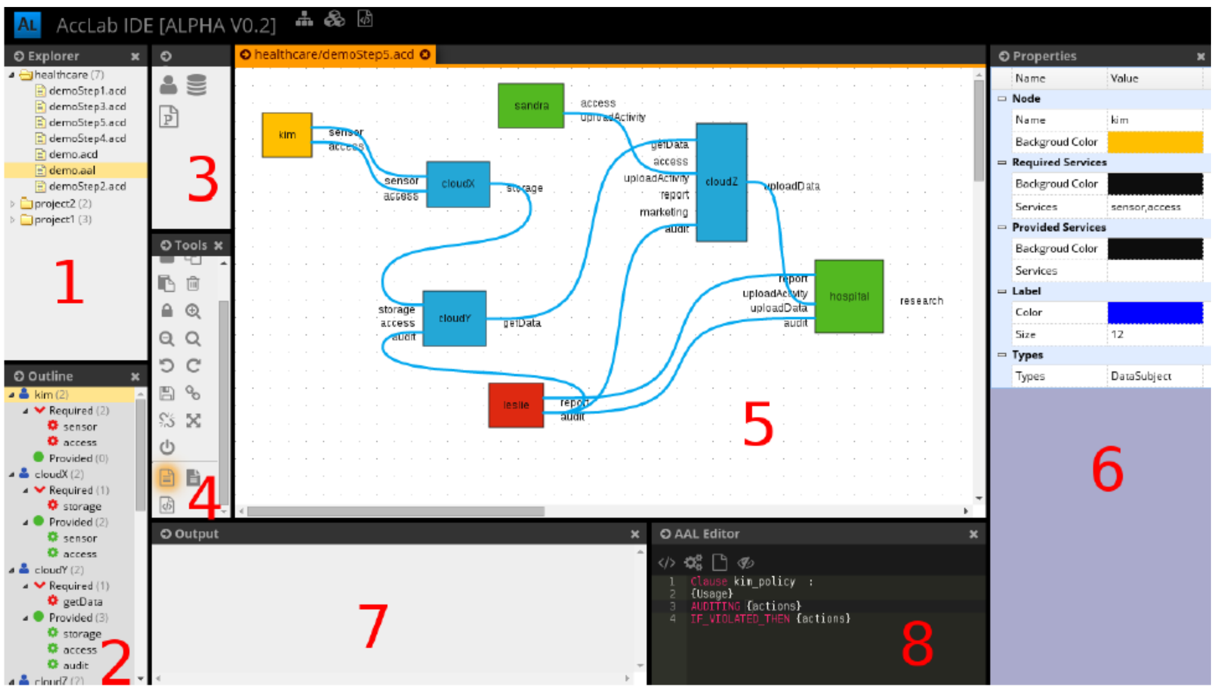


Figure 10: The high level architecture of the Accountability Lab.

3.1.3 User Interface

AccLab offers a Web User Interface, which is shown in Figure 11 and it is called the Component Editor.



```
Clause kim_policy :
{Usage}
AUDITING {actions}
IF_VIOLATED_THEN {actions}
```

Figure 11: The UI of the Accountability Lab tool.

This UI involves the following seven views:

i. The Explorer: This panel contains a tree view of the workspace;
ii. Outline: The outline contains a tree view of the current components (with blue icon) in the opened workspace file, and for each agent its required services (in red icon) and provided services (green icon).
iii. Components: Contains the elements that can be used in the diagram

iv. Tools: A panel containing different tools to be used while editing the workspace files.
v. Diagram: The current component diagram.
vi. Properties: A grid containing the properties of a selected element in the diagram. The user can edit its name, style properties (color, font, etc) and, also, types and services.
vii. Output: The output where we show the result coming from the back-end.
viii. AAL editor: Allows to quickly edit a component's policy, before generating the AAL program.

3.1.4 Tool Application Programming Interfaces

The AccLab tool offers an API relevant to the functionality exposed by the AAL Checker, as shown in Table 6.

Name of the API	Purpose of use	Consumed by	Data format	API format
AAL Parser	Parses an AAL program and performs some checks	The UI of AccLab (target all envisaged users of the tool)	JSON	RESTful

Table 6: Interfaces offered by the Accountability Lab tool.

Currently, AccLab does not consume any external API by other tools and environments.

3.2 Data Protection Policies Tool

3.2.1 Tool Overview and Target Stakeholders

The Data Protection Policies Tool (DPPT) is used by Cloud Providers controlling and/or processing personal data (Personally Identifiable Information – PII) to achieve the following three goals:

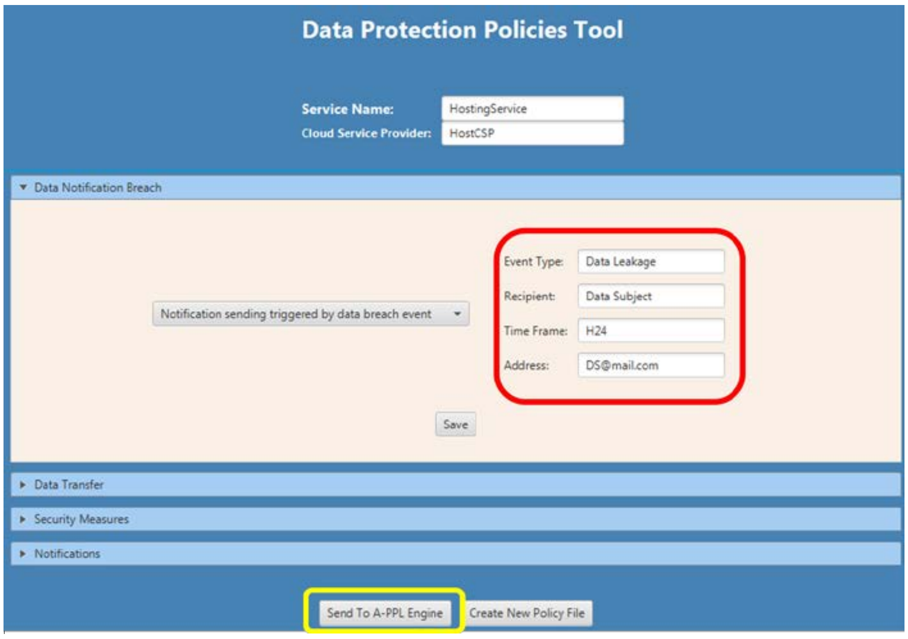
- to create a machine readable representation of the policy statements;
- to translate the policy statements into an enforceable language (i.e. A-PPL);
- to send the policy to the component in charge of the policy enforcement (i.e. A-PPL Engine).

The first goal is achieved by creating an ontology-based representation of the policy statement. This is enabled by allowing the users to select actions, which are automatically translated into instances of ontology concepts (in an .owl format). After that, DPPT creates an A-PPL representation by instantiating templates that have been bound to the policy statements shown in the GUI. Each template is customised taking into account data provided by the tool user.. Finally, the created policy elements are assembled into a valid A-PPL policy, which is sent to a policy enforcement engine, like A-PPLE. The A-PPL policy corresponds to the obligations defined through the AAL clauses, used in AccLab.

3.2.2 User Interface

DPPT is a standalone tool, which offers a graphical interface for the policy administrators of the involved cloud providers to define the different aspects of a data protection policy, as shown in Figure 12. Each panel view represents a section pertaining to a specific aspect of the data protection policy, reflecting the structure of a Privacy Level Agreement template [7]. More specifically, the red frame highlights the fields that the user has to fill in to enable the creation of the ontology-based representation and the A-PPL policies.

The yellow frame shows the button that has to be used to send the policy to the respective enforcement engine, which, also, wraps the policy as a PII element, as required by the A-PPL Engine APIs.


Data Protection Policies Tool

Service Name: HostingService
Cloud Service Provider: HostCSP

Data Notification Breach

Notification sending triggered by data breach event

Event Type: Data Leakage
Recipient: Data Subject
Time Frame: H24
Address: DS@mail.com

Figure 12: The Graphical User Interface of the Data Protection Policies Tool.

3.2.3 Tool Application Programming Interfaces

DPPT is a standalone tool. To achieve the third goal, the tool has been extended to interface with a policy enforcement engine. For our case the API of A-PPLE is used, as shown in Table 7.

Name of the API/methods	Purpose of use	Offered by	Data format	API format
Store Policy	Store the A-PPL policy created by DPPT into the policy repository of the policy enforcement tool so that this latter is setup to enforce the policies.	A-PPLE	XML (A-PPL)	RESTful

Table 7: Interfaces and methods optionally needed by the Data Protection Policies Tool.

3.3 Accountable Primelife Policy Engine

3.3.1 Tool Overview and Target Stakeholders

The Accountable Primelife Policy Engine (A-PPLE) is an extension of the PPL engine initially designed in the PrimeLife project[1] with additional modules that enable accountability features. The main role of the original PPL engine was to enforce privacy policies related to personal data handling. A-PPLE extends the PPL engine with functionality that enables the enforcement of accountability obligations defined in the A-PPL language.

The target user group of A-PPLE is the set of cloud actors, who control or process personal and/or confidential data (thus cloud providers being either data controllers or data processors). The tool receives the accountability policy in A-PPL format and associates the policy rules and actions to the PII disclosed and stored together with the data in a PII repository (PII Store). From the time that this PII is created and onwards, and for as long as a cloud provider holds a copy of those data, any access

[1] http://primelife.ercim.eu/

requests to the data are regulated by A-PPLE, which enforces all data handling rules and obligations specified in the A-PPL policy. A-PPLE relies on data transfer logs and evidence repositories populated by third-party tools in order to identify operations on data covered by a policy. These operations occur outside of A-PPLE reach and ultimately determine whether a policy violation has occurred.

3.3.2 High Level Architecture

Figure 13 shows the high level view of A-PPLE. As shown there, the design specification of A-PPLE adopts a two-layer architecture to enforce isolation of engine components and data. The core elements of the policy engine providing the enforcement functionality reside in the Business layer. All personal data and their associated policies stored in the PII Store reside in the Persistence layer. Access to the persistence layer is mediated through a Persistence Handler component, which abstracts the underlying location and storage data model to the Business layer functions above. The architecture defines the Policy Decision Point (PDP) as the central element of A-PPLE, responsible for taking usage control decisions, with regards to a data usage request.

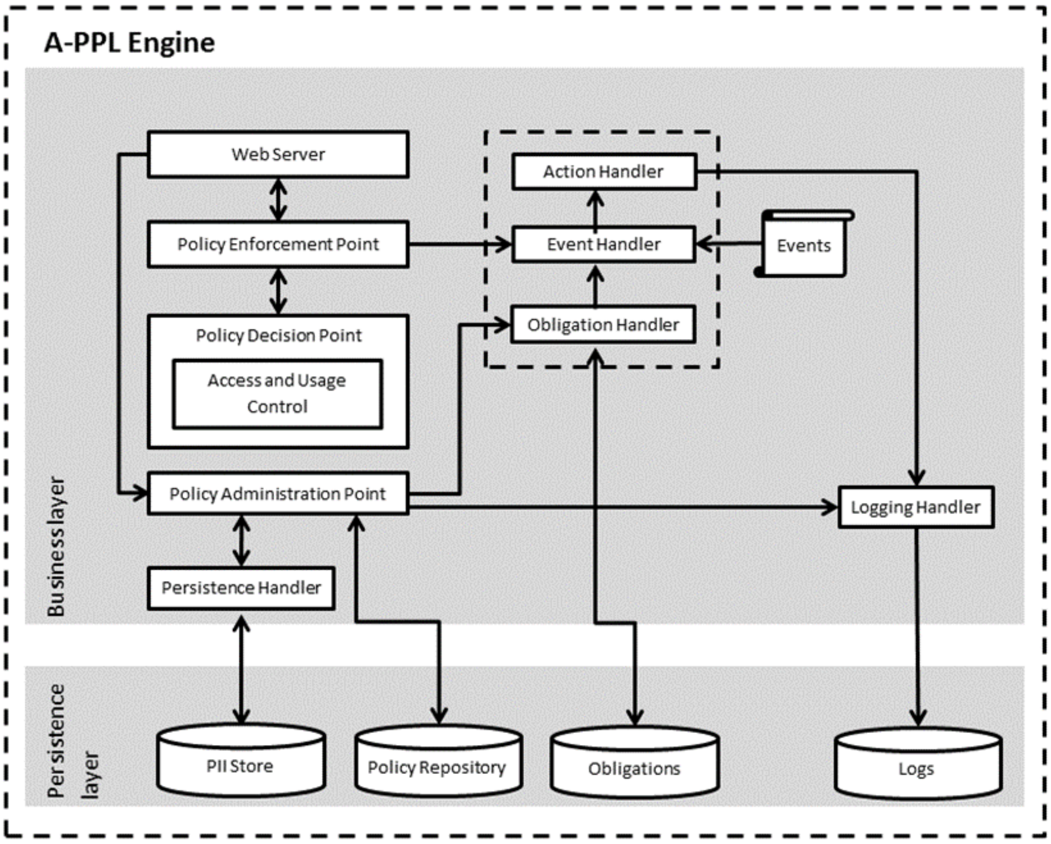


Figure 13: High level architecture of the Accountable Primelife Policy Engine.

More specifically, A-PPLE consists of the following main components:

- The Policy Decision Point (PDP), which is responsible for taking usage control decisions.
- The Policy Enforcement Point (PEP), which enforces the PDP decisions with respect to allowing usage of a piece of data. The PEP acts as an orchestrator of the enforcement process, orchestrating two modules, namely the Action and Obligation Handlers.
- The Policy Administration Point (PAP), which records the obligations associated with the PIIs.
- The Obligation Handler, which executes A-PPL obligations related to the personal data handling and is complemented by the Event Handler (serving as the entry point to trigger the events which are responsible for the event based triggers) and the Action Handler (facilitating actual action execution)

- The Logging Handler, which provides an extensible secure logging interface to support all logging related functionality during the policy enforcement and personal data handling process.
-

These components are complemented by the Policy Repository, which provides storage for generic A-PPL policies that the engine will retrieve and possibly modify, based on the context of the particular data disclosure (such as the user preferences).

A-PPLE does not offer any Web interface. The tool is configured to enforce accountability policies, in A-PPL, which can be produced by policy definition tools, like DPPT (see section 3.2). Along with the A-PPL policies, the engine gets as input events (signifying trigger actions as described in obligations), relevant evidence from the environment with respect to actions occurred and data access request. The tool analyses the policies by retrieving access and usage control rules and enforces the relevant obligations described in them, through analysing triggers and actions. It, also, provides configuration of handlers for events monitoring and obligations, collection of evidence (such as the logs created by A-PPLE) and message exchanges related to event handling and action executions.

3.3.3 Tool Application Programming Interfaces

A-PPLE provides APIs, which enable interaction with the operations of the business applications. During these interactions, end users can indirectly access the relevant API calls offered by A-PPLE and the respective *IAppEngine* Interface, as shown in Table 8.

Name of the API/method	Purpose of use	Consumed by	Data format	API format
IAppEngine / storePii	Stores PII inside the PII repository, together with attached A-PPL policy that defines how the data can be processed	The application (for the Data Controller)	XML-based A-PPL specification	RESTful
IAppEngine / deletePii	Deletes PII from the PII repository	The application (for the Data Controller)	JSON	RESTful
IAppEngine / getPii	Retrieves specific PII given a set of attributes	The application (for the Data Controller)	JSON	RESTful
IAppEngine / getAllPii	Retrieves all PII given a specific owner	The application (for the Data Controller to be finalised exposed to Data Subjects through DT)	JSON	RESTful
IAppEngine / updatePii	Updates/corrects specific PII's value given a set of attributes	The application (for the Data Controller/Data Subject)	JSON	RESTful
IAppEngine / deleteAllPii	Deletes all PII of a user from the PII store	The application (for the Data Controller/Data Subject)	JSON	RESTful
IAppEngine / storePolicy	Stores a policy template	The application (for the Data Controller/ potentially for Data Subject)	XML-based A-PPL specification	RESTful
IAppEngine / getPolicy	Retrieves a policy template	The application (for the Data	XML-based A-PPL specification	RESTful

Name of the API/method	Purpose of use	Consumed by	Data format	API format
		Controller/ potentially for Data Subject)		
IAppEngine / deletePolicy	Deletes a policy template	The application (for the Data Controller)	JSON	RESTful
IAppEngine / requestPii	Used for downstream usage. A third party Data Processor can request access to a piece of personal data.	The application (for the Data Processor)	XML-based A-PPL specification	RESTful
IAppEngine / triggerUserRegistration	A-PPLE is notified when personal data of a data subject is collected for the first time	The application (for the Data Controller)	JSON	RESTful
IAppEngine / registeredusers	A-PPLE retrieves the list of data subjects (PII owner identifiers) about whom personal data is currently stored in the engine	IMT	JSON	RESTful
IAppEngine / notification	A-PPLE informs a particular data subject about a policy violation, i.e. incident.	IMT	JSON	RESTful
IAppEngine / notification/all	A-PPLE informs all registered data subjects about a policy violation, i.e. incident.	IMT	JSON	RESTful

Table 8: Interfaces and the respective methods offered by the Accountable Primelife Policy Engine.

A-PPLE consumes the APIs provided by other tools and environments, as shown in Table 9.

Name of the API	Purpose of use	Should be offered by	Data format	API format
Store encrypted logs	The API of TL sender module of Transparency log. This is used to log a message using a pile identifier (“apple”) to a given recipient (PII owner identifier)	TL	JSON	RESTful
Send Incidents	This API is used to communicate the incidents detected from A-PPLE to the relevant stakeholders of the cloud providers	IMT	JSON	RESTful

Table 9: Interfaces and methods needed by the Accountable Primelife Policy Engine.

4 Evidence and Validation

The Evidence and Validation functional area offers accountability support in both a preventive and detective manner. The set of tools belonging to this category provide an assertion that the prevention mechanisms against the loss of data governance in complex cloud service provision chains are enforced at runtime. These tools, also, provide mechanisms for the monitoring of the appropriate software resources to control and verify the accountability policy-based operation of these chains. More specifically, the tools in the Evidence and Validation area fulfil two main purposes:

- They provide means for the audit of cloud applications and infrastructures during their execution. This objective principally requires that appropriate auditing policies can be defined, data can be monitored during execution and evidence can be stored, and that auditing properties can be verified at appropriate times during execution.
- They enable the validation of A4Cloud tools. Properties to be validated include, for example, that actions of one tool do not invalidate hypotheses needed for the application of another and that information is passed correctly between tools.

The respective mechanisms to meet these objectives are being developed within the scope of the A4Cloud software tools, namely the Audit Agent System (AAS), the Data Transfer Monitoring Tool (DTMT) and the Assertion Tool (AT).

4.1 Audit Agent System

4.1.1 Tool Overview and Target Stakeholders

The Audit Agent System (AAS) tool is the evidence collection and audit service in A4Cloud, which implements the Framework of Evidence (please refer to WP:C8 work), its components, mechanisms and processes in an efficient and scalable way. It enables the automated audit of multi-tenant and multi-layer cloud applications and cloud infrastructures for compliance with consumer-defined policies, using software agents. Agents can be deployed at different cloud architectural layers (i.e., network, host, hypervisor, IaaS, PaaS, SaaS) with the purpose of i) collecting and processing evidence, ii) generating audit reports and iii) aggregating new evidence. AAS uses audit tasks that specify the data collection sources and tools to use to collect data, and policies to specify the thresholds and constraints, against which the evidence is examined to generate the audit results.

AAS is used by internal and external auditors to perform continuous and periodic audits.

4.1.2 High Level Architecture

Figure 14 shows the high level view of the AAS architecture, with the following components:

- Audit Policy Module (APM): The APM is used by the auditor to define audit policies. On the basis of these policies (describing the goal of what needs to be checked - tasks which shall be performed during audits and thresholds for compliance and failure respectively), a suitable set of audit tasks is selected by the Auditor and configured properly. These Audit Tasks are managed by the AAC.
- Audit Agent Controller (AAC): The AAC prepares audit agents according to the audit task description and manages their life-cycle from configuration, deployment, migration between agent containers to deletion. Evidence collected by agents is stored securely in the evidence store.
- Evidence Sources: The cloud stack consists of several architectural layers, on which evidence can be collected (i.e., network, host, hypervisor, IaaS, PaaS, SaaS and the cloud management system). For each evidence source (e.g., logs, Cloud Management System API, but also other A4Cloud tools such as A-PPL-E and DTMT etc.), a specialized Collection Agent is implemented. The Collection Agent transforms raw data (such as logs) into the Evidence Record Format (as defined in WP:C-8).
- Evidence Store (ES): The Evidence Store is used to store data collected by the agents for audit processing and report generation by the Evidence Processor and Presenter, which logically groups audit evaluation and reporting agents. The Evidence Store is encrypted, and there is isolation

between tenants. Encryption, isolation and preservation of evidence integrity are achieved by using TL as the underlying technology for secure evidence storage in the Evidence Store (see Section 5.3). The TL recipient is the evaluation agent. Therefore, the evaluation agent can decrypt records collected for it. Every tenant has its own evidence store.

- Evidence Processor & Presenter (EPP): The Evidence Processor & Presenter is a logical component running audit evaluation and reporting agents. Audit results are generated using evidence stored in the ES. The results are presented to the auditor primarily in the form of reports on a web-based dashboard. Additionally, individual evidence records can be requested via the dashboard. The same information is also made available to other tools by pushing notifications using Notification Agents (e.g., IMT Notification Agent).

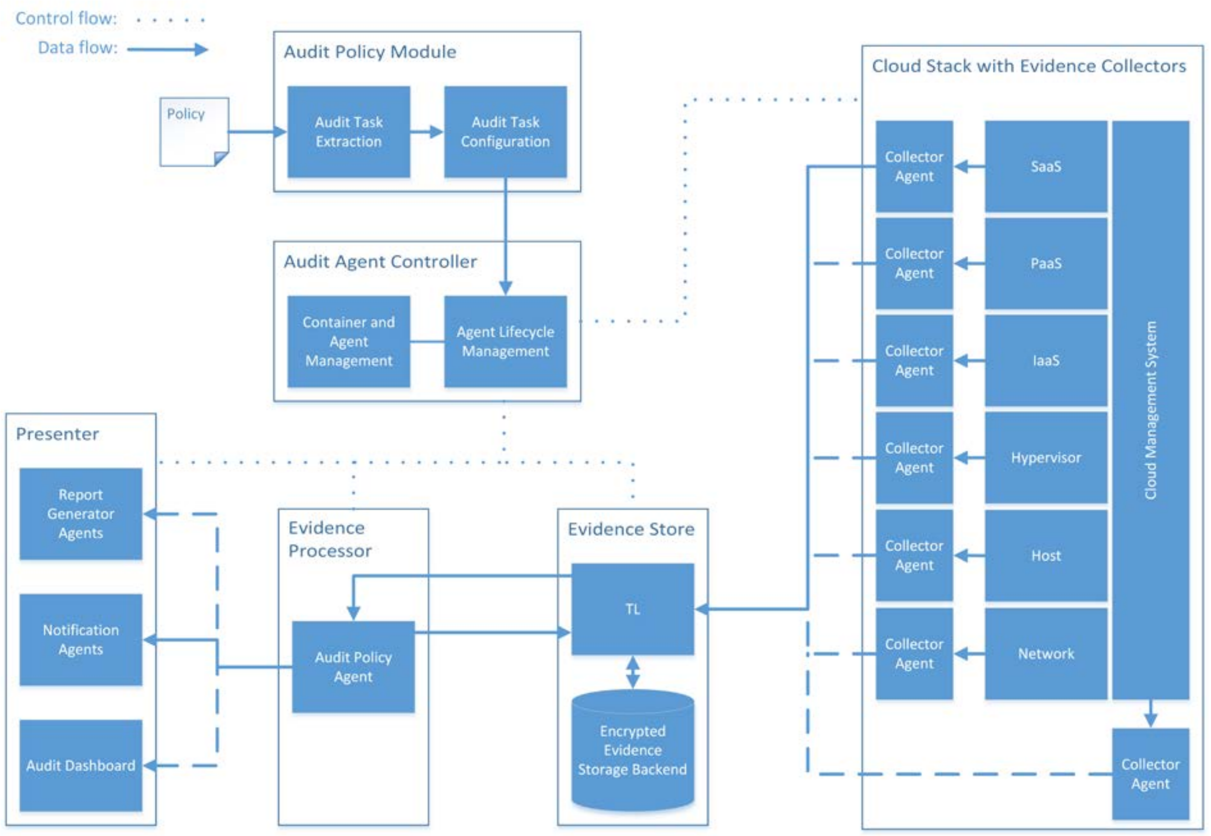


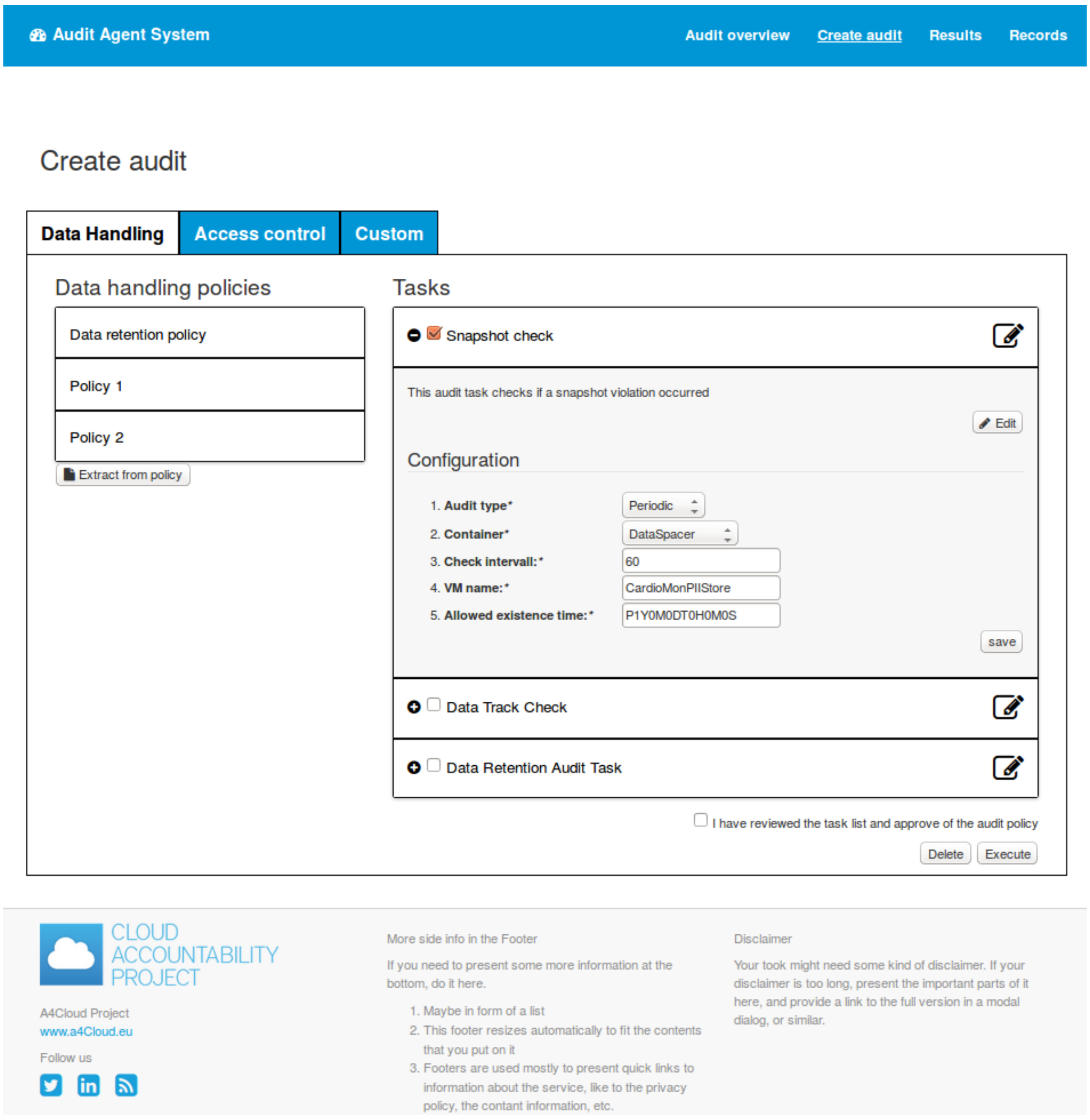
Figure 14: The high level architecture of the Audit Agent System tool.

4.1.3 User Interface

The tool receives as input the policy to audit (in a machine-readable format, A-PPL) and further audit configuration properties (e.g., audit type, interval and further parameters that cannot be extracted from the policy).

The specification of audit tasks is done by an auditor using a Web-based User Interface, as shown in Figure 15.

The output of the AAS tool is a report containing the compliance status (check results) and supporting proof/evidence for the compliance claim. Additional output may also be (aggregated) evidence from the data gathered during the audit process (e.g., output of intermediate results for further evaluation). The report is presented to the auditor in the form of Web dashboard.



Audit Agent System — Audit overview | Create audit | Results | Records

Create audit

Data Handling | Access control | Custom

Data handling policies

- Data retention policy
- Policy 1
- Policy 2

Extract from policy

Tasks

Snapshot check

This audit task checks if a snapshot violation occurred

Edit

Configuration

1. **Audit type***	Periodic
2. **Container***	DataSpacer
3. **Check intervall:***	60
4. **VM name:***	CardioMonPIIStore
5. **Allowed existence time:***	P1Y0M0DT0H0M0S

save

Data Track Check

Data Retention Audit Task

I have reviewed the task list and approve of the audit policy

Delete | Execute

CLOUD ACCOUNTABILITY PROJECT

A4Cloud Project
www.a4Cloud.eu

Follow us

More side info in the Footer

If you need to present some more information at the bottom, do it here.

1. Maybe in form of a list
2. This footer resizes automatically to fit the contents that you put on it
3. Footers are used mostly to present quick links to information about the service, like to the privacy policy, the contant information, etc.

Disclaimer

Your took might need some kind of disclaimer. If your disclaimer is too long, present the important parts of it here, and provide a link to the full version in a modal dialog, or similar.

Figure 15: An example view of the AAS dashboard (for the case of a data retention audit task).

4.1.4 Tool Application Programming Interfaces

The interaction of the auditors, as the end users of the AAS tool, and the tool itself, is based on a set of UI functions that are the outcome of relevant API calls to the interface methods offered by AAS. These interface methods can be broadly categorized into being audit/AAS-specific and evidence-specific. Evidence-specific interface methods are closely related to the functionalities described in the Framework of Evidence (please refer to the work in WP:C-8) and the collection and storage of evidence, whereas audit/AAS-specific interface methods provide an audit-focused layer on top of the evidence collection functionality. The latter interface methods also include the definition and evaluation of audit policies as well as the presentation of audit results. Table 10 contains a summary of the AAS REST interface broadly categorized functions related to evidence collection and audits (*IAudit* interface), runtime monitoring of the AAS platform (*IMonitor* interface) and evidence functionality (*IEvidence* interface).

Name of the API/method	Purpose of use	Consumed by	Data format	API format
IAudit				
extractFromPolicy	Request the extraction of policy file(s) for task configuration purposes (policy dir. in AAS configuration file).	The UI of the AAS (for Auditor)	XML	RESTful
setAuditTask	Definition of audit task to be performed for checking compliance with audit policy, through various parameters detailed in the tool specification handbook.	The UI of the AAS (for Auditor)	JSON	RESTful
requestAuditReport	Requests a specific audit result of a single audit task	The UI of the AAS (for Auditor)	JSON	RESTful
requestAllAuditReports	Requests all audit reports persisted by AAS	The UI of the AAS (for Auditor)	JSON	RESTful
deleteAuditTask	Deletes an currently running audit task	The UI of the AAS (for Auditor)	JSON	RESTful
IMonitor				
requestAuditTaskConfiguration	Requests the configuration for a given audit task	The UI of the AAS (for Auditor)	JSON	RESTful
requestRunningAgents	Requests all agents running on the platform (i.e., all containers)	The UI of the AAS (for Auditor)	JSON	RESTful
requestRunningAgentsByAuditTask	Requests the running agents for a given audit task	The UI of the AAS (for Auditor)	JSON	RESTful
requestRunningAgentsByContainer	Requests the agents running in a given container	The UI of the AAS (for Auditor)	JSON	RESTful
requestRunningAuditTasks	Requests the currently executed audit tasks	The UI of the AAS (for Auditor)	JSON	RESTful
requestRunningContainers	Requests the running JADE agent containers (core and clients) inside an AAS platform	The UI of the AAS (for Auditor)	JSON	RESTful
IEvidence				
requestEvidenceObject	Requests a specific EvidenceRecordfrom the evidence repository, given an identifier of the record	The UI of the AAS (for Auditor)	XML (transformed to JSON)	RESTful
requestAllEvidenceObjects	Requests a collection of all records persisted by AAS	The UI of the AAS (for Auditor)	XML (transformed to JSON)	RESTful
requestEvidenceObjectsByTimeframe	Requests a collection of all task-related records persisted by AAS in a specified timeframe	The UI of the AAS (for Auditor)	XML (transformed to JSON)	RESTful

Name of the API/method	Purpose of use	Consumed by	Data format	API format
requestEvidenceObjectsByPolicy	Requests a collection of all task-related records persisted by AAS	The UI of the AAS (for Auditor)	XML (transformed to JSON)	RESTful

Table 10: Interfaces and respective API methods offered by the Audit Agent System.

Furthermore, AAS consumes the APIs provided by other tools and environments, as shown in Table 11.

Name of the API	Purpose of use	Should be offered by	Data format	API format
Send Incidents	This API is used to communicate the incidents detected from AAS to the relevant stakeholders of the cloud providers	IMT	JSON	RESTful
Store Evidence	Persists an evidence record object gathered by AAS in TL (used as evidence store)	TL	XML	RESTful
Read Evidence	Reading evidence persisted by AAS before, using the API of TL (used as evidence store)	TL	XML	RESTful
Read A-PPLE logs	AAS is processing logs created by A-PPLE for different audit tasks.	TL (logged by A-PPL-E)	JSON	RESTful
Get Policy (IAppEngine / getPolicy)	Retrieves a policy template	The application (for the Data Controller/ potentially for Data Subject)	XML-based A-PPL specification	RESTful

Table 11: Interfaces and methods needed by the Audit Agent System.

4.2 Data Transfer Monitoring Tool

4.2.1 Tool Overview and Target Stakeholders

The Data Transfer Monitoring Tool (DTMT) aims to enable cloud service providers to demonstrate compliance with personal data protection and other regulations regarding where and by whom the processing occurs. The tool automates the collection of evidence about how data transfers comply with the personal data handling obligations, by generating logs regarding the operations performed on personal data involving transfers at the infrastructure level (for example when performing load-balancing or creating backups and storing them in different hosting machines). It, then, analyses these logs using rules, helping an auditor to identify whether all transfers were compliant with the authorisations from the Data Protection Authority being obtained beforehand by the Data Controller. In this way, policy violations can be identified.

Part of the necessary information to monitor data transfers can be obtained from machine-readable policies specifying accountability obligations, such as A-PPL policies. Furthermore, the tool enables manual configuration for constraints about which parties, and geographical locations are allowed for a given data set.

Upon identification of a potential violation (by the tool), further data can be collected to provide supporting evidence of an incident. Notifications can be triggered by the auditor. The tool may rely on A-PPLE to carry out actions for policy enforcement related to the events it generates and on other tools (i.e. IMT) to act upon the detected violations (e.g. for notification or remediation).

The tool is configured with information for the geographical location of host machines in order to determine the current physical location of the data. This information is not part of the A-PPL policy statement, but must be provided by the data processors (such as IaaS providers), in order for the monitoring part of the DTMT to work properly.

The data controller needs to feed the DTMT with the authorisations made to data processors and other third parties that handle personal data. The data controller creates a configuration file describing these authorisations, using information obtained from the data processors (who can further delegate the processing to other parties with the prior permission of the data controller and consent from the data subjects). Thus, if there is a transfer of the personal data to a party outside this list, it is deemed as a violation of privacy obligations. Furthermore, DTMT is fed with queries related to data location.

4.2.2 High Level Architecture

Figure 16 shows the high level view of the DTMT internal architecture. The tool continuously logs any activity on the cloud infrastructure that implies data transfers. It, then, performs inference operations on the logs to identify whether transfers were compliant to the defined A-PPL policies and/or constraints manually configured in the tool. Thus, DTMT is able to answer specific queries about the location of the processing and the parties involved in the processing for the given data set, by producing logs related to data transfers.

As shown in Figure 16, DTMT is mainly composed of two main parts. Initially, a proxy module is used to monitor the machine interface calls from different tenants (e.g. data controllers or cloud platform administrators) to the cloud services and extract the audit trail. The latter is exploited to construct a data tracking knowledge base that represents operations on personal data as logical facts, suitable for analysis. The knowledge base involves an inference engine that can assert where a given data set is located in the virtualised environment. It can also answer to audit queries that help an auditor to check whether previous data transfers where compliant.

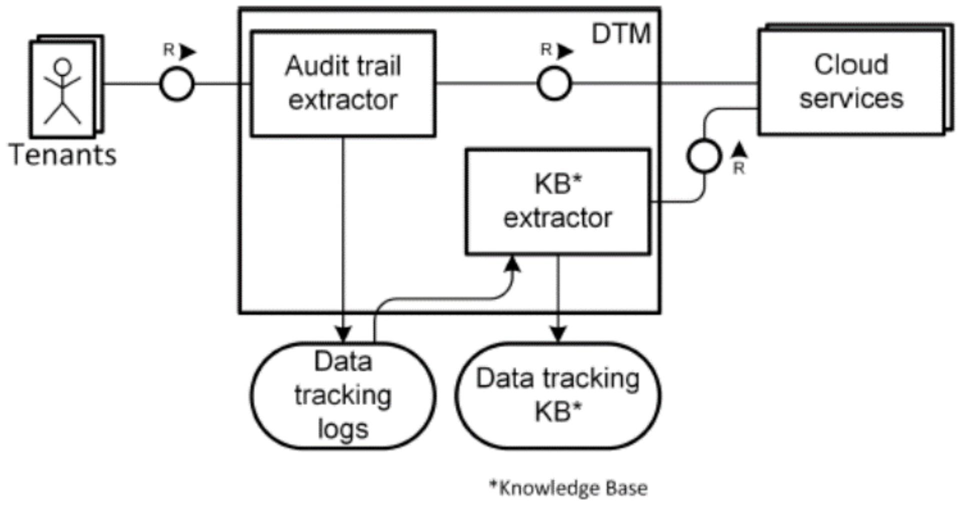


Figure 16: The high level architecture of the Data Transfer Monitoring Tool.

4.2.3 Tool Application Programming Interfaces

Currently, DTMT is designed as a standalone tool, which is accessible through command line to run queries over the collected logs. Although the tool can be extended to interface with a policy enforcement engine, like A-PPLE, to automatically retrieve authorised locations and transfers, it is currently not exposing any API for the other A4Cloud tools. Instead, it needs to consume the functionalities provided by other tools, such as an incident management tool, like IMT, to trigger notifications on potential data transfer violations, and an evidence collection system, like AAS, to store logs generated about data transfers in the upper cloud layer (SaaS or PaaS) for future reference and for auditing purposes. The latter is implemented through TL. Thus, DTMT consumes the APIs exposed by other tools and environments, as shown in Table 12.

Name of the API	Purpose of use	Should be offered by	Data format	API format
Send Incidents	This API is used to communicate the incidents detected from DTMT to the relevant stakeholders of the cloud providers	IMT	JSON	RESTful
Store encrypted logs	The API of TL sender module of Transparency log. This is used to log a message using a pile identifier “aas”. Storing the logs generated by the DTMT and translating them to evidence for future reference (i.e. in case of audits)	TL	JSON	RESTful

Table 12: Interfaces and methods needed by the Data Transfer Monitoring Tool.

4.3 Assertion Tool

4.3.1 Tool Overview and Target Stakeholders

The Assertion Tool (AT) is part of the Assertion Framework, a framework that provides assurance to the cloud service provider of the valid combination of the A4Cloud tools with respect to a set of pre-defined accountability properties. Within the scope of the Assertion Framework, AT ensures the validation of the A4Cloud tools, using a methodology using runtime verification based on validation cases that validate the interactions taking place among two or more tools. This validation might involve data coming from other tools (e.g. logs collected through AAS, IMT notifications). AT is responsible for gathering this data and their application to the validating scenario.

The AT provides two modes of validation: a scenario running mode and a scenario matching mode. In the scenario running mode, AT actively triggers A4Cloud tools in order to execute the validation cases and verifies the accountability properties defined as a validation scenario. While in the matching mode, it passively observe the interactions of the autonomously executing tools, and checks the accountability properties based on the available runtime information, such as logs and notifications.

The validation of the A4Cloud tools is intended to take place before they are delivered to their final users, notably by tool developers. It can also be used by Cloud providers that set up their ecosystems.

4.3.2 High Level Architecture

Figure 17 shows the high level view of the AT architecture. The tool gets as input a combination of the A4Cloud tools, a set of accountability properties to be validated and a set of associated validation scenarios that define how validation will be performed in terms of tool interactions and accountability assertions. The AT offers a graphical user interface, which allows A4Cloud developers and cloud providers to configure accountability tools and the policies, define validation scenarios and their associated accountability properties, and initiate the execution of validation cases.

Overall, the AT automatically executes the validation cases on the selected A4Cloud tools to accomplish the validation tasks. The correct execution of these validation cases may require information coming directly from the tools (e.g., logs, audit reports, etc.) accessible via the tools’ APIs. The execution of the validation cases may also require data that is not usually available through the A4Cloud tools' standard APIs. Thus, a set of extensible interfaces, called validation APIs, has been implemented for validation purposes. These APIs represent dedicated validation interfaces to pass information in a controlled, notably secure and sanitized manner, directly from the A4Cloud tools to the assertion tool.

The outcome of this tool is a report stating whether or not the implementation of a tool combination satisfies certain accountability properties. A validation report indicates which accountability properties have been satisfied and which are not, together with some indication of the underlying reasons.

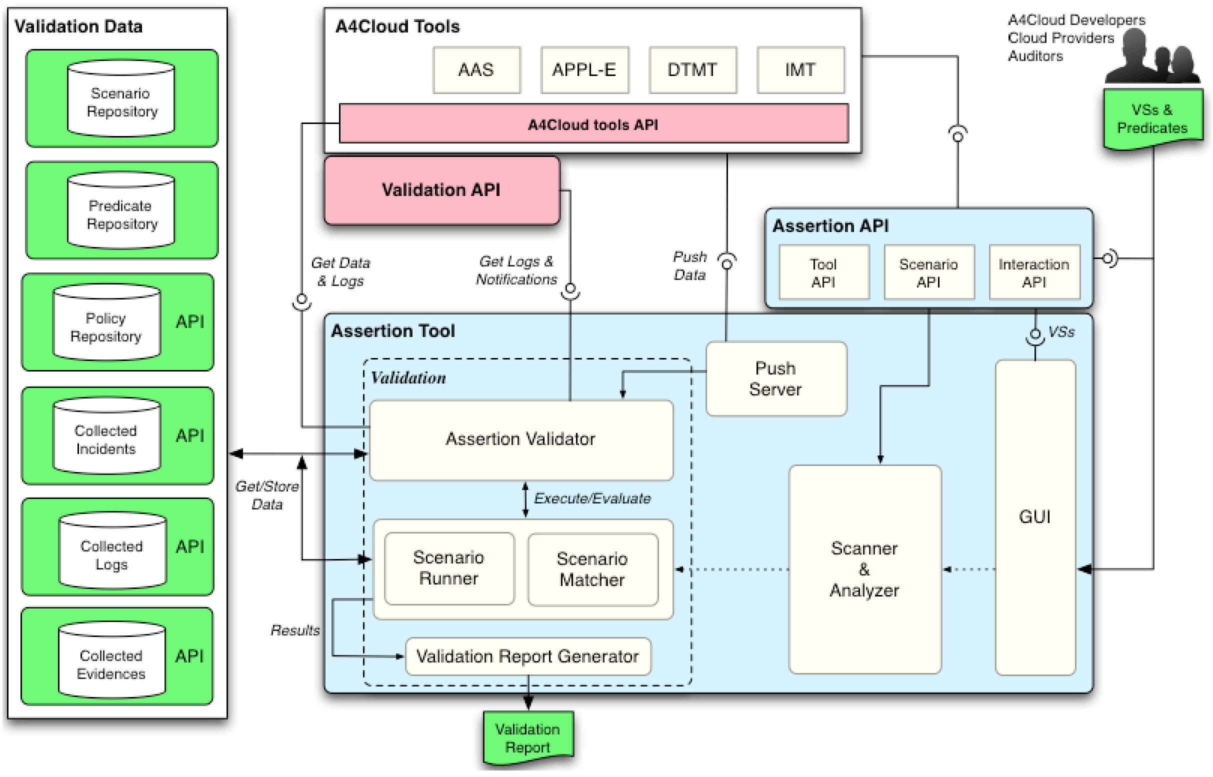


Figure 17: The high level architecture of the Assertion Tool.

4.3.3 Tool Application Programming Interfaces

AT supports a Graphical User Interface, which enables interaction with the target users, in the sense that the developers can work with a Java-based Domain Specific Language (DSLs), which enables the definition of validation scenarios in a concise and declarative way.

The AT offers a set of APIs which allow other tools and A4Cloud developers to communicate with AT programmatically or through the GUI. Table 13 shows the APIs provided by AT. The main service offered by Assertion API (through Interaction API, see Figure 17) is the interaction with other accountability tools service. This subinterface mainly enables the call of functionalities provided through other tools, often through calls to REST services. The Assertion API also involves the following two services through dedicated sub-interfaces:

- Tool API, which provides tool declaration and configuration service: This sub-interface allows the definition of new accountability tools, the listing of current tools, etc. Such a declaration mainly requires the definition of access methods to the functionality provided by the tools, principally in form of calls to REST services, notably for access to logs and notifications provided by the tool.

- Scenario API, which provides scenario definition and execution service: This subinterface permits validation scenarios to be loaded, configured and executed. It also provides services for listing the existing scenarios and their modification.

Name of the API	Purpose of use	Consumed by	Data format	API format
Assertion API	Test an Assertion for a given toolkit. It instantiates an evaluation for a given toolkit over an accountability attribute.	The UI of the AT (for A4Cloud developers)	JSON	RESTful
Scenario API	- Load or reload a validation scenario based on abstract description - List all loaded validation Scenarios	The UI of the AT (for A4Cloud developers)	JSON	RESTful
Tool API	- Load or reload a Tool abstract definition. - List all loaded tools	The UI of the AT (for A4Cloud developers)	JSON	RESTful

Table 13: Interfaces offered by the Assertion Tool.

Furthermore, AT consumes the APIs provided by other tools and environments, as shown in Table 14. The AT mainly consumes:

- Validation API: Validation APIs are offered by some A4Cloud tools in order to provide data that is not publicly offered by the tools, but is useful for validation. The validation API may support pushing such information from the external tool to the AT. Furthermore, it can be used to send data in a more secure way, after sanitizing and anonymization. The AAS and IMT tools have been extended by validation interfaces that have been used in concrete validation tasks as part of the D-6 demo.

- A4Cloud tools APIs: The AT employs the APIs of the external tools in order to interact with them through REST calls. These interactions include, for example, the installation of A-PPLE policies in order to initialize the execution of a validation scenario or the request of access to a log that is stored within a Transparency Log component.

Name of the API	Purpose of use	Should be offered by	Data format	API format
Validation APIs	Operations to get or generate the logs, evidences, incidents and policies associated to a given tool	A4Cloud tools	JSON	RESTful
A4Cloud tools APIs	Allows access to data (test cases, policies, incidents, logs and evidences) needed for an A4Cloud tools validation	Ideally by the repositories provided by the A4Cloud tools	JSON	RESTful

Table 14: Interfaces needed by the Assertion Tool.

4.4 Accountability Monitor Tool

4.4.1 Tool Overview and Target Stakeholders

The goal of the Accountability Monitor Tool (AccMon) is to provide means to monitor accountability policies in the context of a real system. AccMon allows to specify policies that are applicable to network traffic, web application code and external components via plugins. These policies are based on a distributed temporal logic with data. The framework allows centralized or distributed monitoring and both on-line and off-line controls.

AccMon is used by IT administrators of cloud providers.

4.4.2 High Level Architecture

As shown in Figure 18, AccMon acts as a middleware framework, which intercepts and logs client's HTTP requests, and server's processing requests and responses. On the web application side, the developer can configure the framework to intercept function/method calls and access to databases. AccMon can act as a daemon and can be interconnected with external tools. The property to be monitored by AccMon is defined inside these tools, which, subsequently, send log events to AccMon via HTTP calls. The framework can also be connected with external hardware, such as Arduino electronic boards[2].

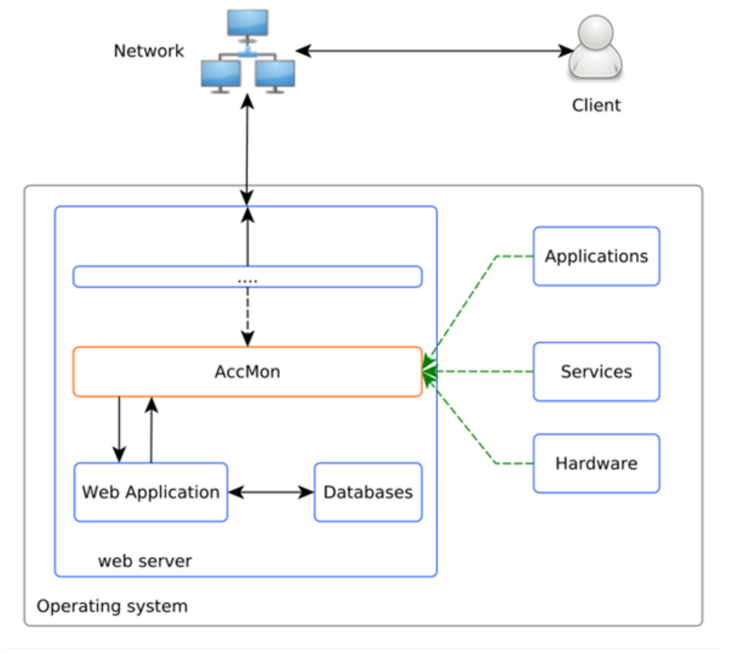


Figure 18: The high level architecture of the Accountability Monitoring Tool.

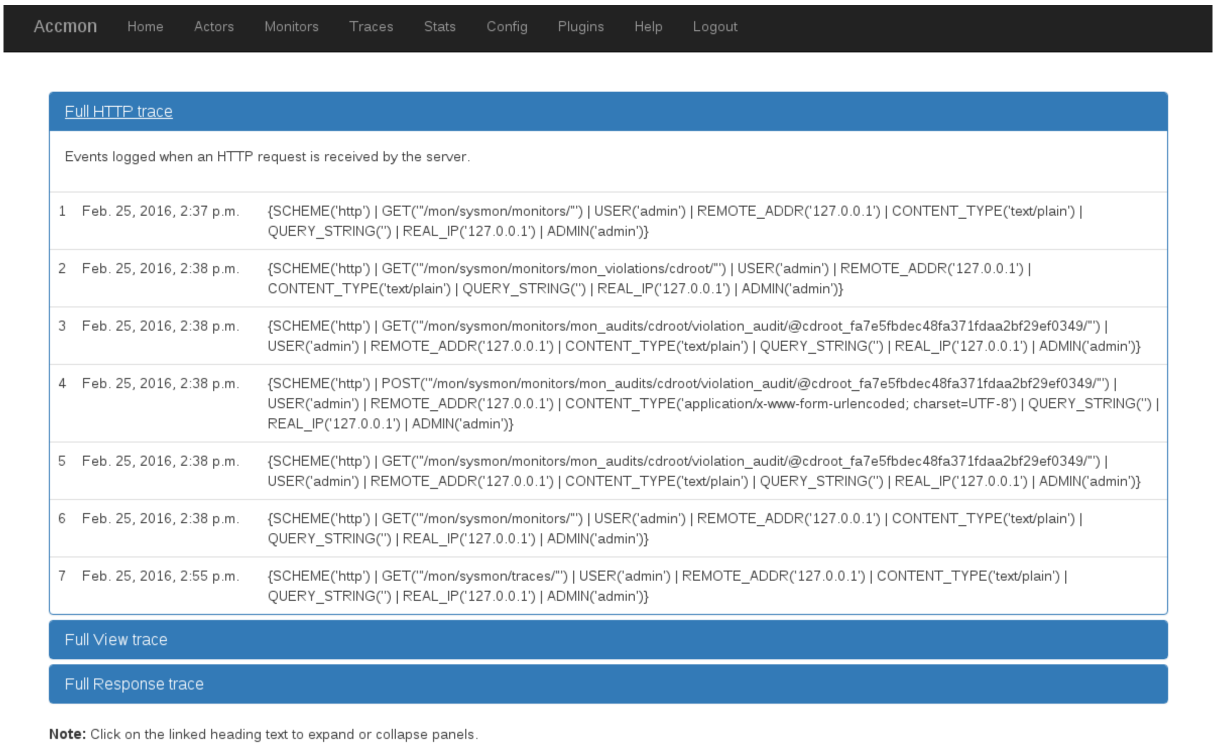
4.4.3 User interface

AccMon provides a User Interface to allow the IT administrators of cloud providers to monitor the list of accountability properties and browse the events logged into the tool. More specifically, Figure 19 shows the events collected by AccMon on the HTTP requests, while Figure 20 shows the list of monitors that are running in AccMon.

4.4.4 Tool Application Programming Interfaces

AccMon does not offer any interfaces, but it consumes the interfaces exposed by the monitoring systems to collect logs on events related to the accountability properties that need to be monitored.

[2] https://learn.sparkfun.com/tutorials/what-is-an-arduino.



Full HTTP trace

Events logged when an HTTP request is received by the server.

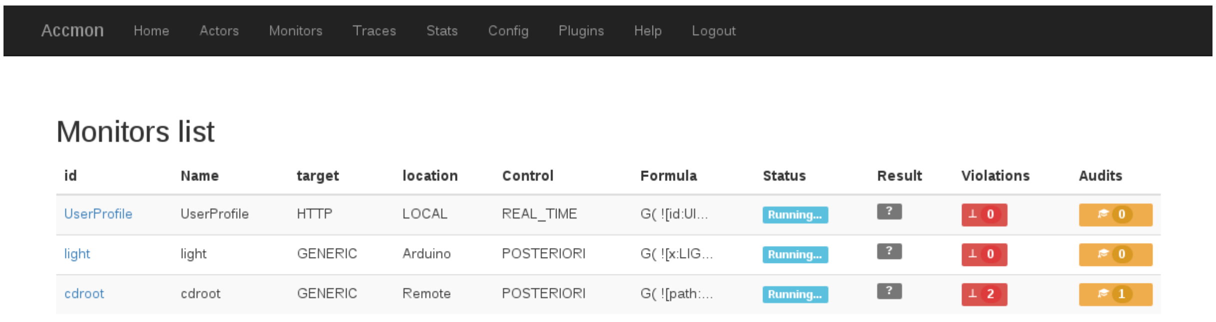
1	Feb. 25, 2016, 2:37 p.m.	{SCHEME('http') | GET(""/mon/sysmon/monitors/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('text/plain') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}
2	Feb. 25, 2016, 2:38 p.m.	{SCHEME('http') | GET(""/mon/sysmon/monitors/mon_violations/cdroot/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('text/plain') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}
3	Feb. 25, 2016, 2:38 p.m.	{SCHEME('http') | GET(""/mon/sysmon/monitors/mon_audits/cdroot/violation_audit/@cdroot_fa7e5fbdec48fa371fdaa2bf29ef0349/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('text/plain') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}
4	Feb. 25, 2016, 2:38 p.m.	{SCHEME('http') | POST(""/mon/sysmon/monitors/mon_audits/cdroot/violation_audit/@cdroot_fa7e5fbdec48fa371fdaa2bf29ef0349/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('application/x-www-form-urlencoded; charset=UTF-8') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}
5	Feb. 25, 2016, 2:38 p.m.	{SCHEME('http') | GET(""/mon/sysmon/monitors/mon_audits/cdroot/violation_audit/@cdroot_fa7e5fbdec48fa371fdaa2bf29ef0349/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('text/plain') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}
6	Feb. 25, 2016, 2:38 p.m.	{SCHEME('http') | GET(""/mon/sysmon/monitors/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('text/plain') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}
7	Feb. 25, 2016, 2:55 p.m.	{SCHEME('http') | GET(""/mon/sysmon/traces/"") | USER('admin') | REMOTE_ADDR('127.0.0.1') | CONTENT_TYPE('text/plain') | QUERY_STRING('') | REAL_IP('127.0.0.1') | ADMIN('admin')}

Full View trace

Full Response trace

Note: Click on the linked heading text to expand or collapse panels.

Figure 19: The traces from HTTP requests logged into AccMon.



Monitors list

id	Name	target	location	Control	Formula	Status	Result	Violations	Audits
UserProfile	UserProfile	HTTP	LOCAL	REAL_TIME	G(![id:UI...	Running...	?	⊥ 0	0
light	light	GENERIC	Arduino	POSTERIORI	G(![x:LIG...	Running...	?	⊥ 0	0
cdroot	cdroot	GENERIC	Remote	POSTERIORI	G(![path:...	Running...	?	⊥ 2	1

Figure 20: The UI of AccMon to list the available monitors in the tool.

4.5 Security and Privacy Assurance Case Environment (SPACE)

4.5.1 Tool Overview and Target Stakeholders

The **Security and Privacy Assurance Case Environment (SPACE)** is concerned with the problem of providing assurance of how cloud providers comply with relevant policies and support them in operational environments, hence increasing trustworthiness and enhancing transparency of cloud supply chains. Cloud service provides have to comply with internal as well as external **security and privacy policies**. Such policies are supported, that is implemented, by different **security and privacy controls,** which are deployed across cloud supply chains. SPACE addresses the limited support for **operational alignment** between policies and associated controls. Current certification and auditing practices provide limited **assurance to customers** and third parties. SPACE helps to:

- **Monitor** security and privacy, leading to transparency and assurance, and enhancing accountability for cloud services;
- **Understand** and contextualize accountability in cloud services;
- **Capture** how services can be accountable in different ways operationally;

- **Build** accountability cases in a systematic way;
- **Provide** dynamic assurance right across cloud service provision chains, for both security and privacy;
- **Communicate** to business stakeholders' accountable experience and behaviour of cloud services.

Figure 21 summarises graphically the rationale underlying the Security and Privacy Assurance Case Environment (SPACE).

SPACE addresses the needs of different cloud actors:

- For Cloud Service Providers:
  - o An assurance case environment for mapping security and privacy policies to associated controls;
  - o A software-defined storage solution for gathering evidence supporting assurance;
  - o Structuring security and privacy policies in terms of arguments and supporting evidence gathered or generated by specific controls and associated technical solutions deployed in the cloud.
- For Cloud Customers:
  - o Evidence-based continuous assurance of the adopted cloud services and related supply chains.
- For Cloud Auditors:
  - o A centralised cloud-native evidence repository for inspecting cloud supply chains.

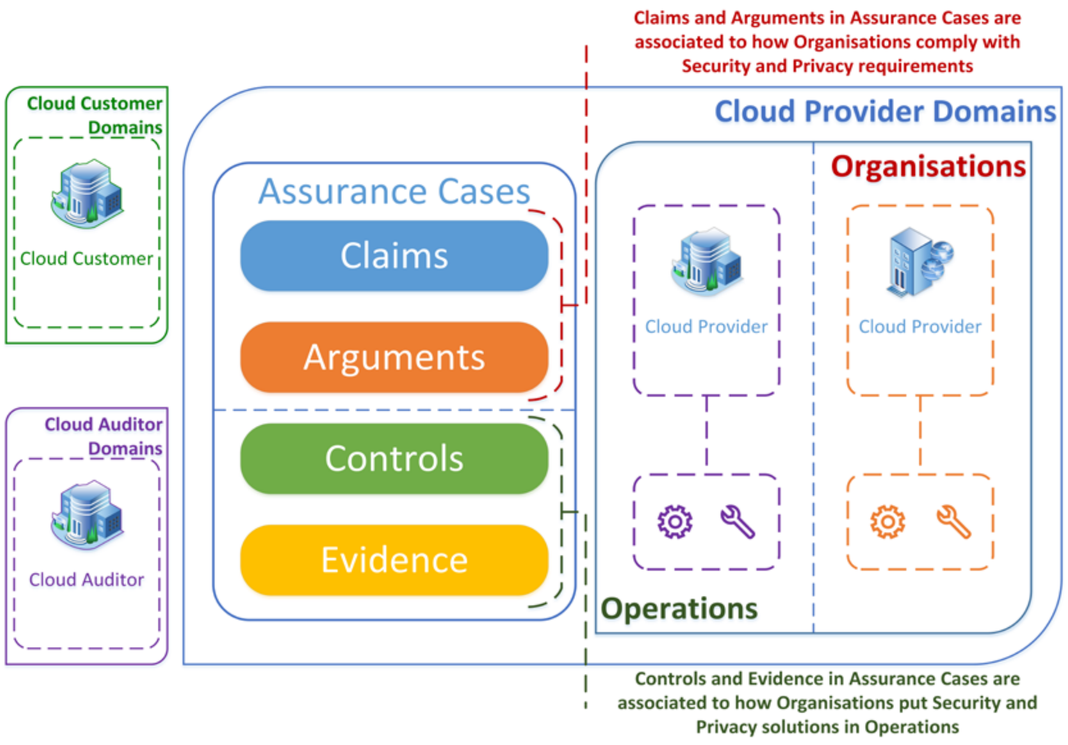

Claims and Arguments in Assurance Cases are associated to how Organisations comply with Security and Privacy requirements
Cloud Customer Domains
Cloud Customer
Cloud Auditor Domains
Cloud Auditor
Cloud Provider Domains
Assurance Cases
Claims
Arguments
Controls
Evidence
Organisations
Cloud Provider
Cloud Provider
Operations
Controls and Evidence in Assurance Cases are associated to how Organisations put Security and Privacy solutions in Operations

Figure 21: Supporting Assurance in Cloud Supply Chains.

SPACE provides a systemic support towards continuous assurance. It enables evidence driven assurance linking evidence to policies. It combines assurance of policy compliance with operations of cloud services.

4.5.2 High Level Architecture

Figure 22 shows the main components forming the SPACE high level architecture.

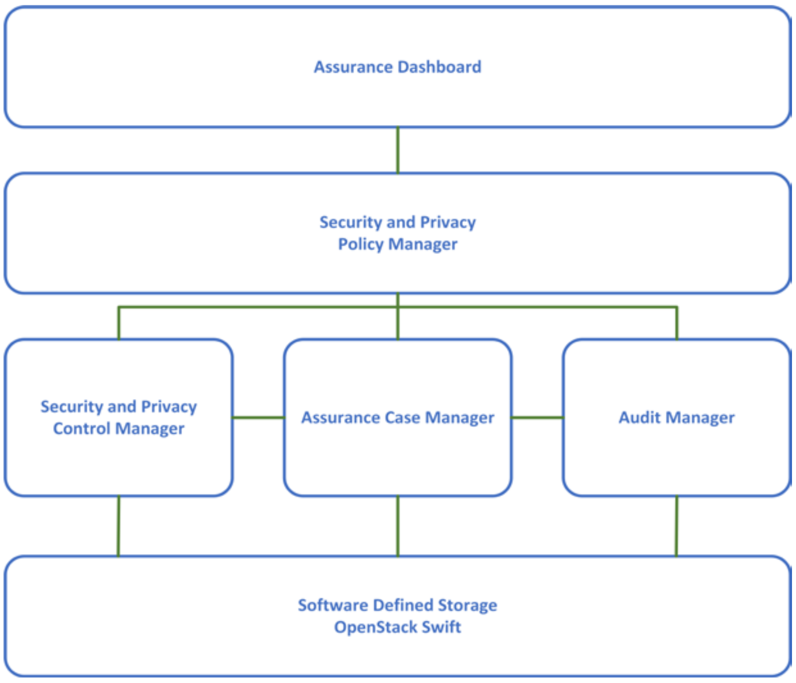

Assurance Dashboard
Security and Privacy Policy Manager
Security and Privacy Control Manager
Assurance Case Manager
Audit Manager
Software Defined Storage OpenStack Swift

Figure 22: SPACE High Level Architecture

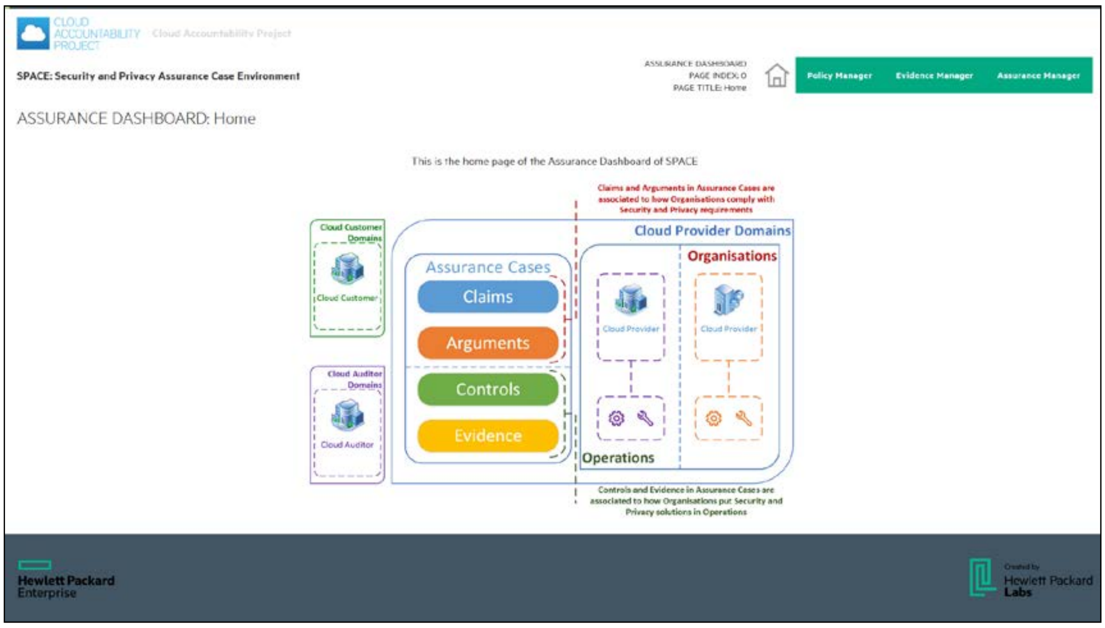
Each component is associated with specific functional elements of SPACE:

- **Software Defined Storage:** storing meta-data (evidence) of deployed security and privacy controls in OpenStack Swift containers.
- **Security and Privacy Control Manager:** managing operational information of security and privacy controls like the ones listed by the CSA Cloud Control Matrix and providing an interface to the Software Defined Storage in order to manage and configure the storage containers.
- **Assurance Case Manager:** Maintaining dynamic assurance cases that reflect operational effectiveness and appropriateness of security and privacy controls.
- **Audit Manager:** Monitoring and assessing (based on stored evidence) deployed controls and their operational dependencies in order to support dynamic assurance cases.
- **Security and Privacy Policy Manager:** Mapping security and privacy objectives (policies) to specific controls.
- **Assurance Dashboard:** supporting transparency and evidence driven assurance, it provides functionalities for monitoring and communicating policy compliance in operational environments.

4.5.3 User interface

This section shows the main views implemented by the SPACE dashboard. The SPACE dashboard has been implemented using the Grommet[3] UX framework.
Figure 23 shows the main view of the dashboard, which provides access to the different views by a menu bar.

[3] Grommet: http://www.grommet.io/


CLOUD ACCOUNTABILITY PROJECT Cloud Accountability Project

SPACE: Security and Privacy Assurance Case Environment

ASSURANCE DASHBOARD
PAGE INDEX: 0
PAGE TITLE: Home

ASSURANCE DASHBOARD: Home

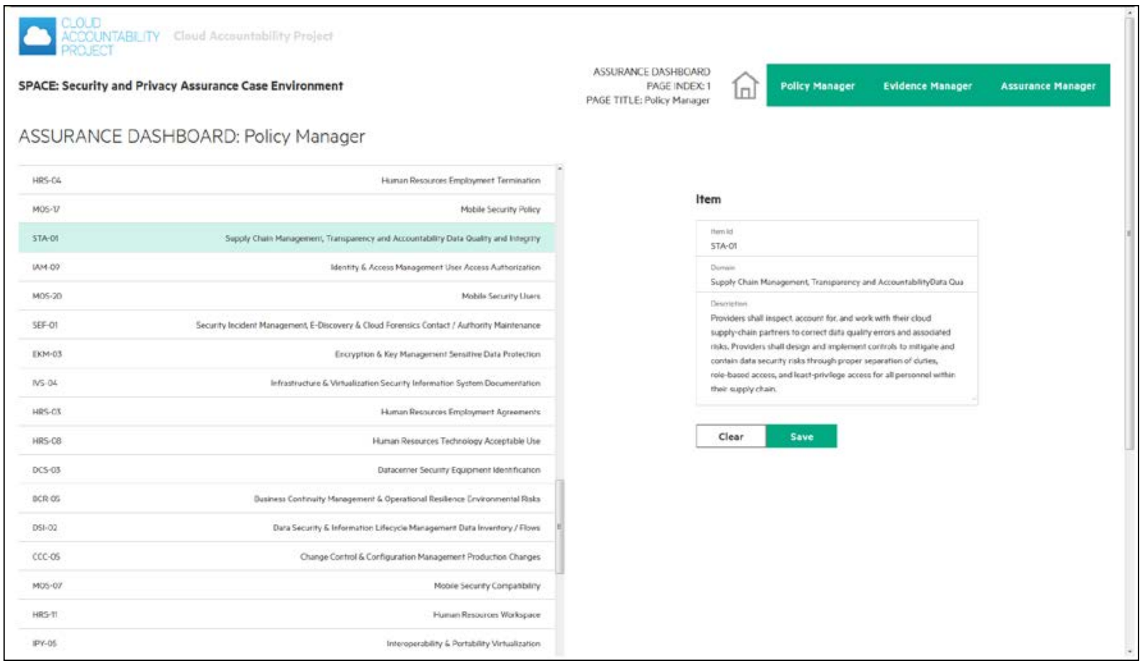
This is the home page of the Assurance Dashboard of SPACE

Claims and Arguments in Assurance Cases are associated to how Organisations comply with Security and Privacy requirements

Controls and Evidence in Assurance Cases are associated to how Organisations put Security and Privacy solutions in Operations

Figure 23: The SPACE dashboard.

Figure 24 shows the Policy Manager view, which provides access to the (predefined) lists of claims, arguments and controls. Such lists need to be managed at the organizational level. Figure 24 shows for example the list of controls defined according to the CSA Cloud Control Matrix.


CLOUD ACCOUNTABILITY PROJECT Cloud Accountability Project

SPACE: Security and Privacy Assurance Case Environment

ASSURANCE DASHBOARD
PAGE INDEX: 1
PAGE TITLE: Policy Manager

ASSURANCE DASHBOARD: Policy Manager

HRS-04	Human Resources Employment Termination
MOS-17	Mobile Security Policy
STA-01	Supply Chain Management, Transparency and Accountability Data Quality and Integrity
IAM-09	Identity & Access Management User Access Authorization
MOS-20	Mobile Security Users
SEF-01	Security Incident Management, E-Discovery & Cloud Forensics Contact / Authority Maintenance
EKM-03	Encryption & Key Management Sensitive Data Protection
IVS-04	Infrastructure & Virtualization Security Information System Documentation
HRS-03	Human Resources Employment Agreements
HRS-08	Human Resources Technology Acceptable Use
DCS-03	Datacenter Security Equipment Identification
BCR-05	Business Continuity Management & Operational Resilience Environmental Risks
DSI-02	Data Security & Information Lifecycle Management Data Inventory / Flows
CCC-05	Change Control & Configuration Management Production Changes
MOS-07	Mobile Security Compatibility
HRS-11	Human Resources Workspace
IPY-05	Interoperability & Portability Virtualization

Item

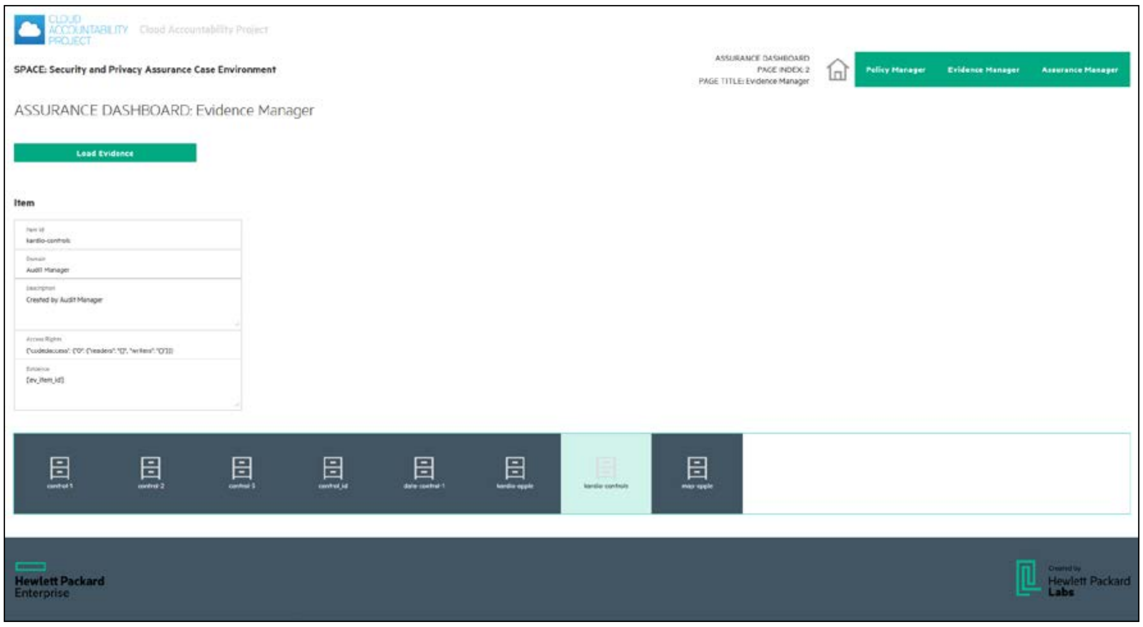
Item Id
STA-01

Domain
Supply Chain Management, Transparency and AccountabilityData Qua

Description
Providers shall inspect, account for, and work with their cloud supply-chain partners to correct data quality errors and associated risks. Providers shall design and implement controls to mitigate and contain data security risks through proper separation of duties, role-based access, and least-privilege access for all personnel within their supply chain.

Figure 24: The Policy Manager view of the SPACE tool.

Figure 25 shows the Evidence Manager view, which shows the storage containers configured in the software defined storage (OpenStack Swift). Each container is associated to a specific control (mechanism) deployed in the cloud supply chain. The content of each container consists of evidence files (storage objects) of meta-data (e.g. alert notifications, policy violations, etc.) associated (that means, generated or processed) with the specific control. The access rights of containers are specified according to the different responsibilities (for the deployed control or mechanism) in the cloud supply chain.


CLOUD ACCOUNTABILITY PROJECT
Cloud Accountability Project
SPACE: Security and Privacy Assurance Case Environment
ASSURANCE DASHBOARD
PAGE INDEX: 2
PAGE TITLE: Evidence Manager
Policy Manager
Evidence Manager
Assurance Manager
ASSURANCE DASHBOARD: Evidence Manager
Load Evidence
control 1
control 2
control 3
control_id
data control 1
kardio-apple
kardio-controls
map-apple
Hewlett Packard Enterprise
Created by Hewlett Packard Labs

Item

Item id
kardio-controls

Domain
Audit Manager

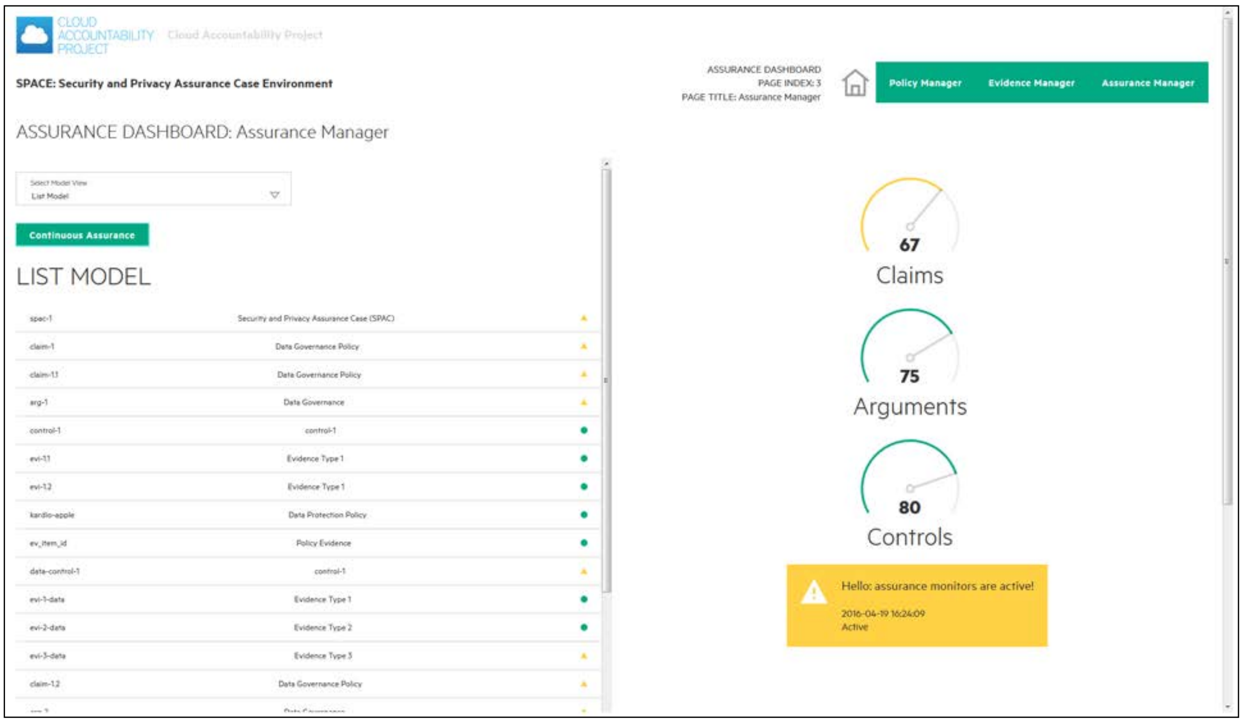
Description
Created by Audit Manager

Access Rights
{"codedaccess": {"0": {"readers": "{}", "writers": "{}"}}}

Evidence
[ev_item_id]

Figure 25: The Evidence Manager view of the SPACE tool.

Figure 26 shows the Assurance Manager view, which shows the assurance model specified for the specific cloud supply chain. Two simple models are supported: a simple overview list model of all claims, arguments, controls and associated evidence that are monitored and assessed in the cloud supply chain, and a map model providing a structured representation how high level claims are mapped to arguments, controls and evidence. The Assurance Manager view also supports the live monitoring of compliance (metrics) based on the successful auditing performed over the evidence stored in the underlying software defined storage of containers. Furthermore, it shows live messages received by audit monitors associated with each monitored control or mechanism.


CLOUD ACCOUNTABILITY PROJECT
Cloud Accountability Project
SPACE: Security and Privacy Assurance Case Environment
ASSURANCE DASHBOARD
PAGE INDEX: 3
PAGE TITLE: Assurance Manager
Policy Manager
Evidence Manager
Assurance Manager
ASSURANCE DASHBOARD: Assurance Manager
Select Model View
List Model
Continuous Assurance
67
Claims
75
Arguments
80
Controls

LIST MODEL

spac-1	Security and Privacy Assurance Case (SPAC)	▲
claim-1	Data Governance Policy	▲
claim-11	Data Governance Policy	▲
arg-1	Data Governance	▲
control-1	control-1	●
evi-11	Evidence Type 1	●
evi-12	Evidence Type 1	●
kardio-apple	Data Protection Policy	●
ev_item_id	Policy Evidence	●
data-control-1	control-1	▲
evi-1-data	Evidence Type 1	●
evi-2-data	Evidence Type 2	●
evi-3-data	Evidence Type 3	▲
claim-12	Data Governance Policy	▲

Hello: assurance monitors are active!
2016-04-19 16:24:09
Active

Figure 26: The Assurance Manager view of the SPACE tool.

4.5.4 Tool Application Programming Interfaces

The SPACE implementation is serviced oriented, therefore there are different servers supporting different functionalities. This section gives just an overview of the interfaces between different components/systems forming space. Three different layers form SPACE, namely: Software Defined Storage, SPACE Backend Logic and SPACE dashboard.

- **Software Defined Storage:** relies on OpenStack Swift. A dedicated server (service) runs OpenStack Swift, which provides a swift client as well as RESTful API for all interactions with the software defined storage. The Security and Privacy Control Manager component implements the necessary functionalities for the interaction between the SWIFT Backend Logic layer and the Software Defined Storage layer.
- **SPACE Backend Logic:** implements all the logical components providing the main functionalities (e.g. managing list of claims, arguments and controls, structuring security and privacy assurance cases, auditing cloud supply chains based on the structured assurance cases and the operational evidence stored) that underpin SPACE. The SPACE Backend Logic is accessible via a RESTful API and a dedicated backend server.
- **SPACE Dashboard:** implements the GUI (Graphical User Interface) supporting the communication of the operational compliance, hence assurance, of the cloud supply chain. The SPACE Dashboard accesses the RESTful API of the SPACE Backend Logic layer in order to retrieve the information (coded in JSON, JavaScript Object Notation) shown in the different views. Furthermore, the SPACE Dashboard uses WebSockets in order to open an interactive communication with the SAPCE Backend Logic server.

5 Data Subject Controls

The Data Subject Controls functional area offers accountability support in a detective manner. The set of tools belonging to this category satisfy the need of the data subjects (i.e. individuals whose personal data are collected and/or processed by cloud service providers) to verify the correct processing of their data. The tools implement runtime mechanisms to follow the proper execution of data handling accountability policies and provide data subjects with notifications on how their data are exploited along cloud service supply chains.

In the context of the A4Cloud architecture, these mechanisms are being developed within the scope of the A4Cloud software tools, namely the Data Track (DT) and the Transparency Log (TL). We also include in this section a separate description of the Plug-in for Assessment of Policy Violation (PAPV), but this plug-in is considered to be an integral part of the Data Track tool.

5.1 Data Track

5.1.1 Tool Overview and Target Stakeholders

Data Track (DT) is a tool used by data subjects to get an overview of all personal data they have disclosed to online services. The tool allows them search through their history of data disclosures to see what personal data they have disclosed, to whom they have disclosed these data to (i.e. which service), and under which privacy policy. Furthermore, the DT tool enables users to directly assert their right to access and rectify the data concerning them held by the service provider.

Specifically, DT enables data subjects to:

- request access to their data stored by a service provider;
- request to correct their data;
- request that their data should be deleted (or at least blocked);
- view detailed information about how the data has been shared and used by the service provider;
- view prior data disclosures with associated metadata like privacy policy, time of disclosure, etc.;
- get notified of policy violations (via other relevant tools);
- seek redress (via handing-over to relevant tool).

DT is used by data subjects, through a front-end visualisation module. It gets as input the user's data disclosures, the data handling policy, describing relevant operations and obligations for the particular data disclosure, the credentials to remotely access user data at the service providers' side, and the notifications received from the providers. DT indexes data disclosures and stores them in an internal format. Upon a request from the user, DT remotely accesses the user's data located at different providers. The tool supports sending requests to correct or delete personal data stored at service providers. Last, but not least, DT analyses log data about how personal data have been processed and receive notifications of policy violations from the service provider.

The output of the tool is a list of data disclosures for both explicitly disclosed personal data and implicitly collected data by service providers, potentially stored remotely, presented via different visualisations. Users are also presented with notifications from service providers through RRT, for example to inform them about policy violations or privacy policy updates from the service provider.

5.1.2 High Level Architecture

Figure 27 illustrates an example of the DT tool and the interactions with other A4Cloud tools in a cloud provider setting. The trace view of DT is shown to the left. DT receives notifications from A-PPLE (and potentially also logging data) from the Transparency Log (TL) components, consisting of the TL Sender running at a cloud provider, a TL Server, and the TL Recipient component that receives messages from the TL Sender. Note that the TL Recipient is embedded as part of DT running locally on the data subject's device, which is assumed to be under the control of the data subject. A-PPLE receives notifications from the Incident Management Tool (IMT).

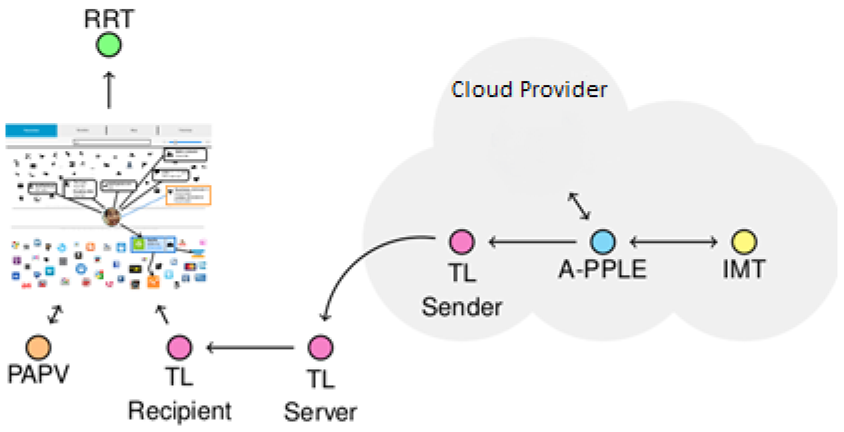


Figure 27: An example of the Data Track and the other A4Cloud tools interactions.

In case of notifications concerning incidents or violations, DT uses the plug-in for assessing policy violations (PAPV) to assess the severity of these notifications. The severity will impact how the notifications are presented to the data subject by DT. For example, a high severity may cause a notification to the data subject as soon as the tool is launched, while less severe notifications are only shown on request. If the data subject wishes to act on a notification concerning an incident or a violation, the DT Core can launch the Remediation and Redress Tool (RRT).

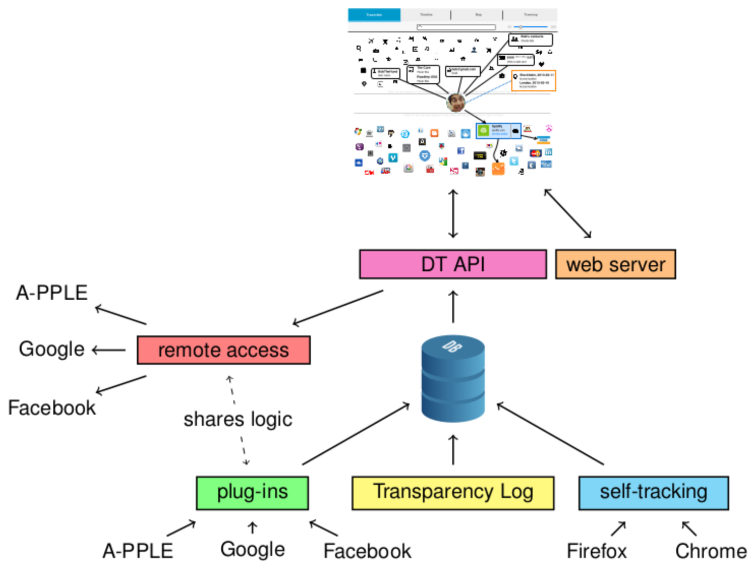


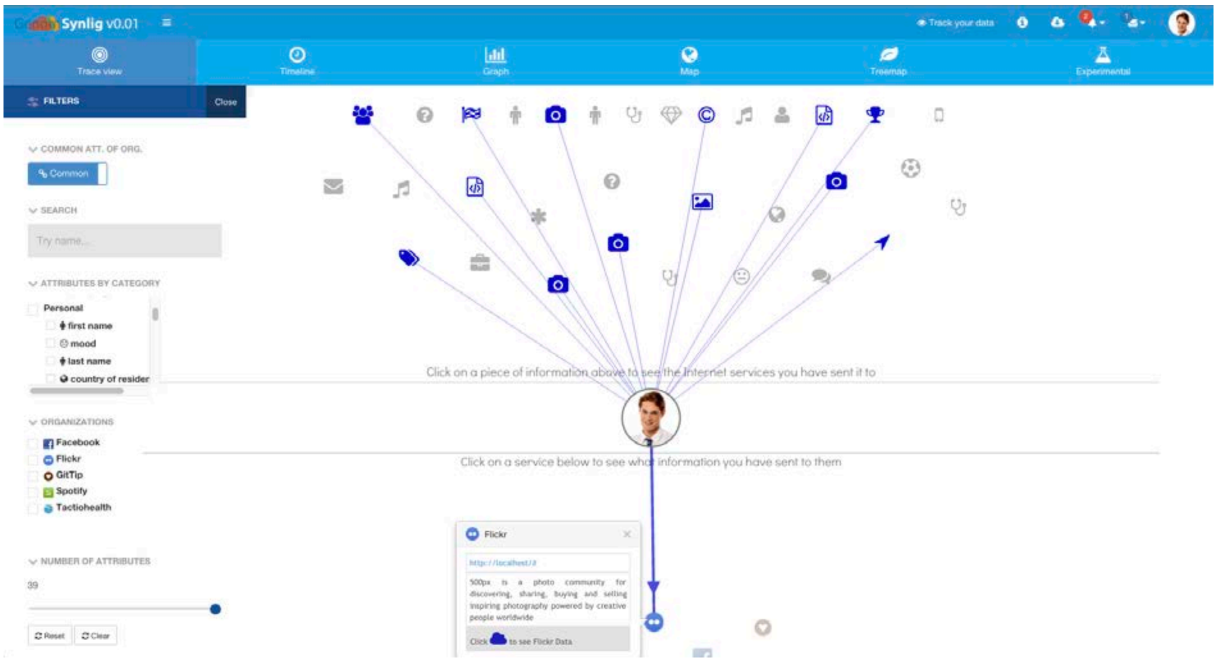
Figure 28: The high level view of the Data Track architecture.

Figure 28 shows an overview of the internal architecture of the Data Track. On the top we have the different front-end user interfaces running in a browser (based on HTML5, CSS, and JavaScript). The front-end is provided by an embedded web server running locally on a device under the data subject's control. The UI communicates with the back-end through the DT API that provides a uniform view on all data disclosures stored in the internal DT database. The API also enables the front-end to make remote access requests to different service providers, such as Google, Facebook, or any service that supports the API provided by A-PPLE. The DT database is populated with data disclosures through three different potential sources:

- **Remote access plug-ins**: DT has a plug-in architecture for making requests to different services in order to retrieve all personal data stored in the service. The plug-in first guides the user through requesting the data, and then once obtained the plug-in translates the data from whatever custom format it now uses into the internal data disclosure format used in the DT database.

- **Transparency Log (TL) messages**: DT receives messages through TL sent from service providers. Based on these messages, DT can derive data disclosures (such as those made down the cloud supply chain) and store them in the internal DT database.
- **Self-tracking**: DT can also, conceptually, support data subject self-tracking plug-ins (the focus of an earlier version of DT) that enables data subjects to track their own data disclosures with the help of, e.g., browser plug-ins that detect explicit data disclosures through web forms.

5.1.3 User Interface



Click on a piece of information above to see the Internet services you have sent it to

Click on a service below to see what information you have sent to them

http://localhost/://

500px is a photo community for discovering, sharing, buying and selling inspiring photography powered by creative people worldwide

Figure 29: The trace view with filters visible on the left-hand side.

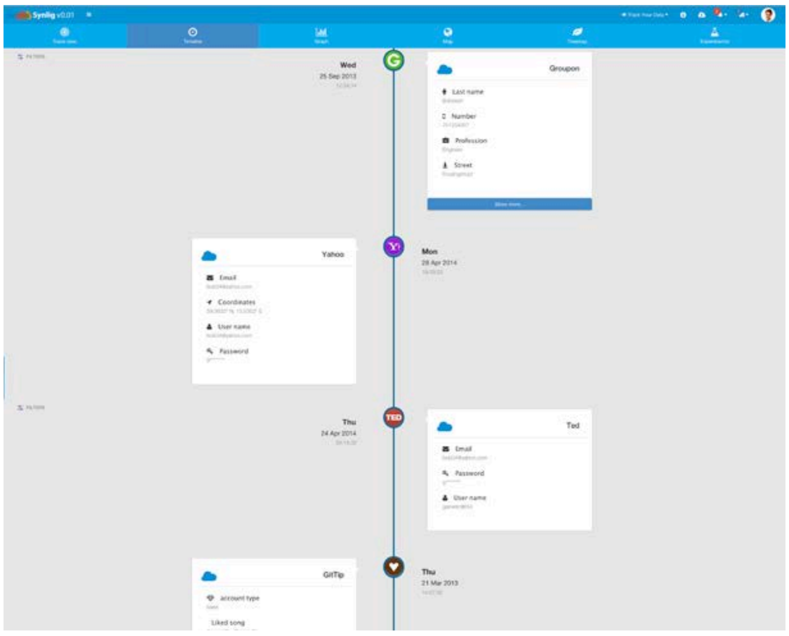


Figure 30: The DT timeline view.

DT provides different visualisations of the user's data disclosures. One of the visualisations shows traces indicating what attributes a user has disclosed to which providers, by visually linking cloud service providers and attributes (Figure 29). Another visualisation shows a timeline of the user's data disclosures (Figure 30). Each view contains a number of different filtering options for enabling the data subject to find relevant information.

5.1.4 Tool Application Programming Interfaces

Furthermore, DT provides a set of API methods that are offered by the *Idatatrack* Interface and can be consumed both by the UI of the DT and other A4Cloud tools, like RRT. These interface methods provided by DT are shown in Table 15. In this table and due to the large size of the API of DT, which consists of more than 50 API method calls, we only present the different API names and their scope.

Name of the API	Purpose of use	Consumed by	Data format	API format
/v1	Print the API reference	The UI of the DT (for data subjects)	JSON	RESTful
/v1/user	Create or update user entries and retrieve user name and picture	The UI of the DT (for data subjects)	JSON	RESTful
/v1/disclosure/	Manage local explicit and implicit disclosure data and their attributes	The UI of the DT (for data subjects)	JSON	RESTful
/v1/attribute/	Manage local explicit and implicit attribute data	The UI of the DT (for data subjects)	JSON	RESTful
/v1/organization/	Manage local organisation data	The UI of the DT (for data subjects)	JSON	RESTful
/v1/type/	Retrieve all types belonging to a (sub)category	The UI of the DT (for data subjects)	JSON	RESTful
/v1/disclosure/ :disclosureid/ remotedata	Manage remote data attributes	The UI of the DT (for data subjects), RRT	JSON	RESTful
/v1/import	Import remote data	The UI of the DT (for data subjects)	JSON	RESTful
/v1/wearable/	Fetch and store upstream data for the wearable application	The UI of the DT (for data subjects)	JSON	RESTful
/v1/insynd/	Manage transparency log keys and messages	The UI of the DT (for data subjects)	JSON	RESTful
/v1/incident/	Manage incident data	The UI of the DT (for data subjects), RRT	JSON	RESTful
/v1/bouncemail	Use to mail a user agent	The UI of the DT (for data subjects), RRT	JSON	RESTful
/v1/testdata	Create static test data for the timeline view of DT	The UI of the DT (for data subjects)	JSON	RESTful

Table 15: Interfaces and respective methods provided by the Data Track tool.

Furthermore, DT consumes the APIs provided by other tools and environments, as shown in Table 16.

Name of the API	Purpose of use	Should be offered by	Data format	API format
TL Recipient	Retrieve all data sent to the data subject through the Transparency Log. Also provides access to other Transparency Log functionality such as cryptographic proofs of authenticity, time, inconsistency, etc.	Transparency Log	Plain text for data (that is, retrieved data is in the same format as written), JSON or Google Protocol Buffers for cryptographic proofs	RESTful
Policy Violation Plugin	Enable the data subject to assess the severity of one or more incidents	Plug-in for Assessment of Policy Violation	-	Go
TL Sender	Enable the DT API to interact with the Transparency Log to authenticate data subjects	Transparency Log	Google Protocol Buffers (internal API between DT and TL), that is base64 encoded in JSON (to fit the generic external DT API)	RESTful
A-PPLE API	Get PII and their types, along with the supported policies	A-PPLE	JSON / XML (A-PPL)	RESTful

Table 16: Interfaces and methods needed by the Data Track tool.

5.2 Plug-in for Assessment of Policy Violation

5.2.1 Tool Overview and Target Stakeholders

The Plug-in for Assessment of Policy Violation (PAPV) provides an assessment on the severity of previously detected policy violations. By using it, data subjects (or their representatives) can check which policy violations are the most relevant ones to be presented to the data subject itself.

PAPV is used by data subjects to assess policy violations. The plug-in gets as input a collection of instances of the policy violations from the DT tool, where an instance of a policy violation is any piece of evidence that describes an occurrence of a policy violation event, and a machine-readable policy description, detailing the obligations of the data controller, regarding the handling procedures for the personal data of the data subjects, which will serve as a reference for evaluating the detected violations.

By receiving a list of the latest policy violations, together with their associated policies, PAPV produces an assessment for the relevance and severity of each reported violation. It then prepares an ordered list of policy violations and sorts them by the level of importance. The output of the plug-in is an ordered measurement (either qualitative or quantitative) of the relevance of the violation event, for each instance of the policy violation. This enables the list of policy violations to be sorted with respect to their relevance and criticality.

5.2.2 High Level Architecture

Figure 31 shows the high level architecture of PAPV.

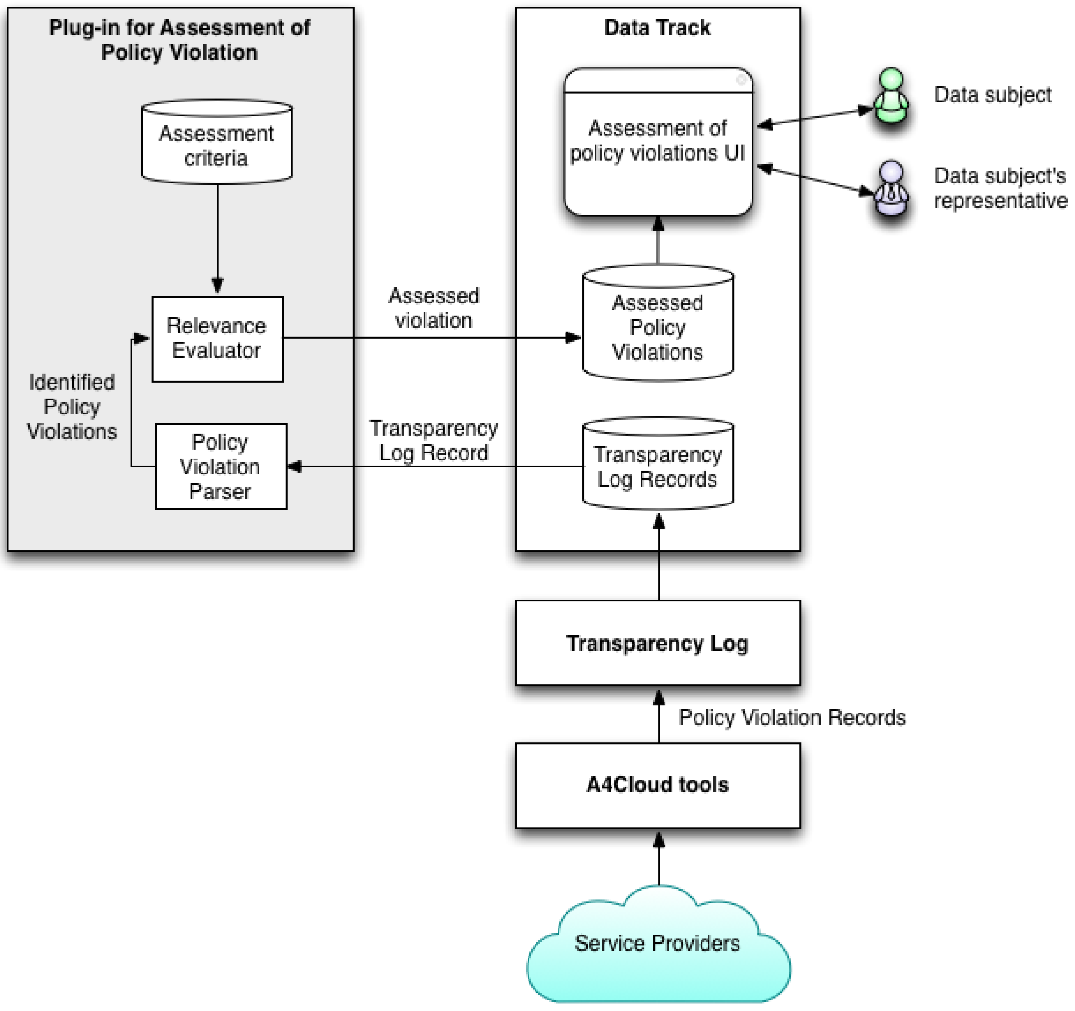


Figure 31: High level architecture of the plug-in for Assessment of Policy Violation.

PAPV does not support any UI, but the functionalities provided by it are realised through the DT UI.

5.2.3 Tool Application Programming Interfaces

The plug-in will be integrated with the DT tool development and there is no need to consume any API provided by other tools. However, the API shown in Table 17 will be offered by PAPV.

Name of the API	Purpose of use	Consumed by	Data format	API format
IPapv	Asses the relevance of the provided policy violation	DT	XML	Go library

Table 17: Interfaces provided by the plug-in for Assessment of Policy Violation.

5.3 Transparency Log

5.3.1 Tool Overview and Target Stakeholders

The Transparency Log (TL) is based on the cryptographic scheme Insynd[4] [8], which is used for secure and privacy-preserving unidirectional asynchronous communication using intermediate untrusted servers. TL uses Insynd to provide a one-way communication channel between service providers and data subjects. This enables a reliable channel for sending notifications to data subjects (who cannot always be assumed to be online to receive data) even for privacy-sensitive data that normally should not be sent by email, for example. At the core of Insynd, and therefore TL, is a forward-secure append-only persistent authenticated data structure [9].

TL is “secure'” in the sense that it provides:

- forward-secrecy of messages: any messages sent through TL are secure (secret) from future compromise of the sender. The recipient can also be forward-secure if it discards key-material (not in the default version).
- deletion-detection forward-integrity: No events[5] sent prior to sender compromise can be deleted or modified without detection.
- publicly verifiable consistency: anyone can verify that snapshots, that fix all data sent through TL, are consistent. This can be seen as a form of publicly verifiable deletion-detection forward-integrity.

TL is “privacy-preserving'” in the sense that it provides:

- forward-unlinkability of events: any two events generated before sender compromise are unlinkable, meaning that an adversary cannot tell if they are related or not. This prevents recipient profiling based on events.
- forward-unlinkability of recipient identifiers: any two identifiers used to identify recipients prior to sender compromise are unlinkable.

TL is “`asynchronous” in the sense that for a sender to send, or a recipient to receive, the other party (recipient or sender, respectively) does not have to be online. TL is also “one-way'”, meaning that a recipient cannot reply to the sender. Finally, TL also enables a sender and recipient to produce publicly verifiable proofs of

- sender: Who was the sender that sent a particular event?
- recipient: Who was the recipient of a particular event?
- message: What is the message inside of an event?
- time: When did a particular event exist, relative to time provided by a time-stamping authority?

TL is categorised in the data subject’s controls functional area, although the tool can be used by all A4Cloud tools to ensure that logging in the cloud service supply change is encrypted. Of particular importance is that TL is used by AAS as an evidence store to protect the evidence collected by AAS agents, both from the cloud environment and the other A4Cloud tools, which generate evidence (like A-PPLE and DTMT).

5.3.2 Tool Application Programming Interfaces

TL exposes the APIs shown in Table 15, in which we distinguish between the APIs offered by the TL Sender (one public and one private using HTTP basic authentication) and the TL Recipient. TL does not need any interfaces provided by other tools.

Name of the API/methods	Purpose of use	Consumed by	Data format	API format
	ITlSenderPublic (GET)			

[4] http://www.cs.kau.se/pulls/insynd/, accessed 2015-08-20.
[5] An event is a container for an encrypted message.

Name of the API/methods	Purpose of use	Consumed by	Data format	API format
/pile/:pile/recipient/:recipient	Get the current *state* of the recipient :recipient in pile :pile.	TL Recipient	JSON	RESTful
/identity	Get the identity (verification) key used to verify signatures made by the TL Sender. This key should be verified through external means, e.g., by a Certificate Authority. Not needed for testing purposes.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
/pile/:pile/event/:id	Get the event with identifier :id from the pile :pile	TL Recipient	JSON	RESTful
/pile/:pile/snapshot/	Get a list of all snapshot identifiers for the pile :pile. A snapshot fixes all data inserted into the pile up to the point of snapshot creation.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
/pile/:pile/snapshot/:id	Get the snapshot with identifier :id from the pile :pile. If :id is -1, all snapshots are returned.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
ITlSenderPrivate				
GET /pile/	Get the list of all piles	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
POST /pile/	Create a new pile	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
DELETE /pile/:pile	Close the pile :pile. This prevents future messages to be sent to all recipients in the pile. Data is not deleted.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
GET /pile/:pile/recipient/	Get the list of all recipients in the pile :pile	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
POST /pile/:pile/recipient/	Register a new recipient in the pile :pile.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
POST /pile/:pile/recipient/:recipient	Send a message to the recipient :recipient in the pile :pile	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
DELETE /pile/:pile/recipient/:recipient	Delete the recipient :recipient from the pile :pile. This prevents new messages to be sent to the recipient.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
ITlReciepient				
GET /	Get list of all key-pairs	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
POST /	Create a new key-pair	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful

Name of the API/methods	Purpose of use	Consumed by	Data format	API format
PUT /:key	Add registration data for the key :key from a TL Sender.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
GET /:key	Get all messages sent to the key :key.	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
GET /:key/:i	Get the i:th message sent to the key :key	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
GET /:key/:i/raw	Get the i:th message send to the key:key without any of the accompanying JSON returned without the /raw suffix. The response body contains the raw bytes of the message	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
GET /:key/size	Get the number of messages sent to the key :key	TL, DT, A-PPLE, AAS, DTMT	JSON	RESTful
GET /:key/verify	Verify the authenticity of all messages sent to the key :key	All A4Cloud Tools	JSON	RESTful

Table 18: Interfaces and respective API methods offered by the Transparency Log tool.

6 Incident Management and Remediation

The Incident Management and Remediation functional area offers accountability support in a corrective manner. The set of tools belonging to this category support mechanisms for remediation of accountability failures and incident response.

In the context of the A4Cloud architecture, these mechanisms are being developed within the scope of the A4Cloud software tools, namely the Incident Management Tool (IMT) and the Remediation and Redress Tool (RRT).

6.1 Incident Management Tool

6.1.1 Tool Overview and Target Stakeholders

The Incident Management Tool (IMT) is the entry point for handling incidents and detected violations (or any other anomaly) in cloud environment scenarios, such as privacy violations or security breaches. The tool receives incident reports through the API or UI and takes the initial steps to respond to these incidents, by sending alerts to the user when a relevant incident has occurred based on different parameters and the incident handler's decision. IMT is not responsible for detecting incidents. Detection may happen through detection tools (such as DTMT, A-PPLE or AAS) or manually. When an incident is detected, IMT needs to receive information about the incident and subsequently perform the following actions:

- Notify those actors that should receive information about the incident;
- Log the information about the incident and the decisions and actions taken regarding the notification.

IMT is used by cloud providers to handle incidents detected in the cloud and manage the flow of the generated notifications. In that respect, IMT runs on the data processor and data controller's side to provide the referring business actors of the respective providers and/or customers an understanding of which incidents have been detected on their cloud infrastructure. The actors responsible for handling incidents can use IMT to verify the severity of the incident notifications and generate relevant alerts for their customers. The latter can be cloud customers (in case of data processors) or the data subjects (in case of a data controller). Potentially, IMT can be used to serve the need of a Cloud Auditor to receive comprehensive information about the incidents that have occurred in the facilities of the cloud provider. In other words, some, but not all of the incidents may need to be reported to entities outside the cloud service provider (such as the cloud customer, the cloud subject and the cloud supervisory authority).

6.1.2 High Level Architecture

The high level view of the IMT architecture is depicted in Figure 32.

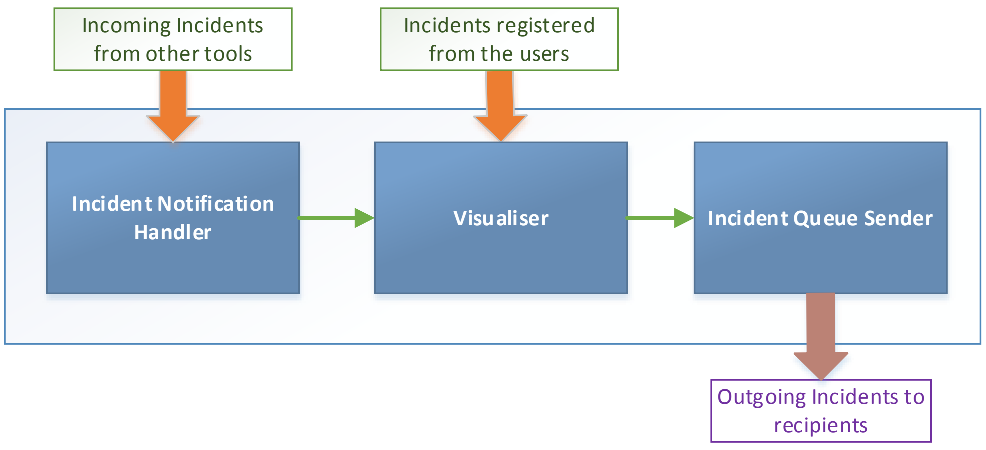

Incoming Incidents from other tools
Incidents registered from the users
Incident Notification Handler
Visualiser
Incident Queue Sender
Outgoing Incidents to recipients

Figure 32: The high level view of the Incident Management Tool architecture.

The functionality of the IMT can be divided into three main parts:

- Receiving incident notifications;
- Handling incident notification;
- Sending incident notifications.

Subsequently, the main components of this architecture are:

- The Incident Notification Handler, which listens to new incidents coming from the detection tools and other instances of IMT;
- The Visualizer, which provides the graphical interface of the tool;
- The Incident Queue sender, which decides on the recipients of the processed incidents.

The tool relies on incidents as they are detected at the cloud provider side. Incidents that can occur at the providers' side include a wide range of phenomena, such as hardware failure, data breach, policy infringements, interception/surveillance by security agencies, use of data in breach of established policies, etc. Not all incidents can be detected automatically. For instance, it may not be possible to detect that a system administrator has made a copy of the data in their system and sold it to an outsider.

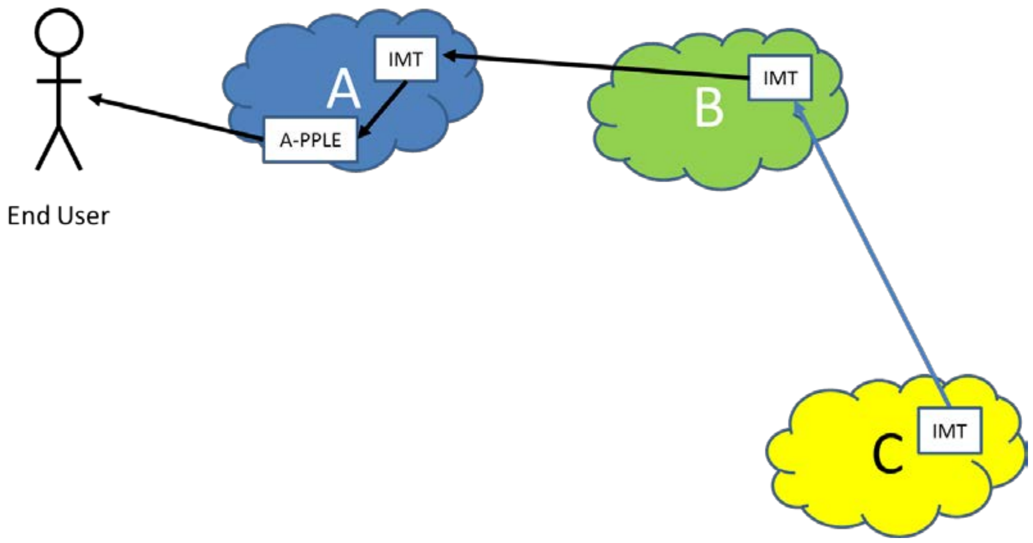
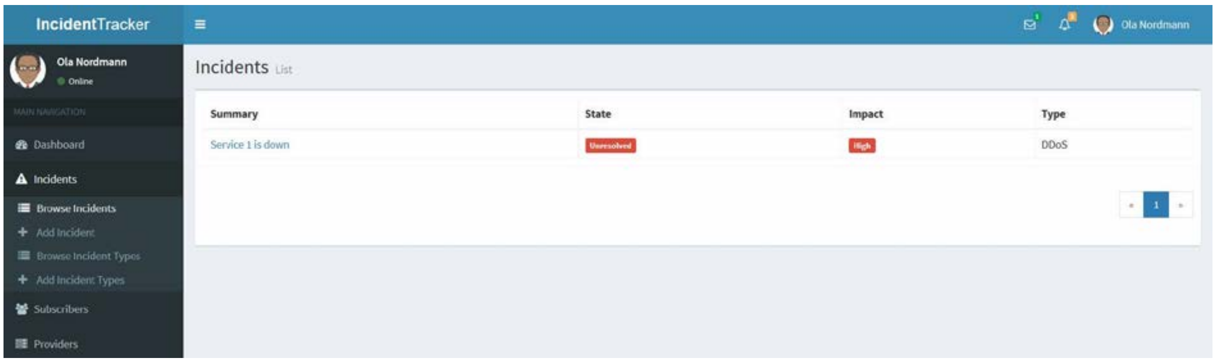


Figure 33: IMT Cloud Provider Chain

While the diagram in Figure 32 describes how the prototype currently works, there is room for multiple improvements and additions if the prototype is to be transitioned into a reference implementation or production software. Important changes include logging of all interaction with the tool, increased modularity and plugin support for, e.g., estimation of local impact and additional notification channels. Finally, as depicted in Figure 33, each cloud service provider has its own IMT instance in a complex cloud service provisioning chain.

6.1.3 User Interface

IMT features a Web User Interface (the IMT dashboard is shown in Figure 34), which is used by cloud providers to receive and send incident notifications. In order to do so, IMT needs to interface with other A4Cloud tools and receive information about policy violations and any incident happening in the cloud service chain, along with potential evidence regarding the specific incident. A graphical interface for manual registration of incidents is also provided.



IncidentTracker

Ola Nordmann
Online

MAIN NAVIGATION
- Dashboard
- Incidents
  - Browse Incidents
  - Add Incident
  - Browse Incident Types
  - Add Incident Types
- Subscribers
- Providers

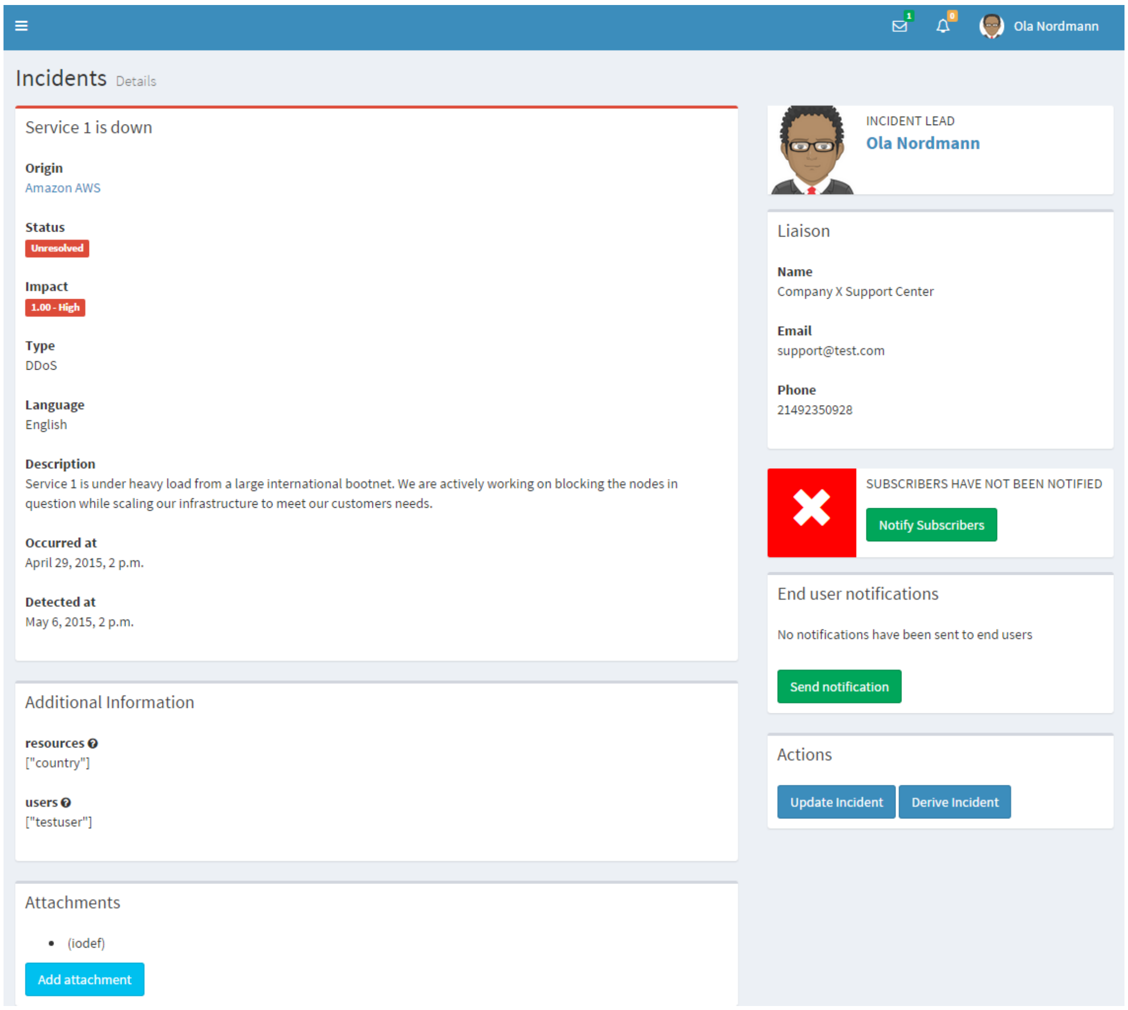
Incidents List

Summary	State	Impact	Type
Service 1 is down	Unresolved	High	DDoS

« 1 »

Figure 34: The dashboard of the Incident Management Tool.

Figure 35 shows the detailed view of an incident notification.



Ola Nordmann

Incidents Details

Service 1 is down

Origin
Amazon AWS

Status
Unresolved

Impact
1.00 - High

Type
DDoS

Language
English

Description
Service 1 is under heavy load from a large international bootnet. We are actively working on blocking the nodes in question while scaling our infrastructure to meet our customers needs.

Occurred at
April 29, 2015, 2 p.m.

Detected at
May 6, 2015, 2 p.m.

Additional Information

resources
["country"]

users
["testuser"]

Attachments
- (iodef)

Add attachment

INCIDENT LEAD
Ola Nordmann

Liaison

Name
Company X Support Center

Email
support@test.com

Phone
21492350928

SUBSCRIBERS HAVE NOT BEEN NOTIFIED
Notify Subscribers

End user notifications

No notifications have been sent to end users

Send notification

Actions

Update Incident | Derive Incident

Figure 35: The incident notification details view of the Incident Management Tool.

6.1.4 Tool Application Programming Interfaces

IMT offers an API that any detection tool can use to notify about an incident. This is the *IImt* interface, which delivers the methods shown in Table 18.

Name of the API	Purpose of use	Consumed by	Data format	API format
/incidents/types	Manage* incident types available for subscription	IMT and any subscriber	JSON	RESTful
/incidents/types/{id}/triggers/types	Manage* the trigger types available for the incident type	IMT and any subscriber	JSON	RESTful
/subscriptions	Manage subscription of incidents from a provider	Subscribers	JSON	RESTful
/subscriptions/{id}/incidents	Manage instantiated incident types for subscription	Subscribers	JSON	RESTful
incidents/{id}/triggers	Manage instantiated trigger types for the incident in a subscription	Subscribers	JSON	RESTful
/notifications/validate	Used to validate notifications**	Subscribers	JSON	RESTful
Configurable endpoint when creating subscription, e.g. /receive	Notify IMT of an incident	DTMT, A-PPL-E, AAS	JSON	RESTful
*Owner of instance is able to POST, GET and DELETE. Subscribers are only able to GET **Currently not implemented				

Table 19: Interfaces and respective methods provided by the Incident Management Tool.

Furthermore, IMT consumes the APIs provided by other tools and environments, as shown in Table 20.

Name of the API	Purpose of use	Should be offered by	Data format	API format
Notification	Initiate the notifications to the end users subject to rules of the accountability policies	A-PPLE	JSON	RESTful

Table 20: Interfaces needed by the Incident Management Tool.

6.2 Remediation and Redress Tool

6.2.1 Tool Overview and Target Stakeholders

The Remediation and Redress Tool (RRT) aims to assist individual data subjects in responding to (perceived) incidents in their cloud arrangement, and the mitigation of the difficulties that data subject are reported to have in accessing judicial or administrative remedies. It is activated as a result of certain incidents reported by IMT to the relevant stakeholders or can be invoked by the user on the basis of information coming from other sources (such as a request for data disclosure from DT).

If the tool is triggered by an incident raised by IMT, then RRT knows what type of incident has occurred and what possible actions can be taken. Then, it will guide the user through these actions, which include getting targeted guidance, performing a number of corrective measures aiming at the mitigation of the incident, and obtaining an account to be used in case the subject decides, as a final measure, to seek judicial redress. In case the tool is consulted by the user without being triggered by IMT, it will engage in a dialogue with the user to establish their concern and next guide the user through appropriate actions. Where appropriate, the tool will take automatic action to communicate requests/complaints etc.

We underline that the tool is not aimed at providing legal advice *strictu sensu*, nor at automating or incentivizing any part of a possible legal action.

6.2.2 High Level Architecture

Figure 36 shows the high level view of the RRT architecture, which is similar to the one for IMT. RRT gets as input the incident data (in the form of type of incident, time and scope), some user related information (e.g. about the location, the allocated roles, the contact details, etc.), any contextual information that can assist in making proper decisions on remediation and the incident response model retrieved from a knowledge base. The latter may include a list of actions for the data subjects that relate to the type of generated incident.

In this context, RRT integrates mechanisms for analysing the incoming incident information and providing explanations, mapping the user concern or incident to the knowledge base, synthesizing the response and producing the relevant response. The output of the tool is a list of potential remediation and/or redress actions, including pre-completed (standard) forms for complaints/request etc. It also educates the respective stakeholders on incidents and potential actions and procedures.

The main components of the RRT architecture are:

- The Response Listener, which listens for new notifications coming from the detective tools, through DT.
- The Response Logger, which logs the information about the actions decided and taken by the data subjects.
- The Dialogue Manager, which is the graphical component of RRT and enables for the interaction with the data subject in order to conclude on the appropriate remediation or redress action.
- The Response Generator, which processes the received incidents to propose appropriate actions.
- The Remediation Queue Sender, which handles the processing of the selected response actions (to DT).

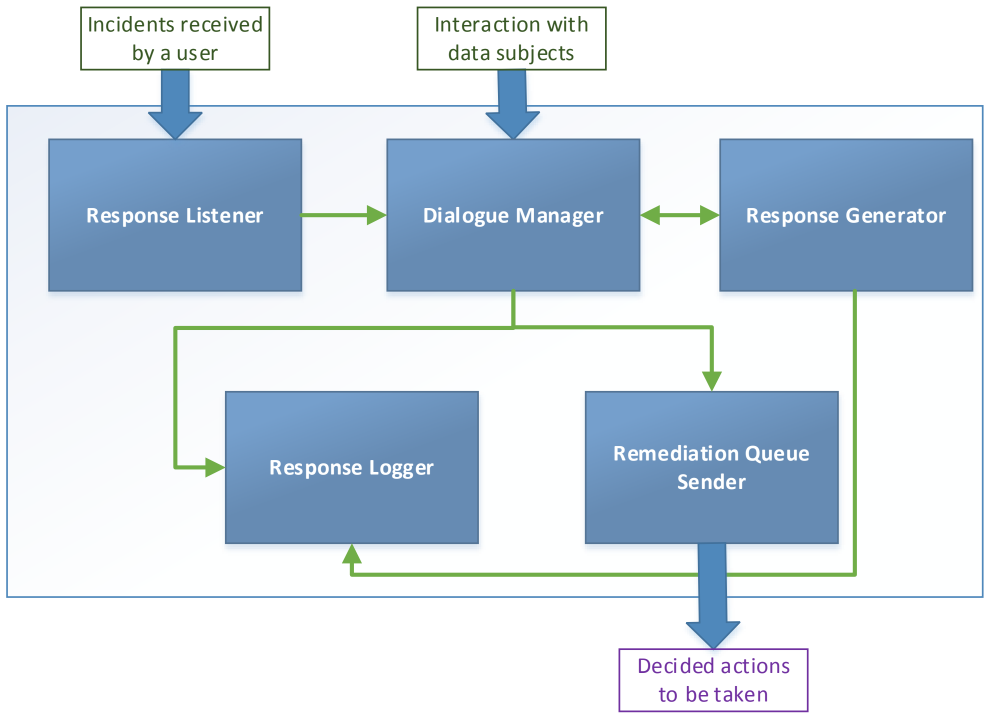
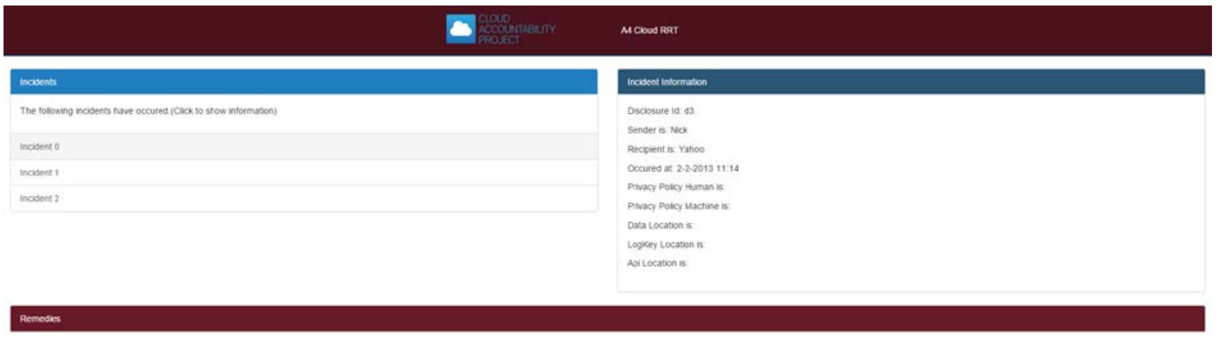


Figure 36: The high level architecture of the Remediation and Redress Tool.

6.2.3 User Interface

RRT offers a Web User Interface (see Figure 37), which integrates the functionalities for visualising the incident notification and composing a remediation request action.



The following incidents have occured.(Click to show information)

Incident 0

Incident 1

Incident 2

Disclosure Id: d3

Sender is: Nick

Recipient is: Yahoo

Occured at: 2-2-2013 11:14

Privacy Policy Human is:

Privacy Policy Machine is:

Data Location is:

LogKey Location is:

Api Location is:

Figure 37: The graphical view of the Remediation and Redress Tool.

6.2.4 Tool Application Programming Interfaces

Currently, the tool does not offer any interface, but it consumes the API offered by DT, as shown in Table 21.

Name of the API	Purpose of use	Should be offered by	Data format	API format
Notifications	Send the list of the received notifications	DT	JSON	RESTful
Actions	Provide the list of actions in response to a received notification	DT	JSON	RESTful

Table 21: Interfaces needed by the Remediation and Redress Tool.

7 Summary of the tool interactions

In this section, we make an attempt to provide the potential connections of the A4Cloud tools with each other in the context of implementing the accountability support services and artefacts of the Cloud Accountability Reference Architecture.

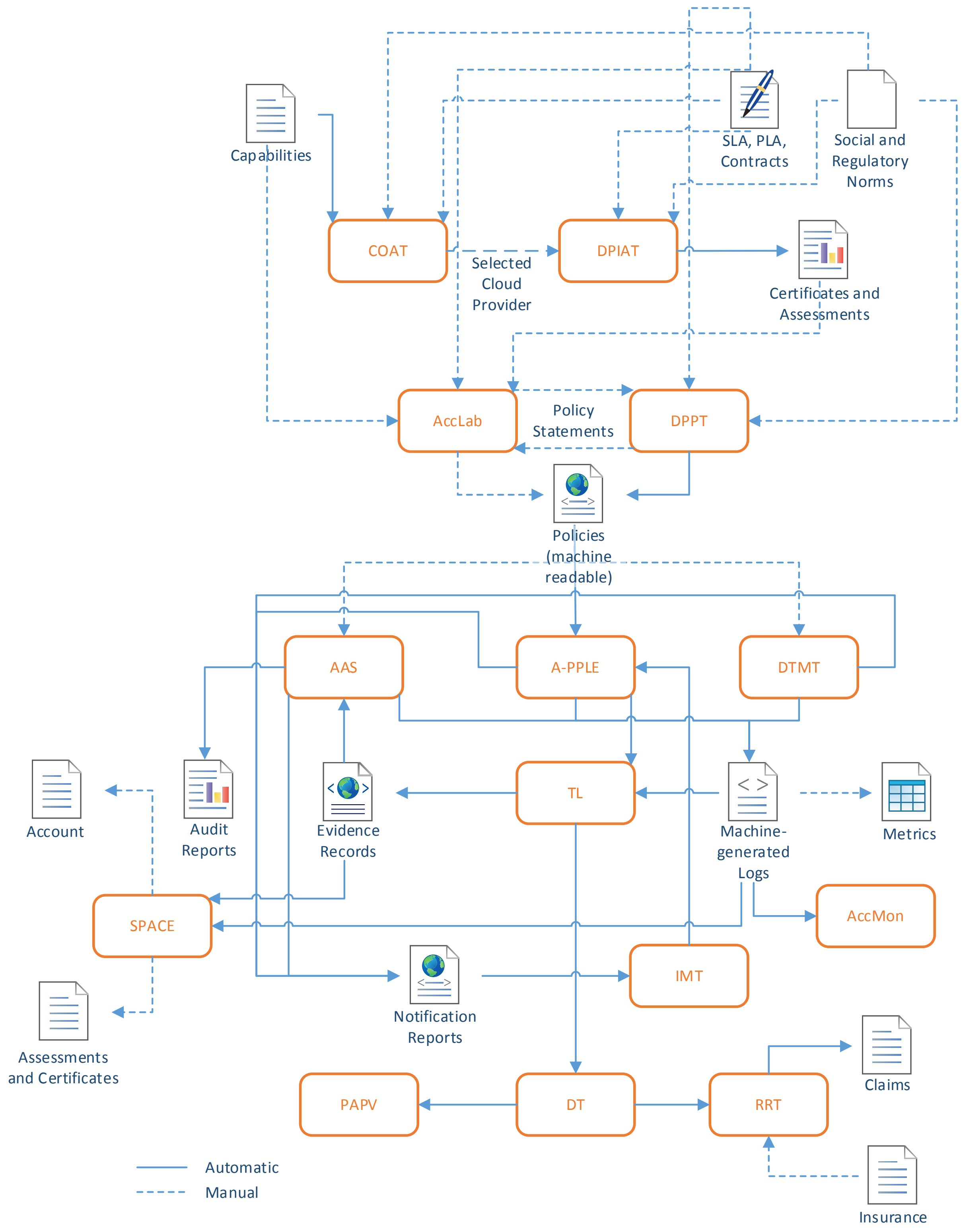


Figure 38: The interaction diagram of the A4Cloud tools.

In that sense, Figure 38 shows the interactions between the A4Cloud tools to accomplish the functions of accountability, as they are identified in the relevant life cycle, and the respective accountability support services. For these interactions, we, also, demonstrate the involvement of the accountability artefacts, as an important input or output for each A4Cloud tool. In this figure, we emphasise on the tool interactions through these artefacts and we distinguish between interactions happening as a result of a manual human intervention (which mostly implies the consumption of the artefact in the respective user interface of the tool) or through relevant communication protocols, which enable an automatic interaction between the tools.

In the rest of this section, we analyse these interactions, in the context of the accountability support services.

Policy Definition and Validation

This service enables the cloud providers and customers to define and configure machine-readable policies, which are based on the functional, security and privacy requirements of the involved actors and the development of policy terms in accordance to their **social and regulatory norms** or their contractual obligations, as they are defined in the agreed **SLAs, PLAs or contracts**. These obligations are used in this service to allow cloud providers specify their **Capabilities**, in terms of their cloud service offerings, which take the form of machine-readable contracts. The policy definition and validation accountability support service is realised through the use of:

- COAT for the capabilities-based selection of a compliant cloud service provider.
- DPIAT for the impact assessment regarding the use of a specific cloud service provider to process personal data. Data Protection Impact assessment is a type of **certificate and assessment** artefact, which is based on the defined organisational and business operations, exhibiting certain **capabilities** with respect to functional, security and privacy provisions, and the resulting obligations, which are compliant to **social and regulatory norms** and the already established **SLAs, PLAs or contracts**, between the selected provider and their third parties.
- DPPT for the compilation of machine readable **policies**, based on the human readable accountability policies and the abstract policy statements, with respect to the **capabilities** of the selected cloud provider, the existing **SLAs, PLAs or contracts** and the **social and regulatory norms**, which the provider has accepted to take responsibility over.
- AccLab for the contract and policy matching when preparing the machine readable **policies**, which is based on the abstract policy statements of the providers, expressing their **capabilities** from a data protection perspective.

The policy definition and compliance accountability support service is based on a human readable representation of the privacy policies, which is translated to A-PPL. This machine-readable specification of the **policies**, in the form of A-PPL, includes the personal data to be handled by the providers and the specific accountability related rules dictating data handling procedures.

Policy Management and Enforcement

This service involves the execution of the data processing practices, in accordance to the machine readable **policies**. The policy enforcement part engages the use of A-PPLE as the enforcement engine to allow and manage data processing operations. While an A-PPL based **policy** is automatically fed into A-PPLE, a manual (human defined) task for the proper configuration of the selected cloud service supply chain environment is required. Thus, AAS and DTMT are involved in this service to be configured, according to the **policy** provisions.

Monitoring and Environment State Collection

This service is realised through the use of the following A4Cloud tools, which are used to generate and/or collect **machine-generated logs**:

- A-PPLE, which generates logs with respect to actual decisions made in the policy enforcement part.
- DTMT, which monitors the cloud environment networking part and generates logs with respect to data transfers identified in it.

- AAS, which monitors the various layers in protocol stack of the cloud service delivery models (SaaS, PaaS, IaaS, etc.) and collects logs that may relate to potential security breaches or policy violations.
- TL, which is involved as the secure and encrypted logs collection channel. It must be clarified that TL may not be a single component per log collection process in the regime of a cloud provider.
- AccMon, which supports monitoring the implementation of accountability policies through collecting logs from the components of a real system.

As part of this service, the logs are exploited to assess the accountability maturity of the cloud providers, through the calculation of the respective **metrics** and the creation of **certificates and assessments** (like AMM).

Collection and Management of Evidence

During this accountability support service, the **machine-generated logs** collected by the A4Cloud tools in the previous service are used to compile **evidence records**. This service also refers to the management of logs within their full lifecycle, according to specific integrity, confidentiality and access control requirements.

Incident Management

The analysis of **machine-generated logs** and **evidence records** provides a realisation of the incidents that can emerge during the operation of the accountability support services. These incidents are raised by A-PPLE, DTMT and AAS, thus the tools, which contribute to the creation of **evidence records** and are notified to IMT. The incidents may refer to potential policy violations and security breaches. Furthermore, IMT may be used to manually register incidents into the environment, through an end-user assessment.

Notification

This service enables the communication of verified incidents to the appropriate recipients in the form of **notification reports.** Such notifications enumerate the types of the detected incidents and other information specific to these incidents. The Notification accountability support service is implemented through the use of IMT and drives the triggering of the next service, which is the Remediation accountability support service.

Remediation

This service contributes to the exception handling process and refers to the remediation and redress actions that should be adopted in response to the discovery of an incident and the notification to the relevant recipients. The response part is implemented through the use of RRT, which assists cloud end users in responding to real or perceived data handling incidents, either by guiding them in inquiring about capitalising the provisions of an already agreed **insurance** and compiling **claims** or by supporting the actual implementation of redress actions.

Validation

This accountability support service involves the Data Subject Controls tools and, particularly, DT. The tool is used by data subjects to manually provide an **assessment** on the compliance of the cloud providers with the claimed **policies**.

Following the evidence collection support service, the validation service engages AAS for the demonstration of compliance to established and agreed data processing practices through the execution of (both internal and external) audits. The result of this process is the generation of **audit reports**, including **evidence records** and related objects such as related **machine-generated logs** and machine readable **policies**.

Finally, the validation accountability support service engages SPACE, which is used by cloud service providers to provide assurance about the way that the providers comply with the agreed accountability policies. Through this tool, the cloud service providers can demonstrate their compliance to data handling procedures and manually provide an **account** and an **assessment** on their effectiveness to address accountability, security and privacy for continuous assurance.

8 Conclusion

This document presented the specifications of the tools comprising the A4Cloud toolkit. It analysed their internal structure and the interfaces exposed to other tools, as well as those interfaces required by each tool to accomplish the expected functionalities envisaged for this tool. The tools were logically presented in five functional areas, which are introduced to highlight the role of the tools in the implementation of accountability. As such, in Figure 39, we place the tool categories in the various phases of the lifecycle for accountability.

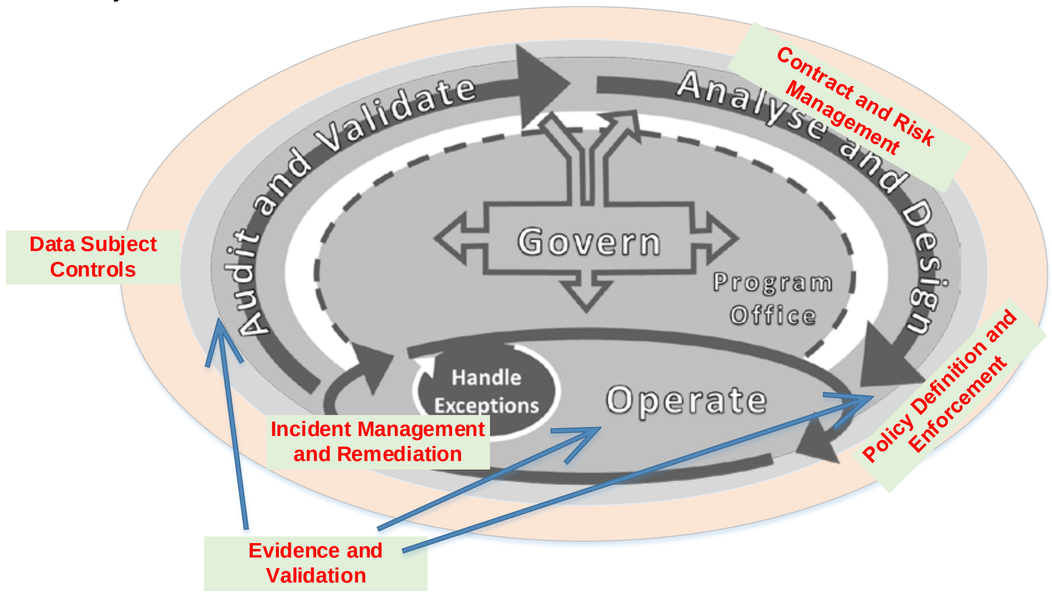


Figure 39: The functional tool categories in the lifecycle for accountability.

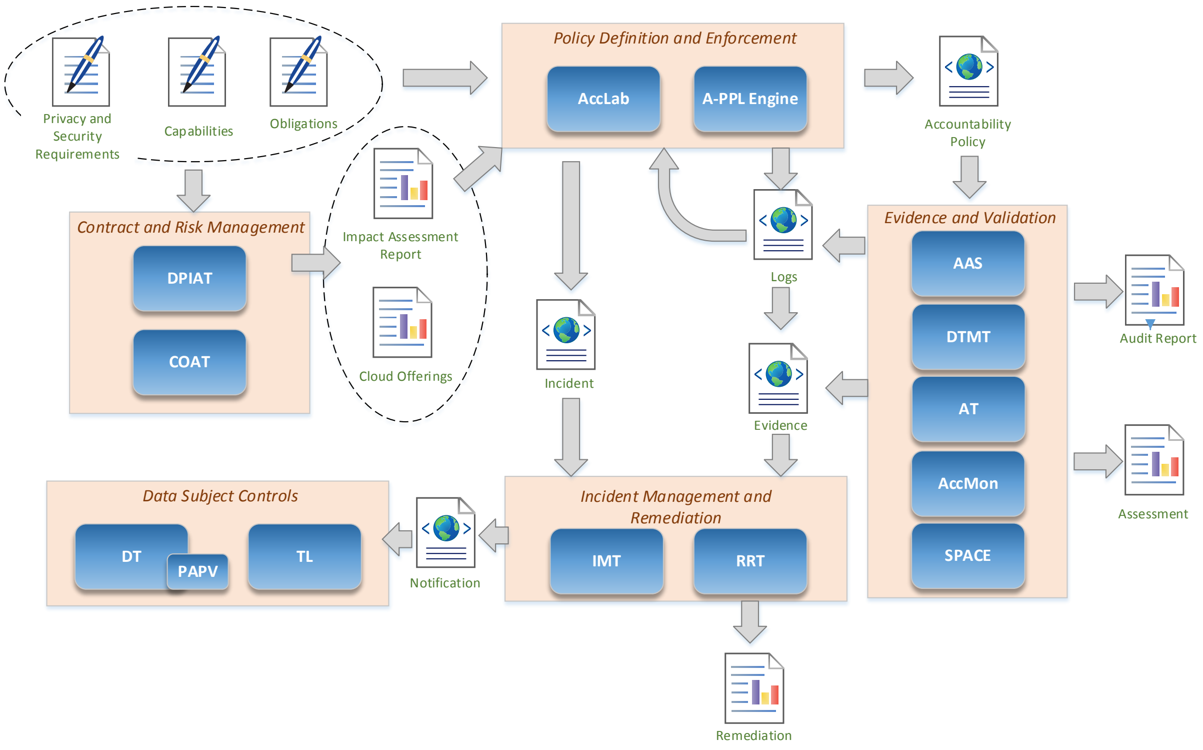


Figure 40: Overview of the tools comprising the A4Cloud toolkit and their interactions.

The document is provided as an Annex to the Cloud Accountability Reference Architecture and the respective A4Cloud Deliverable D42.4. It complements the reference architecture by presenting the way that the accountability support services and artefacts are instantiated through reference implementations to facilitate the defined preventive, detective and corrective accountability mechanisms. In that sense, the toolkit does not aim to prevail as the only implementation environment of the reference architecture, but as the means to showcase how the accountability concepts and framework are technically realised as a complete solution in cloud settings and respective business paradigms.

Further to it, the tools fit to the integration and adoption patterns of the reference architecture and analyse the technical means that the different stakeholders are equipped with to address the accountability problem in the cloud. The tools integrate with each other and with an external cloud environment and implement the accountability support services and the respective artefacts, as shown in Figure 40. The reference implementation of the tools has been used in the development of the A4Cloud use case prototype in WP47.

9 References

[1] Erdal Cayirci, Alexandr Garaga, Anderson Santana De Oliveira, Yves Roudier: A Cloud Adoption Risk Assessment Model, Proceedings of the CLASP 2014, International Workshop on Advances in Cloud Computing Legislation, Accountability, Security and Privacy, in conjunction with the 7th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2014), pp. 908-913, 8-11 December 2014, London, United Kingdom, DOI: 10.1109/UCC.2014.148.

[2] Cloud Security Alliance (CSA), Consensus Assessments Initiative Questionnaire v3.0.1 (CAIQ), https://cloudsecurityalliance.org/download/consensus-assessments-initiative-questionnaire-v3-0-1/.

[3] ENISA, Cloud Computing risk Assessment, https://www.enisa.europa.eu/activities/risk-management/files/deliverables/cloud-computing-risk-assessment.

[4] ENISA, Inventory of Risk Management / Risk Assessment Tools, https://www.enisa.europa.eu/activities/risk-management/current-risk/risk-management-inventory/rm-ra-tools.

[5] Cloud Security Alliance (CSA), CSA Security, Trust & Assurance Registry (STAR), https://cloudsecurityalliance.org/star/

[6] Apache SOLR, lucene.apache.org/solr/.

[7] Cloud Security Alliance (CSA) Privacy Level Agreement: https://downloads.cloudsecurityalliance.org/initiatives/pla/Privacy_Level_Agreement_Outline.pdf.

[8] Tobias Pulls and Roel Peeters, Insynd: Privacy-Preserving Transparency Logging Using Balloons, Cryptology ePrint Archive, Report 2015/150, 2015, http://eprint.iacr.org/.

[9] Tobias Pulls and Roel Peeters, Balloon: A Forward-Secure Append-Only Persistent Authenticated Data Structure, Cryptology ePrint Archive, Report 2015/007, 2015, http://eprint.iacr.org/.

10 Index of Figures

11 Index of Tables