Monitoring Personal Data Transfers in the Cloud

Anderson Santana de Oliveira*, Jakub Sendor*, Alexander Garaga* and Kateline Jenatton†
*SAP Labs, France
{anderson.santana.de.oliveira | jakub.sendor | alexander.garaga}@sap.com
†EPFL, Switzerland
kateline.jenatton@epfl.ch

***Abstract*—Cloud computing brings a number of compliance risks to organisations because physical perimeters are not clearly delimited. Many regulations relate to the location of the data processing (and storage), including the EU Data protection directive. A major problem for cloud service consumers, acting as data controllers, is how to demonstrate compliance to data transfer constraints. We address the lack of tools to support accountable data localization and transfer across cloud software, platform and infrastructure services, usually run by data processors. In this paper we design a framework for automating the collection of evidence that obligations with respect to personal data handling are being carried out in what concerns personal data transfers. We experiment our approach in the OpenStack open source IaaS implementation, showing how auditors can verify whether data transfers were compliant.**

***Keywords*-Cloud computing; Security; Accountability; Compliance; Data tracking; Auditing;**

I. Introduction

Cloud computing presents tremendous advantages to organizations of all kinds and sizes, who can extend their information processing resources dynamically. The use of these on demand services reduces expenses on technology infrastructure, improves flexibility, and accessibility to applications and systems. The downside of the adoption of cloud services is that it raises a number of accountability questions, as parts of the risk and compliance management are delegated to third parties, the cloud service providers (CSPs). In order to be compliant with privacy regulations, data controllers using a cloud service to process personal data need means to verify compliant data handling (including possible international data transfers) across the cloud service provisioning chain.

An open problem is how to deal with compliance implications related to data location in cloud ecosystems (combining service supply chains involving software, platform and infrastructure as a service). In the cloud, physical perimeters are not clearly delimited. This characteristic allows CSPs to effectively and profitably use cloud resources, balancing the workload through data transfers and storage across services and different cloud infrastructures (which may have different jurisdictional restrictions).

We frame our contributions on the EU data protection regulation [1] and related recommendations on the use of cloud computing [2]. The latter clarifies the typical roles related to the legislation for the different cloud actors: usually the Cloud Consumer outsources parts of its IT processes to a CSP. In the case the processing involves personal data from data subjects within the European Economic Area, the Cloud Consumer is a data controller and the CSP is a data processor. Any subcontractor processing data in the cloud on behalf of the CSP is also considered as a data processor. These actors have different obligations with respect to the data processing and safeguard, but all need to give account about where the data is processed.

A number of mechanisms exist to allow international personal data transfers, such as Safe Harbour agreement and Binding Corporate Rules [3], [4]. Once these are approved by the relevant Data Protection Authorities (DPAs), it is difficult to obtain transparency from the CSPs on how data is transferred in the cloud. Today, cloud contracts rarely allow to specify where data is to be geographically stored. For example, in the study [5] only one CSP allowed its customers to select whether their data should be stored in the United States or the European Union. We advocate that the next generation of cloud services will emphasize on accountability for processing business and personal data, providing the assurance that international data transfer constraints are respected.

On the other hand, CSPs lack nowadays appropriate means to demonstrate compliance. We address the lack of tools to support accountable data localization and transfer across cloud platforms, services and infrastructures. Our goal is to collect evidence that data processing only occurs at allowed locations from the perspective of the EU data protection regulation. Our assumption is that whichever specific regulation applies (even within the EU where regulatory requirements are defined at a national level), reliable data localization and accountability mechanisms are fundamental for audits and to dispute resolution purposes.

In this paper we design a framework for automating the collection of evidence that obligations with respect to personal data handling are being carried out in what concerns personal data transfers. We define how accountability services can generate and protect evidence about data transfers across outsourced layers, e.g. the platform and infrastructure levels in the cloud service delivery chain. We experiment our approach in the OpenStack [6] open source IaaS implementation.

Our solution allows data controllers and auditors (mandated by the data protection authorities) to reliably answer the following questions: *a*) Who processes the personal data and where is it stored in the cloud?; *b*) Have all personal data transfers in a cloud service provisioning chain been compliant to the privacy obligations regarding data location?; *c*) When,

where, and by whom data transfer operations were performed on personal data in the cloud?

The remainder of the paper is organized as follows: Sec. II elicits the requirements for location-aware personal data handling in the cloud based on related privacy obligations from EU data protection laws; Sec. III analyzes the challenges in data tracking for different delivery models; Sec. IV describes the related work; Sec. V provides a cloud scenario that serves as a case study; Sec. VI introduces the proposed architecture; Sec. VII analyzes the threats to the proposed solution and outlines possible mitigations; and Sec. VIII concludes the paper.

II. Data Location and Accountability

In this section we discuss the implications of data location to the compliance with the European data protection regulation [1] and recommendations [2] for the cloud. We explain the responsibilities of data controllers and processors in the cloud.

The lack of transparency brings a number of risks to data controllers and subjects in the cloud. Cloud consumers (assuming the role of data controllers) may simply be unaware of implicit chains of processing of personal data in the cloud, because further subcontracting is done by the CSP, making it impossible to know where, who and when data is being processed in the cloud. In terms of the European regulation, independently of where the processing of data in the cloud takes place, data controllers are primarily accountable for fulfilling the privacy obligations, including its limitations concerning transfers to third countries [7]. Data controllers have to make sure, via contracts, that the data processors will guarantee their duty of confidentiality concerning personal data. Further subcontractors also need to be bound to the same data processor obligations.

A central problem is therefore asserting where data is processed in a reliable, transparent and responsible manner. This has been recognized by the Cloud Security Alliance (CSA) who includes an element of transparency to the Cloud Trust Protocol (CTP) about data geographic location [8], without, however, further guidance on how evidences about the rightful processing can be collected and provided consistently. In the CSA Cloud Control Matrix (CCM) [9], CSPs can self-assess whether they "demonstrate transparency in the usage of third party services and compliance to laws and regulation requirements by documents and evidences". But more fine grained control is needed when dealing with personal data, as the purposes of the processing have an impact on the allowed transfers to third parties and third countries.

Below we discuss obligations and recommendations to data controllers and processors, closely related to the location of the data processing, as recommended in [2]:

- The data processor should notify the data controller about all his subcontractors contributing to the provision of the respective cloud service as well as about the locations of all the data centers where personal data may be processed.
- The data controller needs to inform the data subjects about the recipients of their personal data. This may include processors or sub-processors as far as this information is necessary to guarantee "fair processing in respect of the data subject". Data location becomes a fundamental element of transparency in this case, as all countries clearly do not provide the same level of privacy.
- The data controller needs to ensure that the cloud provider's processing activities are auditable, in order to demonstrate that only allowed operations were performed by the cloud provider or one of his subcontractors, as well as capture where the processing took place.
- The data controller needs to be aware of multiple copies of personal data, and it must be ensured that in the event of deletion, each instance of them is erased permanently (i.e., previous versions, temporary files and even file fragments are to be deleted as well).

III. Data tracking challenges in the cloud

In order to effectively enforce privacy obligations the data should be tracked at all levels in the cloud ecosystem (application, platform, infrastructure). Currently the relationships between the virtual and physical data locations are not transparent to the cloud consumers (data subjects, data controllers) [10], thus hindering accountability. The virtualization performed by a IaaS solution is usually dynamic and opaque to the cloud users. Physical borders of the cloud infrastructures are also vague, hence often data controllers are not aware of where the data resides (at which data processor, in which country).

There are two types of transfers: the data can be transferred either vertically, that is from one service layer to another, or horizontally, i.e. in a given layer, data is moved due to elasticity, load balancing, backup, etc. Examples of horizontal transfers are: VM migration to another host (infrastructure level), migration of a tenant to another database server instance (platform level), as well as replications performed by the CSP for enabling disaster recovery, that are usually unknown to the cloud consumer; examples of vertical transfers are: storing data subject's personal data in a database record, flushing a database record to disc. During data transfers in the cloud the association between the data representation at a given service level, its sensitivity, the responsible data controller and related obligations is often lost. Monitoring both vertical and horizontal data transfers and keeping this link to the data controller's obligations is essential to provide accountability.

In this paper we assume that the data processors wish to demonstrate compliance, therefore they will not move cloud consumer data without authorization. That is difficult to demonstrate, as there are many ways to access and copy data. To reliably demonstrate compliance, CSPs need to limit the standard set of operations performed over their customers' data to the cloud service API calls. By adhering to this principle, it is possible to identify which operations over the platform and infrastructure imply data transfers and thus need to be monitored.

One can argue that CSPs will not likely restrict their operations in this way. On the other hand, the full market potential of the cloud can only be reached by building on transparency and trust. The cloud service provider has no interest in doing unauthorized data transfers, and a way to demonstrate that is to integrate to their landscape an accountability service that will be able to provide data about the compliance of the operations.

IV. Related Work

The need for accountability in the cloud is highlighted in [10] and in [11], which explain the benefits of achieving data accountability. They underline the lack of transparency regarding the virtual-to-physical mapping in the cloud, which we also try to address in this paper.

A solution for bringing accountability to the cloud is suggested in [12] and [13]. Several entities (trusted and semi-trusted) exchange signed messages and tokens to provide provable and non-disputable evidences that the operations performed by the services are compliant with the Service-level Agreements (SLAs) and are "legitimate with respect to the business logic". But this addresses business exchanges at SaaS level, and personal data handling is not considered. Only [13] includes data location information in the collected evidences, but no further information is given on its use.

The HyTrust[1] tool is a virtual appliance that intercepts all the administrative requests for a virtual infrastructure, and determines if these requests are in accordance with the policies defined by the virtual infrastructure manager. This is a preventive tool that can deny or allow non-compliant requests, by enforcing the policies applied to the labels given to virtual objects. The HyTrust team advertises that the "policies are fully customizable and flexible enough to handle any complex situation", but it is unclear whether this tool can restrict data location in an accountable way. Similarly, the work in [14] presents a preventive system for data location compliance that allows the data controller to specify his requirements for data geolocation, enabling the Virtual Execution Environment Manager to place the virtual machines according to these requirements. From this perspective, these systems are more advanced than the detective tool we describe in the next sections. However, these two solutions only address the IaaS layer and do not provide a way to map the data transfers across the different layers. Indeed, data movements at the PaaS and SaaS levels are not detectable by these architectures, whereas our solution can, for instance, detect at the SaaS layer that a CRM application is transferring data to a billing system. Moreover, in [14], the monitoring architecture is designed on top of the RESERVOIR architecture, thus creating a strong dependency. In contrast, our work is based on widely used cloud computing platforms and APIs.

V. Use Case

In our research we took a closer look at the accountability and the compliance concerns arising in a use-case involving enterprise cloud application built upon all three different cloud service delivery models. In Figure 1, the described software application is a cloud CRM system operated by an imaginary supermarket chain called MarchéAzur, based in France, who plays the role of the data controller. It launched a fidelity program for its customers (the data subjects from all over Europe), supported by a mobile application (delivered in a Software-as-a-Service fashion). This gives them the possibility to create a virtual wish-list basket. Customers can add the products from the supermarket's on-line catalog and manage the list of products they would like to purchase during the next visit to the supermarket. Moreover, they will be able to benefit from personalized shopping offers, based on their shopping habits and their virtual basket content. We assume that the data subjects have been properly informed of the purpose, the intermediate processors, and provided their consent for the personal data collection.

The SaaS relies on further outsourcing to run. It uses the PaaSPort - Platform-as-a-Service provider for the supporting Java-based web services middleware and database servers. We assume that PaaSPort is a company based in Germany, here acting as a data processor. For transparency reasons, PaaSPort has indicated to its cloud consumer, MarchéAzur, that it further outsources its infrastructure where web and database servers run using contracts and further legally binding documents.

In order to assure high availability PaaSPort uses the InfraRed - Infrastructure-as-a-Service provider, who maintains data centers in Europe (Ireland) - the one required by PaaSPort, but also in other countries, US, Australia, etc. InfraRed uses OpenStack to provide its computation, image, network, and storage services. In this scenario InfraRed is also a data processor. In addition to obligations regarding confidentiality, integrity, and availability of the personal data collected by MarchéAzur, the data processors have to comply with the restrictions agreed with the data controller about the data location.

In our running example, we will show the mapping of the database service administrated by PaaSPort, and deployed on a virtual machine hosted by InfraRed, which is itself associated with a number of volumes. The information concerning physical location of the servers hosting the database server instance needs to be formally provided by InfraRed, preferably stated in the contracts. Then, we can use this data for the compliance audits: at the time services are contracted, we can establish the mapping from more abstract cloud service layers SaaS and PaaS to the virtual machine instances at the infrastructure level.

The database will contain personal data from MarchéAzur's customers. We associate an identifier to this data set, which we will use for tracking the data movements across the layers. We are not distinguishing, however, the sensitivity level of the personal data. For instance, if the SaaS is handling health care data and less sensitive data as well, then the whole data set would be treated with the higher classification. At the SaaS level, we will observe activities concerning personal data transfers to external domains, at the PaaS level – record information

[1] http://www.hytrust.com/product/overview/

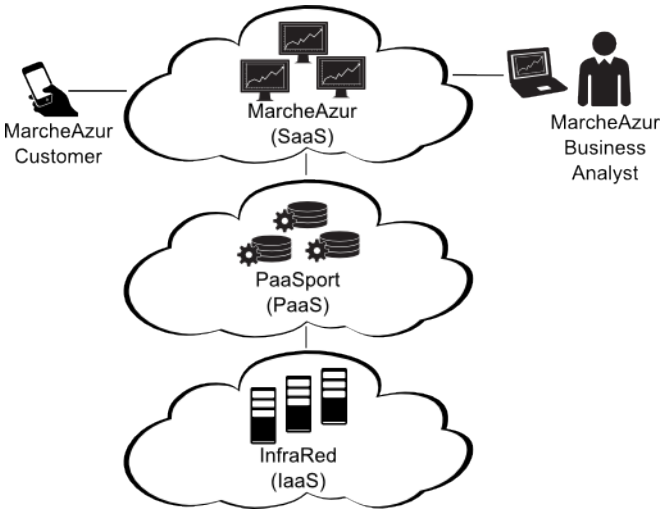

Fig. 1. Sample use-case scenario

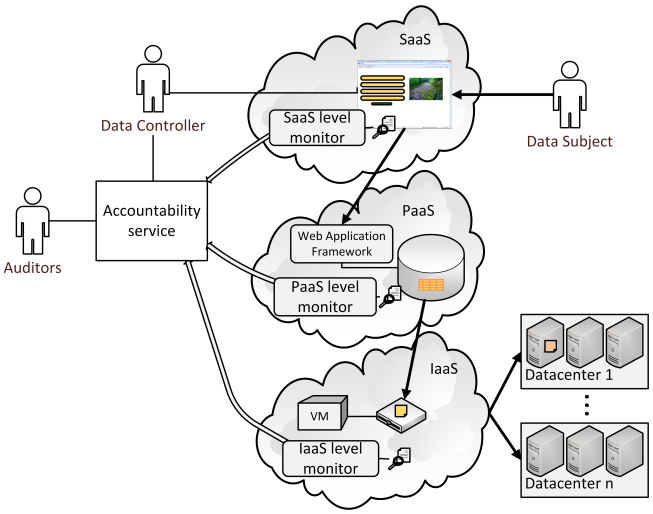

Fig. 2. Data tracking architecture

about database backups, migrations, and duplications; and at IaaS level – observe how instances, volumes, etc. are allocated to physical servers.

VI. An Architecture for Tracking Data in the Cloud

In this section we describe our architecture for monitoring data transfers in the cloud. Figure 2 presents the architecture for accountable personal data tracking in the cloud. The architecture is generic and can be used not only for monitoring personal data transfers, but also potentially any sensitive data. This may include business sensitive data (e.g. financial records, product designs) providing competitive advantage.

At each service layer, we add a data transfer monitor (DTM) that tracks and logs all personal data movements. They are described in the next sections on SaaS, PaaS and IaaS monitoring. DTM segregates data for each distinct pair of personal data set and data controller using cryptographic means, further

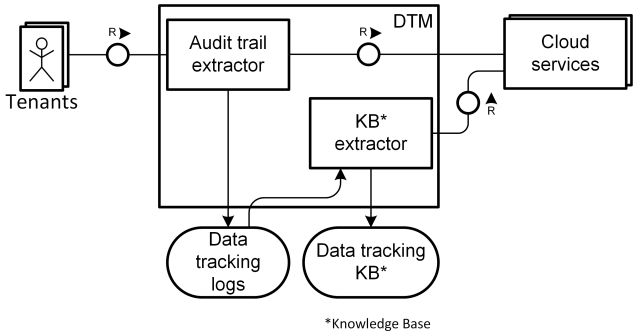

Fig. 3. Data Tracking Monitor Architecture

detailed in Section VII. The information collected by DTMs is then processed by the accountability service (AS), from which auditors and data controllers can check compliance to data privacy regulations and contracts.

A. Audit trails

In order to enable the correlation between the events from different service levels coming from various sources, we define a common audit trail structure, adapted from [10]:

$\langle Actor, Operation, When, Where \rangle$

Where

- $\langle Actor \rangle$ is the initiator of the operation and can be the tenant id, CSP operator name, etc.
- $\langle Operation \rangle$ is the action on the personal data item. Ideally all operations concerning personal data should be tracked. The actual operations are very dependent on the implementation and on the service level. In order to normalize these differences we distinguish the following operation categories relative to data tracking: Create, Read, Update, Delete, Copy — which are mapped to actual operations.
- $\langle Data \rangle$ is the identifier of the data object, under which the operation is performed. It is linked with a personal data category and data controller.
- $\langle When \rangle$ is the event date and time.
- $\langle Where \rangle$ is the data processing location. Depending on the service level this can be a database server instance (platform level), a virtual disk volume, a physical host IP or the geographic location (infrastructure level).

Depending on the service level these elements differ and are described in the following sections. Audit trails coming from different sources are normalized according to this format, translated into logical facts and stored in the DTM knowledge base (detailed below).

B. Data transfer monitors

Figures 3 illustrates the DTM architecture. In this approach DTM has a proxy that monitors all API calls from tenants (e.g. data controller or cloud platform administrator) to the cloud services, and extracts the audit trail in the format defined in Sec. VI-A. The DTM can optionally query the cloud services for some additional information. Further, the audit

trail is used to construct data tracking knowledge base that represents operations on personal data as logical facts, suitable for automated analysis by the AS.

Below, we show how the DTMs can be integrated and used at the SaaS, PaaS and IaaS layers, focusing on concrete technologies and related challenges.

C. SaaS and PaaS monitoring

We show some examples of operations at the platform level influencing on data transfers. In the use case presented in Section V, a database server instance is shared by multiple tenants at the application (SaaS) level. We associate the tenant id to the data controller and the corresponding data subject's personal data set identifier. This can be done at the moment of the tenant creation, for instance. Below we present some examples inspired from the SAP HANA Cloud Platform API [15] . For database tenant creation, the RESTful PaaS level API calls observe the following schema:

URL	http://(host)/persistence/admin/tenants/(name)
Method	PUT
Returns	201 Created and tenant JSON 400 Bad Request 403 Forbidden 500 Internal Server Error

A concrete example in JSON format would be as follows:

```
http://paasport.com/persistance/admin/MarcheAzur/fidelity
HTTP/1.1 200 OK
X-Compute-Request-Id: req-c461e07-9b18-4d9c-898b-37262fd9063
Content-Type: application/json
Content-Length: 1420
Request Method: PUT
Date: Mon, 27 May 2013 07:42:02 GMT

{
"create" : {
"name" : "MarcheAzur",
"database" : "paasport_instance_1",
"creation_time" : "2013-07-16 3:11 pm",
"creator" : "pass_provider_db_admin",
"dbuser" : "db_admin",
"password" : "admin1234"
}
}
```

The response for this request is the following

```
{ "database_space": {
        "id": "34465727-8c79-22a0-6b24-567f565bc83c",
        "tenant_id": "MarcheAzurFR",
        "creator" : "paas_provider_db_admin",
        "dbuser" : " db_admin",
        "name": "fidelity_program",
        "created": "2013-07-16T15:11:10Z",
        "hostId": "e4d909c290d0fb1ca068ffaddf22cbd0",
        "jdbcconnection":
           "jdbc:sap:dbdriver://dbhost1.paasport.com:2323",
}
```

A second example of operation monitored at the PaaS level is to migrate a tenant database to another database server instance with distinct performance:

```
http://ma.paasport.com/persistance/admin/MarcheAzur/fidelity
HTTP/1.1 200 OK
X-Compute-Request-Id: req-c41e07-9b18-4d9c-898b-3762fd9063
Content-Type: application/json
Content-Length: 1420
Request Method: PUT
Date: Mon, 27 May 2013 07:42:02 GMT

{       "migrate_tenant" : {
        "name" : "MarcheAzur",
        "database" : "paasport_instance_2",
        "creation_time" : "2013-07-17 00:11 am",
        "creator" : "pass_provider_db_admin",
        "dbuser" : " db_admin",
        "password" : "admin1234"
        }
}
```

Whose response would consist in the following:

```
{       "database_space": {
        "id": "34465727-8c79-22a0-6b24-567f565bc83c",
        "tenant_id": "MarcheAzurFR",
        "creator" : "pass_provider_db_admin",
        "dbuser" : " db_admin",
        "name": "MarcheAzur",
        "created": "2013-07-16 T00:15:10Z",
        "hostId": "e4d909c290d0fb1ca068ffaddf22cbd0",
        "jdbcconnection":
           "jdbc:sap:dbdriver://dbhost2.paasport.com:2323",
}
```

In summary, many operations at the platform level can have implications to data transfers. The strength of our approach is to provide mappings from a service delivery layer to the underlying ones.

At the SaaS level, APIs are application specific. The SaaS provider is then responsible to adopt a privacy management approach and to identify which sensitive parts of the business process can transfer personal data from and to external systems, possibly under the control of third parties, and subject to regulatory constraints. Specific privacy concerns for the use case described here exist, some of them are mentioned in [16].

D. IaaS monitoring

At the infrastructure level, we monitor read-write accesses from upper layers or transfers occurring within the infrastructure, or between two IaaS providers. In this work we analyzed relevant operations provided by the OpenStack API [17]. OpenStack is an open source cloud solution that provides compute, network and storage services. Any operation required by a tenant or an admin user is performed through the API.

In OpenStack, we distinguish three types of entities that can hold data: instances, volumes and object stores. Instances are virtual servers managed by the compute service. They can store data in their file system. Volumes are block storage entities. A tenant can create a volume of the desired size and attach it to an instance. A volume can only be attached to a single instance at a time, but can be detached and attached to any instance. It can be seen as an external hard drive being plugged to a virtual machine. Object store is a persistent storage for static data. Creation, configuration and deletion of these entities can be detected by monitoring the API calls to the services that manage them.

Data can then be transferred within the infrastructure due to mechanisms such as snapshot, replication, instance migration, etc. For instance, let us assume there is a virtual machine, in which is stored a file with the mail addresses of the data subjects. When a snapshot of the instance is required, the mail addresses will be copied and included in the snapshot. The

snapshot is managed by the OpenStack image service, thus will be stored in the host that owns the service. From this point, the tenant can create several virtual machines using this snapshot. Therefore, the mail addresses will also be stored in the new instances that are booted on the snapshot image. When the data is stored in a volume, it will also be duplicated when a snapshot of the volume is requested. Then, a new instance can boot on the volume snapshot, or a new volume can be created from this snapshot, resulting in a data duplication in both cases. At the volume creation, a scheduler defines the storage node on which it will be located. The OpenStack Object Storage service can be used to store backups of volumes and instances. The objects stored in OpenStack are replicated to ensure availability. Thus, a file will be copied at least on three disks, preferably located in different availability zones in the OpenStack cloud. For load-balancing purposes, a data replicate might be moved from one place to another. Due to load balancing operations or after a (physical) host failure, an instance might be migrated from one server to another. All the operations described above result in duplication of data, where the data duplicate could either be located in the same region or country, or in a different one.

Data transfers can also occur between two IaaS providers. Indeed, one could use a cloud infrastructure for computing services and another one for storage services. We can again use the OpenStack example, that has a compatible API to the Amazon S3 API. Thus, we could easily imagine a scenario involving OpenStack and Amazon clouds together.

In an OpenStack environment, the DTM monitors all the API calls to log the events and to fill the data tracking knowledge base. It also has to send API requests to the OpenStack services in order to get the details about the servers that are not present in the monitored API calls. In particular, if the tenant requests a VM migration, the new host is not given in the response message. Moreover, some information are not given to a normal tenant, while it can be obtained if the tenant has an admin role. We assume our tool would get the same level of information as an admin tenant.

The Knowledge Base Extractor uses a logical knowledge engine called PyKE [18], which allows performing logic programming in Python. Logic programming lends itself well to this case, since it makes it straightforward to perform pattern matching on messages, and to generate decisions based on available facts. The following code excerpt illustrates facts from the data tracking KB.

```
#Volume Snapshot creation request
#volume_snapshot_creation(name, description, volume_id)

volume_snapshot_creation_req(database_volume_replication,
volume_Snapshot_for_PaaSPort_DB_Duplication,
e26a64348a5040458a524d5fff7d313e)
```

These facts showing the volume snapshot creation are built from the original request given below, intercepted by the DTM:

```
POST /v1/93859cf00e7741d4bdb6a37285cc827a/snapshots HTTP/1.1
        Host: 10.55.129.42:8776
        Content-Length: 145
        x-auth-project-id: admin
        accept-encoding: gzip, deflate
        accept: application/json
        x-auth-token: 0d608bb2235040ffb557fef2bbee826d
        user-agent: python-cinderclient
        content-type: application/json

        {"snapshot": {"display_name":
           "volume_Snapshot_for_PaaSPort_DB_Duplication",
        "force": "True", "display_description": null,
        "volume_id": "e26a6434-8a50-4045-8a52-4d5fff7d313e"}}
```

The HTTP response contains the following information:

```
HTTP/1.1 200 OK
        X-Compute-Request-Id:
          req-2260d35f-6503-4aa2-aa35-7a1a1558c5b0
        Content-Type: application/json
        Content-Length: 251
        Date: Tue, 25 Jun 2013 06:57:00 GMT

        {"snapshot": {"status": "creating",
        "display_name":
          "volume_Snapshot_for_PaaSPort_DB_Duplication",
        "created_at": "2013-06-25T06:57:00.463182",
        "display_description": null,
        "volume_id": "e26a6434-8a50-4045-8a52-4d5fff7d313e",
        "id": "0e55163e-794f-4712-87d6-ec10e9070941",
        "size": 1}}
```

In our running example, this corresponds to creating a copy of the volume containing the PaaSPort database server instance. Below we show part of the rule set for tracking volume duplication and attachment to server instances in OpenStack. These capture the fact that volume snapshots can be used to boot server instances. The `holds_pii` predicate is used to "tag" an infrastructure object as containing personal data for a given data subject group (`data_subjects_id`):

```
personal_data_propagation_volume
foreach
  iaas_level_rules.holds_pii($instance_id, $data_subjects_id)
  iaas_level_rules.os_volume_attach($instance_id, $volume_id)
assert
  iaas_level_rules.holds_pii($volume_id, $data_subjects_id)
personal_data_propagation_vol_snapshot
foreach
  iaas_level_rules.volume_snapshot_creation($name,
    $description, $volume_id)
  iaas_level_rules.volume_snapshot_creation_resp($name,
       $description, $volume_id, $snapshot_id )
  iaas_level_rules.holds_pii($volume_id, $data_subjects_id)
assert
  iaas_level_rules.holds_pii($snapshot_id, $data_subjects_id)
personal_data_propagation_vol_creation
foreach
  iaas_level_rules.create_volume_req($snapshot_id, $name,
    $availability_zone, $attach_status, $project_id)
  iaas_level_rules.holds_pii($snapshot_id, $data_subjects_id)
  iaas_level_rules.create_volume_resp($status,
    $display_name, $snapshot_id, $volume_id)
assert
  iaas_level_rules.holds_pii($volume_id, $data_subjects_id)
personal_data_propagation_vol_attach
foreach
  iaas_level_rules.os_volume_attach($instance_id, $volume_id)
  iaas_level_rules.holds_pii($volume_id, $data_subjects_id)
assert
  iaas_level_rules.holds_pii($instance_id, $data_subjects_id)
```

E. Topology

In order for the AS to get the physical location of data related to a specific data controller, we introduce a topology knowledge base that captures the mapping between the virtual machines, images and volumes on one side and their physical representation (availability zones, network, host) on the other side.

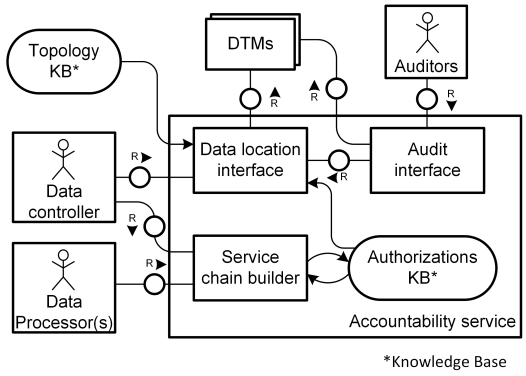

Fig. 4. Accountability service architecture

In particular, it describes the physical boundaries of the infrastructure, the mapping between virtual servers and physical hosts and the location of the hosts. The mapping is provided and managed by the infrastructure provider, in contrast to AS, and hence its trustworthiness is linked to that of the CSP. As data processor, the the IaaS provider is responsible for creating and maintaining this information up to date, as it is the only entity aware of the constantly changing physical landscape of the cloud infrastructure. We estimate that the cost of maintaining this information is negligible, as network administration practices already require to keep this kind of record. The following code excerpt illustrates facts contained in this knowledge base.

```
#host(host_name, host_id)
host(Infrared_IE, e9ef8cf20d89f8ee6cfa6)

#Determines where a host is located
#host_location(host_name, location)
host_location(Infrared_IE, Ireland)

#Instances for a given host
# instance_host(host_id, instance_id)
instance_host(e9ef8cf20d89f8ee6cfa6,
  a4ab4ef7c92264d1c8c14958f8d6f4318)
```

The excerpt above shows the mappings among data processors and the corresponding data handling objects at the infrastructure level. Additionally, it also relates the hosts maintained by the infrastructure provider to their actual physical location.

F. Accountability service

Figure 4 illustrates the architecture of the AS. The main component in AS is Data location interface, that enables data controllers and auditors to determine the location of the data subjects' data entrusted to the data controller. In particular, it returns all data processors holding the data subjects' data, the sensitivity category of personal data and the physical storage location at the country level. This allows data controllers to check their state of compliance with the privacy obligations: they could determine potential unauthorized transfers of personal data to other parties and storage in a country, not considered offering sufficient data privacy protection guarantees.

In order to provide this response Data location interface aggregates information from all the DTMs — more precisely from Data tracking KBs — in the cloud ecosystem of this particular data controller. In addition, it makes use of the Topology KB (see VI-E) in order to derive the physical location of the data; and Authorizations KB, that captures the data processors and other third parties that are authorized by data controller to process its data subjects' personal data. This database is filled together by the data controller and data processors (data processors can further delegate the processing to other parties with the prior permission of 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.

The following code snippet shows possible facts in Authorizations KB.

```
data_subjects(1000, Europe)
data_controller(MarcheAzur, 1000, France, SaaS)
data_processor(PaaSPort, Germany, PaaS)
data_processor(Infrared, Ireland, IaaS)
authorized_party_transfers(MarcheAzur, PaaSPort, 1000)
authorized_party_transfers(PaaSPort, Infrared,1000)
authorized_location_transfers(Europe, Germany, 1000)
```

The following code shows a forward chaining query to identify the multiple locations the data for a given data subject set.

```
personal_data_location
 foreach
  iaas_level_rules.instance_host($host_id, $instance_id)
  iaas_level_rules.host($host_name, $host_id)
  iaas_level_rules.host_location($host_name, $location)
  iaas_level_rules.holds_pii($instance_id, $data_subjects_id)
 assert
  iaas_level_rules.personal_data_location($data_subjects_id,
   $location)

with fc_goal.prove(engine,
 data_subjects_id1=data_subjects_id1) as gen:
 for vars, plan in gen:
  print "%s is located in %s" % \
   (data_subjects_id1, vars['location1'])
```

Auditors could check all operations performed on personal data and verify if there was any possible violation of privacy obligations, in particular during disputes. This also relates to the Cloud Trust Protocol, as it provides a way to respond to the Element of Transparency number 8 which requests disclosure about geographic location of "units (including data sets) being used on behalf of the consumer"[8].

VII. Security Considerations

Here we describe some possible threats to this solution and proposed mitigations. The main assumption in this work is that the CSPs are trusted and they would configure their service so that all control operations flow through DTMs. However, we still consider various malicious insider threats, some of which are mentioned below.

a) An adversary tampers with the logs or knowledge bases generated by a DTM: Our solution has to provide logs to the auditors that are trustworthy, by protecting their integrity. The principle of *forward security* [19], [20], [21] can be applied to the logs, that is an attacker must not be able to undetectably modify or delete log entries that were generated before the compromise. Integrity of logs or

data tracking knowledge bases can be protected by adding a Message Authentication Code (MAC). In this case, the DTM could sign all log entries with a private key, provided e.g. by a Trusted Platform Module (TPM)[2] and Auditors could verify their integrity. We also consider that our DTMs will consume trusted timestamps and that further configuration data have also been certified by audits to minimize this risk.

b) An adversary steals the logs generated by a DTM: Audit trails generated by DTMs (see Sec. VI-A) can contain sensitive personal data. Thus, the confidentiality of data tracking logs has to be protected. Confidentiality of the logs can be obtained through a combination of symmetric and asymmetric encryption (for performance reasons), where the private key is shared between the data controller and Auditors. Moreover, to ensure segregation between the different data controllers, the encryption keys will be specific to each data controller.

c) An adversary tampers with the DTM: As the DTMs are located in the CSP's premises (for performance reasons) they are specifically vulnerable to tampering. In order to ensure the generation of evidence by DTMs is not tampered with, apart from proper physical and logical access control, they could use TPMs for remote attestation of DTM's configuration.

d) An adversary is intentionally bypassing DTM: If an adversary (e.g. an insider) uses a covert channel to transfer sensitive data instead of the service APIs, potential violations can happen without being detected. This risk is somewhat diminished in case of frequent onsite audits, but should still be taken into account. DTMs can be incremented with additional rules to identify such data transfers by observing events at the OS level and at the networking interfaces.

VIII. Conclusion

In this paper we presented an architecture to monitor data transfers in the cloud, supporting evidence collection to demonstrate compliance with regulations concerning international data transfers. Our data tracking monitors can be integrated to distinct delivery models (SaaS, PaaS, IaaS), from which events reflecting data movements are correlated using a rule engine, fed with knowledge extracted from real world cloud APIs — for example, we experimented our approach against OpenStack, for the IaaS level. In this way we are able to answer diverse questions about the compliance of data controllers and of the data processors with respect to data localization. As future works, we will extend the approach with preventive and corrective measures, as to enforce additional policies and obligations regarding personal data, but also other kinds of sensitive data in the cloud.

Acknowledgment

This work was partially supported by the Cloud Accountability project - A4Cloud[3], grant EC 317550. We also thank Jean-Christophe Pazzaglia and Melek Önen for their valuable comments on a previous version of this paper.

[2]http://www.trustedcomputinggroup.org/resources/tpm_main_specification

[3]http://www.a4cloud.eu/

References

[1] European Parliament and the Council, "Directive 95/46/EC of the European Parliament and of the council of october 24 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data," October 1995. [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML

[2] Article 29 Data Protection Working Party, "Opinion 05/2012 on cloud computing," July 2012. [Online]. Available: http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2012/wp196_en.pdf

[3] US Department of Commerce, "US-EU safe harbour framework," http://export.gov/safeharbor/eu/eg_main_018365.asp, 2012.

[4] European Commission, "Overview on binding corporate rules," http://ec.europa.eu/justice/data-protection/document/international-transfers/binding-corporate-rules/index_en.htm, 2013.

[5] Cloud Standards Customer Council, "Public cloud service agreements: What to expect and what to negotiate," http://www.cloud-council.org/publiccloudSLA.pdf.

[6] OpenStack, "The OpenStack project," http://www.openstack.org/, 2013.

[7] R. Gellman, "Privacy in the clouds: risks to privacy and confidentiality from cloud computing." in *Proceedings of the World privacy forum,*, 2012.

[8] R. Knode and D. Egan, "Digital trust in the cloud: Into the cloud with ctp — a precis for the cloudtrust protocol, v2.0," https://cloudsecurityalliance.org/wp-content/uploads/2011/05/cloudtrustprotocolprecis_073010.pdf, 2010, cloud Security Alliance.

[9] Cloud Security Alliance, "Cloud controls matrix v3.0 draft," 2013.

[10] R. K. L. Ko, P. Jagadpramana, M. Mowbray, S. Pearson, M. Kirchberg, Q. Liang, and B.-S. Lee, "Trustcloud: A framework for accountability and trust in cloud computing," in *SERVICES*. IEEE Computer Society, 2011, pp. 584–588.

[11] S. Pearson, V. Tountopoulos, D. Catteddu, M. Südholt, R. Molva, C. Reich, S. Fischer-Hübner, C. Millard, V. Lotz, M. G. Jaatun, R. Leenes, C. Rong, and J. Lopez, "Accountability for cloud and other future internet services," in *CloudCom*. IEEE, 2012, pp. 629–632.

[12] J. Yao, S. Chen, C. Wang, D. Levy, and J. Zic, "Accountability as a service for the cloud," in *IEEE SCC*. IEEE Computer Society, 2010, pp. 81–88.

[13] ——, "Accountability services for verifying compliance in the cloud," *IJCC*, vol. 1, no. 2/3, pp. 240–260, 2012.

[14] P. Massonet, S. Naqvi, C. Ponsard, J. Latanicki, B. Rochwerger, and M. Villari, "A monitoring and audit logging architecture for data location compliance in federated cloud infrastructures," in *IPDPS Workshops*. IEEE, 2011, pp. 1510–1517.

[15] SAP, "SAP HANA cloud platform," http://scn.sap.com/community/cloud-platform, 2013.

[16] P. Yu, J. Sendor, G. Serme, and A. S. de Oliveira, "Automating privacy enforcement in cloud platforms," in *DPM/SETOP*, ser. Lecture Notes in Computer Science, R. D. Pietro, J. Herranz, E. Damiani, and R. State, Eds., vol. 7731. Springer, 2012, pp. 160–173.

[17] OpenStack, "OpenStack API reference," http://api.openstack.org/api-ref.html, [Online, accessed 11-July-2013].

[18] B. Frederiksen, "Applying expert system technology to code reuse with pyke," in *PyCon*, 2008. [Online]. Available: http://pyke.sourceforge.net/PyCon2008-paper.html

[19] B. Schneier and J. Kelsey, "Cryptographic support for secure logs on untrusted machines," in *Proceedings of the 7th conference on USENIX Security Symposium - Volume 7*, ser. SSYM'98. Berkeley, CA, USA: USENIX Association, 1998, pp. 4–4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1267549.1267553

[20] D. Ma and G. Tsudik, "A new approach to secure logging," *TOS*, vol. 5, no. 1, 2009.

[21] A. A. Yavuz and P. Ning, "Baf: An efficient publicly verifiable secure audit logging scheme for distributed systems," in *ACSAC*. IEEE Computer Society, 2009, pp. 219–228.