Project title: Enforceable Security in the Cloud to Uphold Data Ownership
Project acronym: ESCUDO-CLOUD
Funding scheme: H2020-ICT-2014
Topic: ICT-07-2014
Project duration: January 2015 – December 2017

D1.4
Final Evaluation Report from Use Cases

Editors: Géry Ducatel (BT)
Ali Sajjad (BT)
Reviewers: Neeraj Suri (TUD)
Pierangela Samarati (UNIMI)

Abstract

The implementation of the ESCUDO-CLOUD Use Cases leverages research provided in WP 2-4. Use Cases are addressing issues and objectives highlighted in D1.1 which aims to provide improved technology for the benefit of consumers. Those objectives and associated requirements are summarised here in order to validate the state of implementation. This document provides a report on the technical implementation, and core functionalities that feed into exploitation plans of IBM, SAP, BT, EMC, and WT. Use Case 1 successfully provides independent KMS for server side encryption. Key functionalities including secure deletion of encrypted material, and consistent marshalling of data commitment are available as prototypes. Use Case 2 implements the onion encryption for privacy preservation, and an order preserving encryption scheme which allows some valid operations on encrypted text. The technology is implemented for individuals. This will also lead to the implementation of group level access management. Use Case 3 has been fully implemented, it provides three types of encryption with user managed keys in the Cloud, with block and object storage encryption in production. Big Data encryption capability is also available as a proof-of-concept. Use Case 4 allows secure sharing of objects stored in public Cloud. It has been fully implemented and provides a valuable means of sharing data while respecting user privacy.

Type	Identifier	Dissemination	Date
Deliverable	D1.4	Public	2017.12.31

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644579. This work was supported in part by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract No 150087. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission or the Swiss Government.

ESCUDO-CLOUD Consortium

1.	Università degli Studi di Milano	UNIMI	Italy
2.	British Telecom	BT	United Kingdom
3.	EMC Corporation	EMC	Ireland
4.	IBM Research GmbH	IBM	Switzerland
5.	SAP SE	SAP	Germany
6.	Technische Universität Darmstadt	TUD	Germany
7.	Università degli Studi di Bergamo	UNIBG	Italy
8.	Wellness Telecom	WT	Spain

Versions

Version	Date	Description
0.1	2017.11.30	Initial Release
0.2	2017.12.14	Second Release
1.0	2017.12.31	Final Release

List of Contributors

This document contains contributions from different ESCUDO-CLOUD partners. Contributors for the chapters of this deliverable are presented in the following table.

Chapter	Author(s)
Executive Summary	Géry Ducatel (BT)
Chapter 1: Use Case 1: OpenStack Framework	Mathias Björkqvist (IBM), Christian Cachin (IBM), Björn Tackmann (IBM)
Chapter 2: Use Case 2: Secure Enterprise Data Management in the Cloud	Daniel Bernau (SAP), Andreas Fischer (SAP), Anselme Kemgne Tueno (SAP)
Chapter 3: Use Case 3: Federated Secure Cloud Storage	Ali Sajjad (BT), Géry Ducatel (BT)
Chapter 4: Use Case 4: Elastic Cloud Service Provider	Sabine Delaitre (WT), Ignacio Campos (WT), Andrew Byrne (EMC)
Chapter 5: Conclusion	All Partners

Contents

List of Figures

Executive Summary

This deliverable reports on the final evaluation of the four ESCUDO-CLOUD Use Cases. These Use Cases have been developed and implemented under Tasks 1.1 and 1.3 of Work Package 1. Task 1.1 identified the detailed functional and non-functional requirements of the services central to the four Use Cases, whereas Task 1.3 was concerned with developing the prototype applications for each Use Case by leveraging the tools and technologies developed in WP2, WP3 and WP4.

The purpose of this document, is to provide an evaluation of the state of development and integration of the four ESCUDO-CLOUD Use Cases. These Use Cases have been developed and implemented under T1.1 and T1.3 of WP1. T1.1 identified the detailed functional and non-functional requirements of the services central to the four Use Cases, whereas Task 1.3 was concerned with developing the prototype applications for each Use Case by leveraging the tools and technologies developed in WP2, WP3 and WP4. The functional requirements are provided with a unique ID, a short description of the system feature and priority (indicated by High, Medium, or Low). Each functional requirements is uniquely identified with a sequence number and associated with a system feature. This clear itemization enables the inference of a complete requirements catalogue for each Use Case. The system features are the software capabilities that must be present in order for the user or system to carry out the services provided by the feature, or to execute the Use Case. They are formulated on the basis of user actions or system responses that are needed to accomplish the goals of the Use Case scenario. A complete set of these requirements catalogue is present in [SDR15] and [BK15].

This deliverable also assesses the applications and solutions developed in T1.3 against the ESCUDO-CLOUD Use Cases and their goals and requirements as specified in D1.1 and D1.2. The purposes of this testing and evaluation activities are two-fold. Firstly, to find out if the requirements of the Use Cases fixed in T1.1 have been met, and secondly, to assess whether the tools and techniques identified or developed in the project are usable in an operational and practical situation. In addition to this technical evaluation, this deliverable also reports on findings that can serve as the basis for the exploitation of ESCUDO-CLOUD technology by the industrial partners, or act as future directions guiding further development.

The double objective of the evaluation is firstly to focus on the requirements of the Use Cases and whether they have been met. Secondly, this documents points at different techniques and tools that can be usable in a real-world and operational Cloud environment bringing innovative solutions. The potential trade-offs between security, performance, cost and functionality, evident from the Use Case prototypes, are compared with the requirements gathered in T1.1 [SDR15], [BK15]. This evaluation of functional and non-functional properties of the prototypes should help in the exploitation of the work carried out in ESCUDO-CLOUD by the industrial partners after the end of the project, and can act as the directions or suggestions for further future development.

Use Case 1 implemented data-at-rest encryption in the OpenStack Swift Cloud object storage system, and developed scalable key-management techniques for OpenStack Swift and Barbican.

A flexible, *hierarchical key management* was developed in which each object stored in Swift object storage service is protected with an individual key, and these object keys are wrapped by keys that are on higher levels in the hierarchy. It also develops an efficient key-update operation for implementing *secure deletion* for data-at-rest protection in Swift. All of the functional requirements of Use Case 1 have been implemented, with a majority of them in production stage, whereas the advanced key management functionality is scheduled for inclusion in future versions of OpenStack.

Use Case 2 implemented encrypted query processing and secure multi-party computation with the industrial grade SAP HANA database in the Cloud. This approach applies to the outsourcing of supply chain interactions in the aerospace engine maintenance industry. The technological challenge of this Use Case is to develop new supply chain cooperation systems, based on encrypted database technology in the Cloud. This technology is based on the search and aggregation of encrypted data. All of the functional requirements along the ESCUDO-CLOUD dimensions were met by enhancing the SEEED JDBE Driver, the SEEED Multi-Tenancy, Key Management and Onion Encryption components, and the introduction of a new Oblivious Order-Preserving Encryption component.

Use Case 3 implemented the Data Protection as a Service (DPaaS) solution to provide BT customers with the means to store and retrieve encrypted data from different types of Cloud-based storage services. This specific Use Case is designed to ensure that the confidentiality and integrity of the customer's data when it is outsourced for storage on different Cloud storage services is protected. The primary challenge addressed by Use Case 3 is to offer the key management and policy-based access control capabilities to customers through a Cloud service store. Each customer should be able to apply key release rules and conditions, taking into account user access rights. In the context of Use Case 3, three types of storage services are addressed; block storage, object storage and Big Data (HDSF) storage services. All of the functional requirements of Use Case 1 have been implemented, with a majority of the components of block and object storage encryption are in production stage, whereas most of the components of Big Data encryption are are prototype stage.

Use Case 4 implemented secure file sharing in multi-Cloud framework, secret storage, devices and sessions management. This Use Case considers the implementation of an elastic Cloud provider, targeting a data storage service. The resulting reference architecture and prototype is of particular relevance to Cloud service brokers and private Cloud providers that offer secure, accessible and scalable storage in the Cloud. The architecture is also applicable to organisations hosting local datacenters that intend to employ Cloud bursting to exploit relatively cheap storage resources of the Cloud. enabling the provision of a competitive product in the current market. All of the functional requirements along the ESCUDO-CLOUD dimensions were met by addressing the hybrid Cloud storage provisioning challenge by enabling the IT departments to securely provision storage resources from an untrusted third party Cloud provider.

One of the main focuses of WP1 has been to provide the validation of Use Cases. In whole, WP1 provides requirements (gathered and analyzed in T1.1), which have been considered in the research and development work of other WPs. The solutions developed in the other WPs were continuously monitored (one activity of T1.2), deployed (T1.3) and then validated (T1.4).

1. Use Case 1: OpenStack Framework

IBM has driven UC1 addressing the OpenStack Framework, with systems for key management addressing data-at-rest encryption. The scenario of this use case relates to a Cloud-storage platform, which supports server-side encryption with flexible key-management solutions. As such, this use case is relevant for internal (private) Cloud solutions as well as for public Cloud providers using the open-source OpenStack framework. The main focus is on OpenStack Swift, an object-storage system that runs on commodity hardware and provides failure resilience, scalability, and high throughput in software. Encryption occurs at the server side under the governance of the storage provider; encryption inside the storage platform is an important feature for large-scale and enterprise-class storage systems. Coupled with a suitable key-management solution that is able to control and securely erase cryptographic keys, the encryption technology should also support the controlled destruction of data, which is called secure data deletion here. Data-at-rest encryption and secure deletion are important requirements for enterprise-level Cloud storage services.

The details of the Use Case scenario and the solutions have been presented in earlier deliverables of ESCUDO-CLOUD, specifically in D1.1–D1.3 and D2.6. Figure 1.1 recalls the overview of UC 1 trust boundaries, with only the data owner being the trusted entity and the data being encrypted at rest on a single OpenStack based CSP.

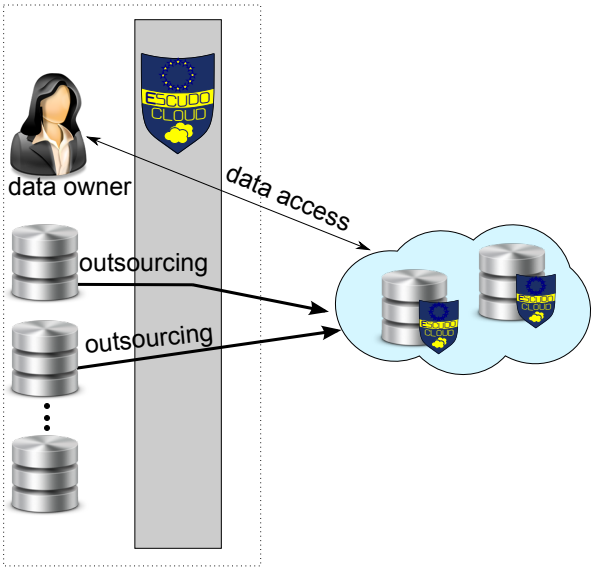

Figure 1.1: Trust assumptions for UC1

Use Case 1 is highly relevant for Cloud operators using OpenStack. In particular, a wide subset of IBM's Cloud offerings and products make use of the OpenStack framework that did not yet have all required enterprise-grade features. Applying this technology and extending the OpenStack framework with cryptographic protection has been an important step forward toward offering enterprise-grade services in the Cloud market.

1.1 Validation of Objectives

1.1.1 Use Case 1 Objectives

The goal of this use case consists of adding cryptographic protection technology inside a Cloud platform, in particular, to storage systems. Clients of Cloud services and operators of the CSPs benefit from data encryption in the storage systems, so as to make the system resistant to attacks that target lower layers of the infrastructure.

More precisely, the operations to be supported in the Cloud storage solution are summarized in the following list. Such operations are executed between a client (or tenant) and a Cloud-storage infrastructure (cf. D1.1):

1. Client enables encryption on object storage,
2. Clients leverage an (off-premise) encryption-enabled object storage to store their data,
3. Client retrieves data on encrypted object storage,
4. Client configures policies for secure deletion of its data,
5. Client securely deletes data on encrypted object storage,
6. Client uses policies to securely delete data on encrypted object storage,
7. Client exploits the scalability of the key management service to serve a large number of keys for its (private) object storage.

The encryption and key-management functions in the OpenStack platform and specifically in OpenStack Swift did not support encryption and secure deletion features as of 2015. The objective of ESCUDO-CLOUD is to contribute such features through schemes for key management. In particular, secure deletion has been addressed by applying specific protection mechanisms based on attributes specifying when data should be deleted securely. Later on, keys relating to those attributes must be deleted. The mechanisms produced by ESCUDO-CLOUD should therefore support these goals in a flexible way.

The objectives for the key-management support in UC1 have been described in D1.1 and are summarized in Table 1.1.

The technologies relevant for addressing the needs of Use Case 1 originate from different efforts within ESCUDO-CLOUD. From WP2, T2.1 (Protection of data at rest), and T2.2 (Key-management solutions), the main results consist of technologies to protect the confidentiality and the authenticity of the stored data. In particular, secure deletion of data is supported and has been demonstrated in the OpenStack Cloud storage platform. The demonstration in context of the use case illustrates one of the main features offered by the key management approach of ESCUDO-CLOUD.

From WP3 (Information sharing in the Cloud), T3.2 (Secure multi-user interactions and sharing), the technology to guarantee integrity and consistency of data accessed by multiple users has been developed and demonstrated through OpenStack Cloud storage. This novel technology protects data that is shared and concurrently accessed by multiple clients from being altered or modified. Showing this enhancement in the OpenStack Cloud storage platform represents a key step to promoting and exploiting this result. This integrity and consistency protection system addresses needs that are beyond the formally captured requirements of Use Case 1 and has not been captured by Table 1.1.

Req. Ref.	Description
REQ-UC1-IKM-1	The system supports CRUD operations for cryptographic keys and related cryptographic material, which is used in the Cloud infrastructure.
REQ-UC1-IKM-2	Deployment and management of infrastructure keys is driven by policies and automated, to the extent possible.
REQ-UC1-IKM-3	The Cloud infrastructure-key management system supports the relevant open standards that are used by the industry (OASIS KMIP for REST APIs and possibly OASIS PKCS #11 for interfaces to secure hardware modules).
REQ-UC1-IKM-4	The Cloud infrastructure-key management system supports the secure deletion of cryptographic material.
REQ-UC1-TKM-1	The system supports CRUD operations for cryptographic keys and related cryptographic material, which is used by a tenant.
REQ-UC1-TKM-2	Deployment and management of tenant keys is driven by policies and automated, to the extent possible.
REQ-UC1-TKM-3	The Cloud tenant-key management system supports the relevant open standards that are used by the industry (OASIS KMIP for REST APIs and possibly OASIS PKCS #11 for interfaces to secure hardware modules).
REQ-UC1-TKM-4	The Cloud tenant-key management system supports the secure deletion of cryptographic material.
REQ-UC1-SKM-1	Key-management systems can be operated in a redundant and fault-tolerant way and do not introduce any single point of failure.
REQ-UC1-SKM-2	Key-management systems do not limit the scalability of the Cloud platform. They must either be offered with large enough throughput or they must be scalable on their own.
REQ-UC1-SKM-3	Key-management systems can cope with the weak forms of consistency found in Cloud platforms such as OpenStack. In particular, the key-management systems provide support for eventual consistency of the underlying operations in the Cloud platform.

Table 1.1: UC1 requirements (from D1.1)

The work and resulting solutions demonstrated in UC1 aim at the business goals of protecting against insider attacks, ensuring compliance with legal regulations, and increasing the value of the storage service in response to customer demand. This directly supports the main approach of ESCUDO-CLOUD.

1.1.2 Status of Implementation

KMS for storage encryption. As explained in more detail in D1.3, the at-rest encryption in Swift encrypts all object data, object etags, and any user metadata values set on objects. The feature is enabled by a cluster operator and completely transparent to the end-user. Internally-encrypted data is never returned to clients via the API. The encryption feature in Swift has been included[1] in the main code distribution of OpenStack Swift. The feature has been available in OpenStack Swift since version 2.9.0 and in OpenStack since the Newton release; it was developed with the help of IBM and of IBM Research – Zurich. The encryption layer is a prerequisite for the advanced key management features described next.

The key management tools for server-side encryption of data at rest are described in D2.6 (Sections 3.2–3.3). They enhance the initial encryption feature in OpenStack Swift by using more elaborate key management techniques, and empower the clients by allowing them to manage the keys used to encrypt their data.

Clients storing data in the Cloud or the Cloud provider itself adds an encryption layer within the storage system. To obtain the keys for encrypting and decrypting the data, the storage system talks to a key management system (KMS). It may be the case that the client is able to manage the keys directly, e.g., through creating new keys or by deleting keys that should no longer be accessed in the KMS. The KMS may be hosted on-premise with the Cloud provider or even at the client, connected to the Cloud over the Internet.

The native key management of the data-at-rest encryption in OpenStack Swift (available with version 2.9.0) is limited, in the sense that the root secret (or master encryption key) is stored in the configuration file of a Cloud-storage cluster node. This key cannot be changed without losing all the data encrypted using that key as the root secret. The encryption keys for the accounts, containers, and objects are deterministically derived from the root secret.

The key-management tools for server-side encryption of data at rest (D1.5) employ a *hierarchy of keys*, with the top level key(s) stored in an external key management system; for Swift object storage, the external service is a Barbican server. These keys can in turn be stored by the OpenStack Barbican server in hardware security modules (HSMs) to further secure the system. The hierarchy level of the top-level keys, as well as the entity or entities managing those keys, impact the scenarios and use cases that the Cloud storage system can support. The key hierarchy is created to match the Swift storage hierarchy of accounts, containers, and objects. Additionally, the key hierarchy can have a master key above the account key, which in the initial Swift encryption implementation corresponds to the root secret.

The key-management tools can be used in an OpenStack Cloud-storage deployment in two modes: with *server-managed keys* and with *client-managed keys*. The key management tools also provide *key rotation*. By periodically rotating the keys in the key hierarchy, two goals can be achieved: 1) the time window during which a compromised key can be misused by an adversary is limited, and 2) the granularity of data that can be securely deleted can be adjusted.

[1] `https://www.swiftstack.com/blog/2016/07/12/at-rest-encryption-in-openstack-swift/`

The advanced key management for OpenStack Swift has been made available as open-source[2] and is currently pending integration with the "master" branch of Swift.

KMS for OpenStack. *Barbican*[3] is the OpenStack key-management service. It provides secure storage, provisioning and management of secret data. This includes keying material such as symmetric keys, asymmetric keys, certificates and raw binary data. Barbican uses a REST API and can serve keys to all components of an OpenStack Cloud installation.

Barbican is developed by the OpenStack Project but has been adapted to serve the needs of ESCUDO-CLOUD's UC1 in the OpenStack framework hosted on the IBM Cloud. In particular, based on the work in ESCUDO-CLOUD, IBM has contributed to the design of the *IBM Key Protect*[4] service, which went live in 2016[5] as part of the IBM Bluemix public Cloud offering.

IBM Key Protect is a multi-tenant key-management service for the Cloud that gives the customers ownership of the encryption process in their environment. Key Protect is based on OpenStack Barbican, which provides a REST API for secure storage, provisioning, and management of cryptographic keys, X.509 certificates, and passwords. IBM Key Protect enables customers to encrypt sensitive data at rest and easily create and manage the entire lifecycle of cryptographic keys that are used to encrypt data.

Furthermore, it is possible to extend the key-management service with Cloud-based hardware security modules (HSM). User keys can be encrypted (or wrapped) with an encryption key available only within an HSM. In this way, the secret key of a user never leaves the HSM in clear, while the encrypted version of this secret key is retained outside the HSM by the user and is an index to the actual secret key in the HSM. When a user needs to perform a cryptographic operation using this key, the encrypted version of the key is used to refer to the actual secret key stored in the HSM where all the cryptographic operations are performed. IBM Key Protect provides an API to programmatically integrate their applications with an HSM.

The KMS for storage encryption in OpenStack Swift has been enabled to work with OpenStack Barbican and IBM Key Protect.

1.2 Technical Evaluation

1.2.1 Functional Requirements

In Table 1.2, the functions provided by the KMS for storage encryption (*Swift-KMS*) and the KMS for OpenStack (*OpenStack-KMS*), as described in the previous section, are compared to the requirements of ESCUDO-CLOUD that they satisfy.

For explanation of the provided functions, we refer to the availability of OpenStack-KMS in the *IBM Bluemix Public* Cloud service and the prototypes of Swift-KMS. *Infrastructure-level* key-management operations are in the realm of OpenStack-KMS. *Tenant-specific* key-management functions may be provided by the OpenStack-KMS and by Swift-KMS.

The requirements relating to scalability and weak consistency (**REQ-UC1-SKM-1**, **REQ-UC1-SKM-2** and **REQ-UC1-SKM-3**) are addressed by the designs, proof-of-concept implementation, and the prototype with a production system of *Scalable Distributed Key Management for*

[2] `https://github.com/ibm-research/swift-keyrotate`
[3] `https://docs.openstack.org/barbican/latest/`
[4] `https://console.bluemix.net/catalog/services/key-protect`
[5] `https://www.ibm.com/blogs/bluemix/2016/12/dallas-key-protect-ga/`

Cloud Storage, as described in D2.3 (Chapter 4).

Requirement Reference	Requirement Description	Covered by Component	Implementation Status
REQ-UC1-IKM-1	CRUD operations for infrastructure keys	OpenStack-KMS	Production
REQ-UC1-IKM-2	Policy-driven and automated infrastructure-key management	OpenStack-KMS	Production
REQ-UC1-IKM-3	Support for standard APIs and protocols in infrastructure-key management	OpenStack-KMS	Production
REQ-UC1-IKM-4	Support for secure deletion of cryptographic material	Swift-KMS and OpenStack-KMS	Roadmap
REQ-UC1-TKM-1	CRUD operations for tenant keys	Swift-KMS and OpenStack-KMS	Production
REQ-UC1-TKM-2	Policy-driven and automated tenant-key management	Swift-KMS and OpenStack-KMS	Production
REQ-UC1-TKM-3	Support for standard APIs and protocols in tenant-key management	Swift-KMS and OpenStack-KMS	Production
REQ-UC1-TKM-4	Support for secure deletion of cryptographic material	Swift-KMS and OpenStack-KMS	Roadmap
REQ-UC1-SKM-1	Redundancy and fault-tolerance in key-management systems	OpenStack-KMS	Prototype
REQ-UC1-SKM-2	Scalable design of key-management system	OpenStack-KMS	Prototype
REQ-UC1-SKM-3	Key-management solutions support weakly consistent operations in Cloud platform	OpenStack-KMS	Prototype

Table 1.2: Requirements for Use Case 1 and evaluation. (*Production* means that the requirement has been met in production systems; *roadmap* means that integration with production systems is planned; and *prototype* signifies that a prototype of the technology is currently available.)

1.2.2 Non-functional Requirements

The non-functional requirements discussed in D1.1 (Section 3.4.7) have been met by all components of the solution. In particular, the OpenStack-KMS supports the relevant standards for key management systems and cryptographic algorithms, such as PKCS #11.

1.3 Further Development

The scalable distributed key management for Cloud storage, as described in D2.3 (Chapter 4), has been developed only as a prototype and is not integrated into the OpenStack Cloud. This will be the subject of future work.

Despite the success of integrating some of the technology developed by ESCUDO-CLOUD with the open-source distribution of OpenStack, the adoption of novel security technology generally remains slow. The experience of working on an open-source production-grade system has demonstrated that the most important step is to establish mutual trust between the contributors and the project maintainers. Only then will novel features and code be accepted into a well-run existing project.

Often the largest inhibitor to deploying a security function (such as support for key rotation and secure deletion) lies in missing awareness for the problem. Advanced encryption, key-management, and secure deletion features for a Cloud platform do not primarily address usability but security itself. However, many operators and clients of Cloud services do not want to add security functions today because they are not aware of their relevance and because of the extra cost. As observed many times in the market, security solutions are only built and deployed when the end-users are aware of the problems they solve. Often the market is not ready for adoption until attacks have demonstrated the dangers of insecure systems or when regulation requires secure solutions.

Adoption of the key-management techniques provided by ESCUDO-CLOUD for Use Case 1 in the context of OpenStack will progress according to those needs. Currently there is a strong push towards integrating the secure-deletion feature for storage with in IBM's public Cloud offering.

1.4 Summary

Use Case 1 covers data-at-rest protection in public and private Clouds. It encompasses both the protection of data by means of encryption within the Cloud infrastructure and the management of the keys used in the encryption. Key management must be scalable, supporting a multi-tenant environment with multiple containers per tenant, and supports advanced functionality such as secure deletion of data. ESCUDO-CLOUD implemented data-at-rest encryption in the OpenStack Swift open-source Cloud object storage system, and developed scalable key-management techniques which have been implemented in OpenStack Swift and Barbican. Data-at-rest encryption is already included in the current OpenStack software distribution, whereas the advanced key-management functionality is scheduled for inclusion in future versions. In summary, ESCUDO-CLOUD has extended the OpenStack software distribution, which is widely used among Cloud providers, with an enterprise-grade data-at-rest protection solution.

2. Use Case 2: Secure Enterprise Data Management in the Cloud

SAP contributed and realized Use Case 2, represented by a supply chain scenario in which companies strive to realize efficiency gains by sharing information in a secure manner. Sensitive information is represented by airplane information from airlines and decision tree evaluation information from a maintenance, repair and overhaul (MRO) forecasting provider. The information is utilized for collaborative demand forecasting (CF) of spare-parts to efficiently plan services and thus realize cost savings by streamlining the maintenance process. The technological challenge of this use case was to develop a novel supply chain cooperation system based on encrypted database technology. For this we developed concepts that bring together central processing at a honest-but-curious Cloud Service Provider and client-side encryption for involved supply chain partners (i.e., airlines and MRO). A schematic representation of the Use Case is depicted in Figure 2.1.

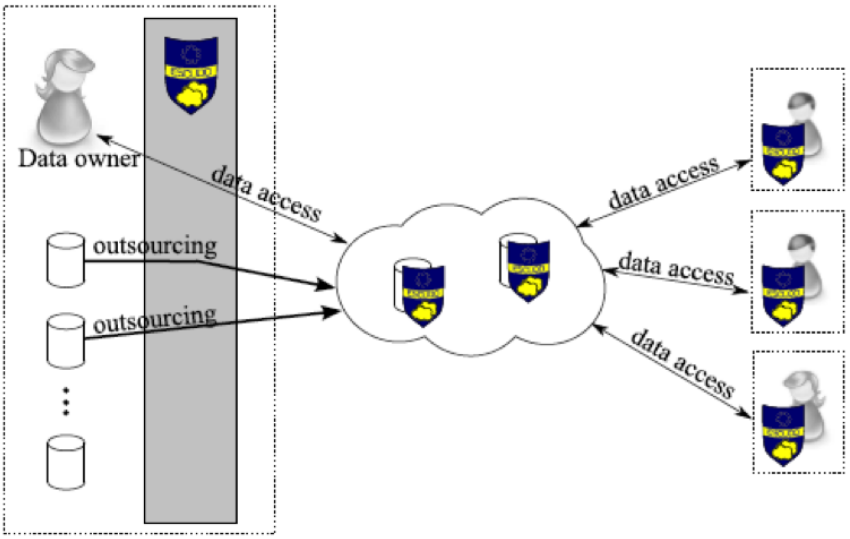

Figure 2.1: Overview of the UC2

Use Case 2 provides key solutions for enterprises who desire to upload and share data for collaborative processing in the Cloud. These technical solutions are combining industrial-grade databases such as SAP HANA and Order-Preserving Encryption to realize transparent encryption. In this area, the work on Use Case 2 advanced the state-of-the-art on private decision tree evaluation and multi-user encryption and key management for processing over client-side encrypted data.

2.1 Validation of Objectives

2.1.1 Use Case 2 Objectives

Use Case 2 addresses the technological challenge of developing new supply chain cooperation systems based on encrypted database technology in the Cloud. The overall objective of Use Case 2 is to support fine-grained access and enforce access control restrictions over outsourced and encrypted, yet searchable, data at the same time. A detailed description of the functional requirements of the Use Case are provided in Table 2.1.

The technology for Use Case 2 is based on search and aggregation operations over encrypted data. These operations can be applied in planning the MRO service to different customers without knowing the actual status of the aero fleet, nor the capacity usage and inventory status of the service provider. ESCUDO-CLOUD strives to realize supply chain collaborative forecasting in Use Case 2 under the following individual objectives.

1. Share supply chain data while overcoming the risk of loss of information advantage.
2. Preserve confidentiality of operations to prevent competitors from anticipating company's future plans.
3. Prevent competitors from developing new competitive products by utilizing supplier information.
4. Prevent Weakening of the bargaining power after disclosure of purchase or supply volume.

A supply chain coordination system based on encrypted database technology fulfills the need by maintenance and support actors for secure ICT solutions that foster collaboration. The technological cryptographic building blocks for Use Case 2 have been mainly fed by the results on selective sharing (T3.1) under WP3. A main result is the direct implementation of Oblivious Order-Preserving Encryption to increase Confidentiality in the case of decision tree evaluation through secure multi-party computation techniques such as garbled circuits. Integration into SAP HANA was achieved via the extension of SAP's encrypted HANA database prototype SEEED (Search over Encrypted Data) from single-user by providing encryption-based access control functionality in the face of multiple users and keys.

2.1.2 Status of Implementation

Within Use Case 2 there are multiple parties, Customers and Maintenance Provider (MRO), operating on a shared database hosted by Cloud Service Provider (CSP). To enable MRO to perform fine-grained relational queries an encryption based access control model has been integrated into the SAP HANA database based on the findings of D3.3.

Confidentiality and access control are achieved by the use of multiple encryption keys. Through this, selective data sharing between multiple users (i.e., entities as an airline business group represented by many individual Customers) in the supply chain scenario of UC2 is realizable. Due to the use of Order-Preserving Encryption, the CSP is able to perform SQL evaluations over encrypted access-restricted relations. By the use of proxy reencryption as a relational operation joins between relations that are encrypted with different keys are enabled. Integrity and Availability is provided by the SAP HANA database hosted at the Cloud Service Provider. By never releasing their keys Customers remain sovereigns of their data in Use Case 2. In the following we will outline the implementation building blocks for Use Case 2 as presented in Figure 2.2.

Req. Ref.	Description
REQ-UC2-AC1	Access control decisions should be based on the subject of a client, i.e. the subject granularity is a client.
REQ-UC2-AC2	It should be possible to group several users into a group and grant or revoke access for the entire group.
REQ-UC2-AC3	Access control decisions should be based on the object of a database cell as identified by a column (in a table) and a row owner.l
REQ-UC2-AC4	The access control model should be an access control matrix.
REQ-UC2-AC5	Access control rights should be grantable and revocable by the database administrator with support of the data owning clients.
REQ-UC2-AC6	Access control should be enforced by the client, i.e. cryptographically enforced.
REQ-UC2-KM1	Each client should have its own key generated and kept confidential at its site. From its master key other keys for access to finer-granular objects may be derived.
REQ-UC2-KM2	There should be keys for access by groups. These keys are shared by groups of clients to access common data. From these keys other keys for access to finer-granular objects may be derived.
REQ-UC2-KM3	Client keys should be stored securely, e.g. in a secure key store (PKCS #12) protected by a password. Additional mechanisms such as hardware security modules are initially optional.
REQ-UC2-KM4	Groups key may be derived using a public key hierarchy stored at the cloud service provider. This is a low priority feature.
REQ-UC2-EQ1	The database should support different encryption schemes for different database operations.
REQ-UC2-EQ2	The encryption should be adjustable to the database operations performed (onion encryption). Probabilistic encryption is more secure than deterministic encryption which is more secure than order-preserving encryption
REQ-UC2-EQ3	The database driver should at least support three different query evaluation techniques: query rewriting, proxy re-encryption and post-processing.
REQ-UC2-EQ4	All database operations should be supported across different client keys, i.e. spanning multiple access groups.

Table 2.1: Functional requirements for UC2 (from D1.1)

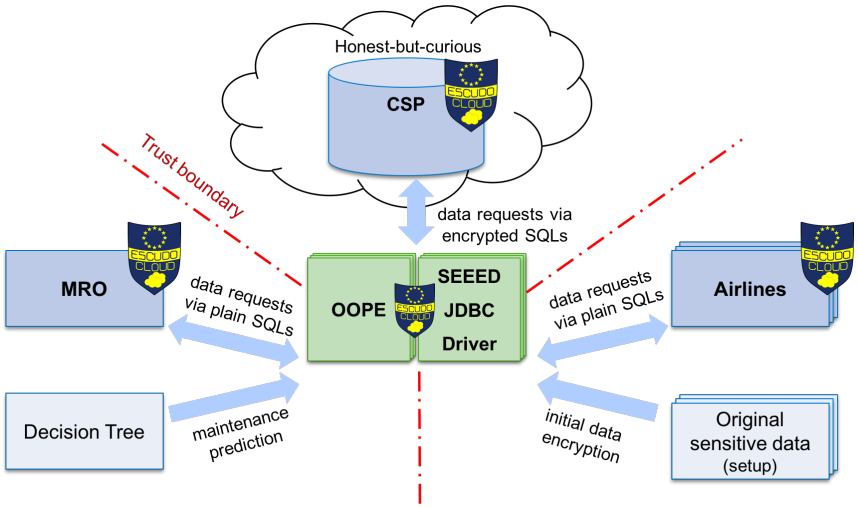

Figure 2.2: Overview of the UC2 components and trust boundaries

Customer Application. The Customer Application for Use Case 2 comprises the encryption of data before transfer to the SAP HANA database using the SEEED Driver instead of the standard JDBC driver. In the Use Case 2 demonstrator, there is only one database accessor component for all parties, but there would be an accessor for every role that needs a database connection in future scenarios that comprise other industrial actors.

MRO Application. The MRO provider, by accessing its view and triggering the generation and visual presentation of the decision tree by the CSP, can see the demand forecast with respect to a specific parameter (i.e., a leaf of the tree). Internally, the decision tree is represented by a JSON object containing the data of the decision tree: attribute names of the nodes, number of nodes, and executed operation. The aggregated probability of an upcoming overhaul is calculated by the paths from root to leaf of the decision tree. The MRO generates a SQL query that describes the tree and forwards it to the Customer. The Customer is therefore able to encrypt the SQL the same way the data in its cloud database is. In that way, after forwarding the needed SQL to the CSP, the CSP is able to use the encrypted SQL for a search on the encrypted data without awareness of the SQL request or the requested data itself.

Server Application. In contrast to the Customer Application, the Cloud Application handles data solely in an encrypted format. This ensures, that the CSP cannot read any plaintext from the customer tables within its database.

OOPE Module. We increase privacy, and thus protection against loss of information advantage, not only for the customers but also the MRO by having implemented Oblicious Order-Preserving Encryption (OOPE). OOPE keeps the criteria of the MRO's decision tree confidential towards Customers, while Customers still remain able to encrypt the tree for evaluation on their data.

2.2 Technical Evaluation

2.2.1 Functional Requirements

Deliverable D1.1 grouped the functional requirements of Use Case 2 into the three ESCUDO-CLOUD dimensions: *Access Control (AC)*, *Key Management (KM)* and *Encrypted Query Processing (EQ)*. AC requirements are covered via extensions of the SEEED *Multi-Tenancy* and SEEED *Key Management* components as well as the SEEED JDBC *Driver*. KM requirements are met by the newly introduced *OOPE Module* and extensions of the SEEED *Key Management* component and the SEEED *Driver*. EQ requirements are covered via extensions to SEEED *Onion Encryption* and the SEEED *Driver*. Table 2.2 provides the implementation status of each requirement and the component it is covered by.

Requirement Reference	Requirement Description	Covered by Component	Implementation Status
REQ-UC2-AC1	Access control per client	SEEED Multi-Tenancy	Production
REQ-UC2-AC2	Access control per group of clients	SEEED Key Management	Prototype
REQ-UC2-AC3	Access control per database cell	SEEED Key Management	Roadmap
REQ-UC2-AC4	Access control matrix model	SEEED Key Management	Production
REQ-UC2-AC5	Access grant and revoke by administrator	SEEED Key Management	Production
REQ-UC2-AC6	Access control enforced by client	SEEED Driver	Production
REQ-UC2-KM1	One key per client	SEEED Key Management	Production
REQ-UC2-KM2	Group key management	OOPE Module	Prototype
REQ-UC2-KM3	Client key securely stored at client only	SEEED Driver	Production
REQ-UC2-KM4	Group keys derivable	OOPE Module	Prototype
REQ-UC2-EQ1	Encryption schemes	SEEED Onion Encryption	Production
REQ-UC2-EQ2	Adjustable onion encryption	SEEED Onion Encryption	Production
REQ-UC2-EQ3	Proxy re-encryption, Query rewriting, Postprocessing	SEEED Driver	Production
REQ-UC2-EQ4	Support for different keys	SEEED Driver	Roadmap

Table 2.2: Functional requirements for Use Case 2 and their implementation status

2.2.2 Non-functional Requirements

Section 4.7.7 of D1.1 requires the performance of database queries under reasonable sharing requirements to remain acceptable from a user perspective. The prototype for UC2 fulfills this requirement.

2.3 Further Development

The tools for selective sharing for secure enterprise data management in the Cloud, as described in D3.5 (Chapter 1), have been developed as prototype using MySQL. While this protype shows that oblivious order-preserving encryption can be implemented with any database management system, it can be enhanced to support more than range queries. SAP created the SEEED library [GHH+14] to support search on encrypted data stored in a relational database. SEEED is based on the popular tool CryptDB [PRZB11] that uses different encryption techniques, such as order-preserving encryption, deterministic encryption, randomized and additively homomorphic encryption, to execute SQL queries on encrypted data. In future work, we envision to extend our OOPE prototype with the capability to handle search on encrypted data using SEEED.

Moreover, the security of OPE is not fully understood and current existing OPE schemes are subject to inference attacks. Therefore, another direction is to continue research on OPE in other to develop new more secure OPE schemes. However, there already exist randomized OPE schemes such as frequency-hiding OPE (FHOPE) scheme [Ker15], that are more resistant to inference attacks than deterministic OPE. In its current version, our prototype is implemented using deterministic OPE schemes [PLZ13, KS14].We are currently working on extending it to support FHOPE thus further enhancing security.

The cornerstone of OOPE is the comparison protocol, which is now implemented using garbled circuits technique [KS08, KSS09]. The performance of our scheme can be improved by implementing this comparison protocol with state-of-the-art garbling technique [BHKR13]. Alternatively, we can also test our scheme with the comparison protocols by [DGK07], which can be implemented using elliptic curve cryptography [WFNL16]. Finally, OOPE requires some additively homomorphic operations, which are implemented in the current version of the ESCUDO-CLOUD prototype using Paillier encryption scheme [Pai99]. These homomorphic operations can also be realized with other encryption scheme, such as exponential Elgalmal, that can be implemented using elliptic curve cryptography resulting in smaller ciphertexts and better performance.

2.4 Summary

UC2 addressed the challenges of supply chain collaboration in the Cloud where the objectives required to perform collaborative forecasting while preserving privacy of the involved parties (i.e., Customers, MRO, CSP). The SAP tool for Use Case 2 unites encrypted query processing and secure multi-party computation with the industrial grade SAP HANA database in the Cloud. The tool is represented by an extended version of the Search-over-encrypted-data (SEEED) database of SAP research. Technically, the functional requirements along the ESCUDO-CLOUD dimensions were addressed by enhancing the SEEED JDBE Driver, the SEEED Multi-Tenancy, Key Management and Onion Encryption components, and the introduction of a new Oblivious Order-Preserving Encryption component. SAP met the use case objectives and delivered productive ready solutions for most requirements.

3. Use Case 3: Federated Secure Cloud Storage

The core deliverable of the ESCUDO-CLOUD UC3 is to ensure the confidentiality and integrity of the data of the BT customers, which is outsourced for storage to multiple Cloud storage services. The primary challenge addressed by UC3 here is to offer a tightly integrated key management and policy-based access control capability to BT customers, and to do this through a Cloud service store. Each customer is able to apply key release rules and conditions, and these rules and conditions also take into account user access rights. This is illustrated in Figure 3.1, which also shows the overview of UC3 trust boundaries, with the data owner being the fully trusted entity and the data is encrypted on different types of storage media on multiple Cloud service providers.

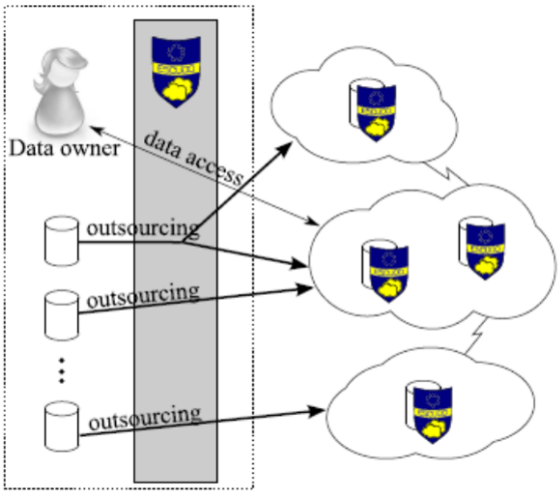

Figure 3.1: Overview of the UC3 trust boundaries

Therefore, ESCUDO-CLOUD UC3 provides the Data Protection as a Service (DPaaS) solution to its users, as it has to provide the customers with the means to store and retrieve encrypted data from different types of Cloud-based storage services. In the context of ESCUDO-CLOUD, three types of storage services are addressed; block storage, object storage and Big Data (HDSF) storage services. The DPaaS solution also addresses aspects of controlling access to the outsourced data through policy-based security rules that enforce the release of encryption keys.

3.1 Validation of Objectives

In this chapter, we showcase the relevance of the work done in the design and development of the DPaaS solution towards the ESCUDO-CLOUD UC3, and the contribution of the tools, techniques and components of the DPaaS solution towards the objectives and functional requirements of UC3.

3.1.1 UC3 Objectives

The four objectives of UC3 have been described in detail in D1.1 [SDR15]. They have been summarized in the following paragraphs and their realization has been discussed there as well.

Objective 1 of UC3 is concerned with the application of data protection in multi-Cloud environments. This objective has been realized by designing a solution that offers DPaaS via a Cloud service store (BT Service Store) that enables its users to manage the security life-cycle of their data stored on multiple Cloud platforms. The design and features of this component have been discussed in detail in D1.5 [BFT17]. As such, the BT Service Store component is particularly further applicable for use by Managed Security Service Providers (MSSP) and for Cloud customers that want to control and manage the protection of data-at-rest across multiple Cloud environments.

Objective 2 of UC3 is concerned with only allowing the customers (or trusted third parties of their choice) to manage the encryption keys in accordance with policy-based access control rules. This objective is addressed by the BT DPaaS solution by applying uniform and tightly-coupled data protection and data access policies for data stored in a multiplicity of Cloud providers (both private and public Cloud platforms).The data is protected by leveraging a Cloud-based data protection service for encrypting data independently and transparently of the Cloud provider hosting it, and ensuring that Cloud service providers have no access to the encryption keys or their protection and access control policies. As described in detail in D1.5 [BFT17], the Access Control Service and the Key Management Service components of the DPaaS are constructed and offered as tightly integrated services that manage the protection of the sensitive outsourced data. This tight integration ensures that the decryption of the protected data is only possible in the client's environment following a policy-based approval procedure and the resulting release of the encryption key.

Objective 3 of UC3 is concerned with allowing the customers to choose and use different Cloud storage services and provide an integrated view of these multiple Cloud storage services to the customers. This objective is realized by the DPaaS solution by utilizing a combination of the BT Service Store component and BT Data Encryption Agents. The BT Service Store is used to offer customers centralized access and management interfaces to different Cloud storage services like block storage, object stores and Big Data clusters that can be hosted on or provided by different Cloud service providers. To cater for the diverse architectural and operational realities of these disparate storage services, the data encryption agents are highly customized. In case of block storage services they fulfil their roles as software agents installed inside individual virtual machines, in case of object stores they fulfil their functions as part of a proxy gateway through which the data objects transit, and in case of Big Data storage clusters they are designed and implemented as HDFS plug-ins that can understand the Hadoop filesystem virtualisation and encrypt and decrypt data on the HDFS DataNodes accordingly.

Lastly, objective 4 of UC3 is concerned with ensuring that the data is secured in accordance to specific data protection regulations and standards e.g., FIPS etc. This objective is realized by the DPaaS solution by only enabling the data encryption agents, encryption gateway and HDFS plug-ins of the DPaaS solution to perform the core encryption and decryption operations on the customer data. The encryption and decryption process is transparent to the applications and end-users while the data-at-rest will always stay in encrypted state on the multiple Cloud platforms. The Key and Access Management services are very granular and flexible, in order to be customized for compliance with a large number of data security standards and regulations. Furthermore, the specific encryption library used by the BT Data Encryption Agents to perform these low-level tasks is FIPS-140-2 [NIS01] compliant.

3.1.2 Status of Implementation

The technologies relevant for addressing the needs of UC3 were identified and realized mainly in T4.3, and have resulted in the forming a prototype solution, referred to as Data Protection as a Service (DPaaS). The DPaaS itself is composed of four main components: the Service Orchestrator, the Key Management Service, the Access Control Service and the Data Encryption Agents. The design and implementation of the DPaaS solution and its components are given in detail in D1.5 (Use Case Prototypes) [BFT17]. A proof-of-concept prototype of the DPaaS solution is currently deployed in BT Research Cloud Platform (an internal private Cloud environment), that can communicate and interact with both private and public commercially available Cloud storage services, like Amazon EC2 and Citrix CloudStack (block storage), Amazon S3 and Caringo Swarm (object storage), and a Hortonworks based Big Data cluster.

3.2 Technical Evaluation

3.2.1 Functional Requirements

The process of identifying and gathering requirements for ESCUDO-CLOUD UC3 was carried out under T1.1 of WP1. The complete set of the functional requirements for UC3 are available in detail in D1.1 [SDR15] and D1.2 [BK15]. Table 3.1 shows the status of these requirements and the components that fulfil these requirements in UC3.

Requirement Reference	Requirement Description	Covered by Component	Implementation Status
REQ-UC3-KM-1	Each tenant[1] should be provisioned with an instance of a key management service from the Cloud service store.	Data Protection Manager (Key Management Service)	Production
REQ-UC3-KM-2	The tenants should be able to generate, modify and remove keys from their key management service instance.	Data Protection Manager (Key Management Service)	Production
REQ-UC3-KM-3	The key management service should be able to offer different key types and algorithms to each tenant, e.g., AES128, AES256, 3DES etc.	Data Protection Manager (Key Management Service)	Production

[1] A tenant is a group of users or a customer organisation.

REQ-UC3-KM-4	Only the tenants should be able to create and manage the keys.	Data Protection Manager (Key Management Service)	Production
REQ-UC3-KM-5	The CSPs should have no access or visibility of the tenants' keys.	Service Orchestrator, Data Protection Manager (Key Management Service)	Production
REQ-UC3-KM-6	The tenants should be able to cache their keys on trusted virtual machines or gateways.	Service Orchestrator, Data Protection Manager (Key Management Service), Data Encryption Agents	Production
REQ-UC3-AC-1	Each tenant should be provisioned with an instance of an access control service from the Cloud service store.	Data Protection Manager (Access Control Service)	Production
REQ-UC3-AC-2	The tenants should be able to create, delete and modify access control policies from their access control service.	Data Protection Manager (Access Control Service)	Production
REQ-UC3-AC-3	The access control service should be able to offer use of different attributes for the construction of a security rule.	Data Protection Manager (Access Control Service)	Production
REQ-UC3-AC-4	Only the tenants should be able to create and manage their access control policies.	Data Protection Manager (Access Control Service)	Production
REQ-UC3-AC-5	The CSPs should have no access or visibility of the tenants' keys.	Service Orchestrator, Data Protection Manager (Access Control Service)	Production

REQ-UC3-AC-6	All data protection operations should be governed by access control policies by either approving or denying access to the required keys.	Data Protection Manager (Access Control Service, Key Management Service)	Production
REQ-UC3-AC-7	The access control service of tenants should be tightly coupled with their key management service.	Data Protection Manager (Access Control Service, Key Management Service)	Production
REQ-UC3-SO-1	Each tenant should be provisioned with a Cloud service store account.	Service Orchestrator	Prototype; Roadmap
REQ-UC3-SO-2	The service store should provide the tenants with access to the storage services of multiple CSPs.	Service Orchestrator	Prototype; Roadmap
REQ-UC3-SO-3	The service store should be able to offer block storage service to the tenants.	Service Orchestrator	Prototype; Roadmap
REQ-UC3-SO-4	The service store should be able to offer object storage service to the tenants.	Service Orchestrator	Prototype; Roadmap
REQ-UC3-SO-5	The service store should be able to offer Big Data storage service (HDFS) to the tenants.	Service Orchestrator	Proof-of-Concept; Roadmap
REQ-UC3-SO-6	The tenants should be able to enable or disable the use of data protection service on the storage service of their choice.	Service Orchestrator	Prototype; Roadmap

REQ-UC3-SO-7	The service store should be able to offer key management as a service to the tenants.	Service Orchestrator	Prototype; Roadmap
REQ-UC3-SO-8	The service store should be able to offer access control as a service to the tenants.	Service Orchestrator	Prototype; Roadmap
REQ-UC3-DE-1	The core encryption process should only be controlled and managed by the tenant.	Service Orchestrator, Data Protection Manager (Access Control Service)	Production
REQ-UC3-DE-2	The tenant should be able to deploy and manage the core encryption process on trusted virtual machines or gateways as an agent or plug-in.	Service Orchestrator, Data Data Encryption Agents	Prototype
REQ-UC3-DE-3	The core encryption process should be FIPS 140 compliant.	Data Protection Manager, Data Encryption Agents	Production
REQ-UC3-DE-4	The encryption agent or plug-in should be able to securely access the tenant's key management service and access control service.	Data Protection Manager, Data Encryption Agents	Production
REQ-UC3-DE-5	The keys should only be released to the encryption agent or plug-in upon approval of an access control policy.	Data Protection Manager, Data Encryption Agents	Production

Table 3.1: Requirements for UC3 and Evaluation

3.2.2 Non-functional Requirements

The complete listing and details of the non-functional requirements for UC3 are available in detail in D1.1 (section 5.4.5) [SDR15]. Their summarized description and current status is given below.
Configuration Management: The deployment and configuration scripts should be exchanged be-

tween the service store and the VMs securely.
This requirement is met by the Service Orchestrator, which uses Puppet over SSL for this purpose.
Transparency: The encryption process should be transparent to the end user of the solution.
This requirement is met by the Data Encryption Agents, which encrypt data at the filesystem level to fulfil this requirement.
Concurrency: The DPaaS solution should be multi-tenant.
The Data Protection Manager and the BT Service Store are multi-tenant, however the Data Encryption Agents have to be single-tenant due to cryptographic principals.
Performance: The encryption and decryption operations on the protected data should carry low I/O performance overheads.
According to experiments carried out in EU FP7 Fed4FIRE project [PSDC15], using the same symmetric cryptography library as used in the DPaaS solution, the encryption process enacts 12% overhead for read operations and 30% overhead for write operations on block storage devices, as compared to the baseline of using no encryption at all.
Business Rules: Tenants are able to subscribe to the DPaaS via the BT Service Store. They are also able to enable or disable their use of the data protection service through the same interface. The installation of encryption agents and plug-ins is carried out automatically and the service store admin is able to integrate, modify and remove the provisioning of storage services of different Cloud providers for the tenants.
Compliance: The Data Protection Manager is FIPS 140-2 compliant and also complies with the PCI DSS 3.0 requirements 3, 7, 8, and 10.
Interoperability: The Key Management Service supports OASIS Key Management Interoperability Protocol (KMIP) version 1.3, Microsoft Extensible Key Management (EKM) and the Public-Key Cryptography Standard 11 (PKCS #11).
Maintainability: The Key Management Service supports key expiration and recovery operations.

3.3 Further Development

Data encryption capabilities are driving commercial opportunities for BT. BT is able to demonstrate encryption of volume and objects deployed in multiple clouds, and centrally managed. Tying access control to policy management enables external key management protocols to be integrated. Customers are able to interface with BT encryption capabilities using their own set of key hosting solution. BT is able to cover Cloud storage for volumes, objects, and big data by deploying customized agents interfacing with the central policy management console. This provides BT with commercial opportunities across different domains including Cloud computing, pharmaceutical, and retail.

3.3.1 Links into Projects or Exploitation

BT for Life Sciences R&D has been developed as a secure and segregated platform for scientists in pharmaceutical, biotech, devices & diagnostics companies as well as in academia and government. This platform enables research scientists to create global project groups and collaborate safely and securely via federated identity management and authentication tools on top of built-in secure sharing/social applications. This ecosystem allows the group to securely upload documents, share results and communicate via IM, voice, video or chat to analyse results in an environment that segments data and uses qualified hardware components and workflows specific to the pharmaceu-

tical industry. The BT for Life Sciences R&D platform is supported by the BT Assure portfolio of security services. These allow BT to assist customers in designing data encryption, anonymization, risk management and resilience to meet their quality and regulatory requirements for the cloud environment. The encryption capabilities of ESCUDO-CLOUD are integrated in this market facing portfolio. The platform builds on BT's On Demand Compute service supporting FIPS data encryption standards. This enables customers to conform to the quality and regulatory requirements, providing an environment suitable for many regulated applications used collaboratively amongst partners who may have differing risk profiles.

Object storage has gained in popularity recently as this solution offers high availability, fast access, easy sharing, or access to multiple devices. However, pushing private data to public places requires constant attention to the risk of exposing sensitive information. BT Compute Storage needed enterprise grade security - removing the risk associated with consumer-focused sharing services. ESCUDO-CLOUD has helped deliver the necessary technology and mechanisms (Cloud Encryption Gateway) to achieve this.

Big Data is driving much of the increased demand for data privacy. Efficient, and safe storage solutions are becoming a critical part of IT infrastructures, and that means having a mix of storage types. BT has extended its range of Big Data storage options by adding enterprise grade Hadoop as a Service (HaaS). The HDFS encryption work carried out in ESCUDO-CLOUD is being put on the integration roadmap as an additional capability. The policy based access control and data encryption can help customers implement complex requirements and help ensure compliance with EU or other existing regulations.

3.3.2 Outlook

The technology developed in ESCUDO-CLOUD UC3 can be used in three distinct contexts: collaborative projects, research extension and direct exploitation. In the future, our research department will continue to build on the technology developed and create new exploitation routes. The opportunities include confidential and privacy-preserving data analytics, where sensitive information needs to be processed, and in the domain of IoT, where there is huge potential to come up with use cases designed to protect privacy.

In the first context, we have been able to make direct re-use of the UC3 technology in the context of some collaborative programmes. BT is the programme coordinator in Trusted Cloud (`http://www.trustedcloudnews.eu/`). This project has improved governance on data encryption and even added capability of intelligent responses to privacy policies. Data encryption/decryption is governed by a combination of user credentials and context rules. This allows access to the data to be dependent on external factors such as time, location, or input from other devices or context. As a demonstration use case, Trusted Cloud is able to show access being denied in case of malware detection in a protected system and even eliminate the sessions of connected users in such cases.

In another collaborative programme called C3ISP (`http://c3isp.eu`), BT is capturing encryption events and security alerts to create a feed reporting to a SOC (Security Operation Centre). The SOC is able to analyse reporting patterns in order to create warnings to other customers. These alerts are used to benefit the community whilst keeping anonymity and confidentiality of the security alert reporting mechanism.

As described in the previous section, the encryption capabilities do exist in production in the BT Cloud Compute product. This is based on volume encryption. An implementation for Big

Data encryption is on-going and this will provide the same interface and experience for BT Cloud storage facilities. BT is investing in cyber security in a major way, including a graduate recruitment campaign in Cyber Security[2], the creation of a Cyber Security operations centre in Australia[3]), and the creation of a post-graduate programme with the collaboration of Airbus, Deloitte and Rolls-Royce at the University of de Montfort in Leicester[4], UK. This will create new oppertunities for researching exploitation routes or collaborative opportunities to further strengthen the BT data protection portfolio.

3.4 Summary

UC3 has built on the BT Cloud Compute data storage capability covering new scenarios more adapted to Cloud based storage. All UC3 objectives have been met with the implementation and integration of the various components of the DPaaS solution architecture. The solution includes not only the creation of policies combining user access rights and data encryption rules, but also key management capabilities. The results provided by ESCUDO-CLOUD have facilitated the introduction of data protection capabilities in the BT Cloud Compute platform, especially for Object and Big Data storage services. The work carried out in ESCUDO-CLOUD also acted as part of the BT's internal due diligence process required to turn the technology into exploitation.

[2] http://www.btplc.com/Careercentre/earlycareers/apprentices/ourprogrammes/cybersecurity/index.htm

[3] https://www.globalservices.bt.com/anze/en/news/bt-selects-sydney-for-global-cyber-security-hub-expansion

[4] http://www.dmu.ac.uk/about-dmu/news/2015/november/uk-cyber-skills-to-receive-breakthrough-boost-with-pioneering-new-training-course.aspx

4. Use Case 4: Elastic Cloud Service Provider

The Elastic Cloud Service Provider use case, UC4, jointly developed by WT and EMC, enables organizations to establish themselves as a Cloud broker, leveraging computational and storage infrastructures offered by third-party providers. This is achieved in a way that respects the confidentiality requirements of data owners by defining a strict trust boundary around the data owner and users authorized to access the data. The core mechanisms that enable this are the implementation of client-side encryption and secure third-party key management that ensure the protection of the data against even the Cloud providers hosting the data. Several novel security tools resulting from the technical Work Packages (WP2-4) were also evaluated for their application in UC4.

The UC4 architecture is also suitable for existing private (and public) Cloud providers that wish to offer such a data protection framework. Implementation of the architecture and security mechanisms developed in ESCUDO-CLOUD provide greater assurances and therefore increase the level of trust that data owners place in Cloud offerings as they remain in total control of the encryption and key management processes. The architecture does not only improve the security profile of the Cloud service, it also provides the ability to employ Cloud bursting techniques that are central to providing an elastic storage infrastructure.

Figure 4.1 illustrates the trust boundaries for UC4, with the data owner and authorized users identified as the only trusted entities. Data stored across the multiple Cloud providers is protected (i.e., encrypted) by the ESCUDO-CLOUD solution with only the trusted entities able to access the data unencrypted.

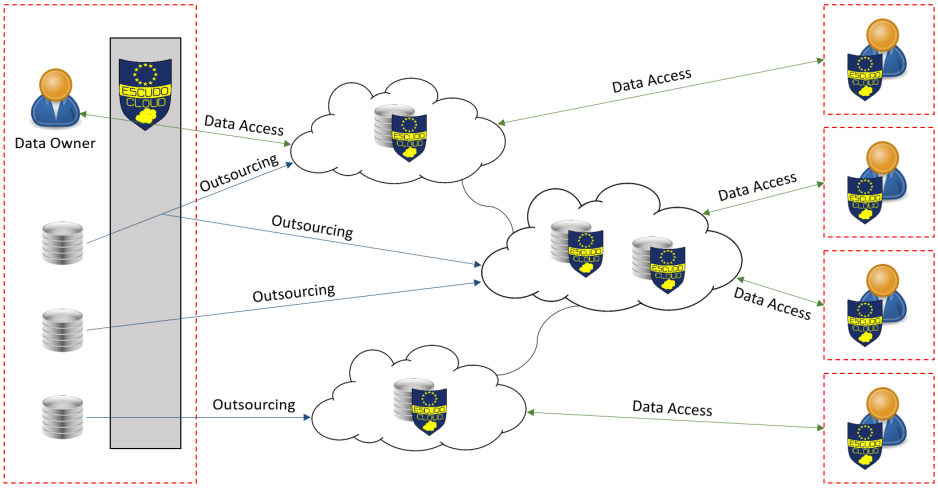

Figure 4.1: Overview of the UC4 trust boundaries

The following sections will describe the objectives of the use case, provide the implementation status of the architecture and validation its implementation against the objectives and requirements set out in D1.1 and D1.2.

4.1 Validation of Objectives

4.1.1 Use Case 4 Objectives

The approach taken in the UC4 solution will foster trust in an Elastic Cloud solution through the implementation of client-side encryption over user data. Due to the inherent risks to data (and key material) stored and processed in the Cloud, encryption is a vital mechanism to be included as part of an organization's security strategy. Trust in the Cloud requires accountability and transparency from the provider, enabling data owners to have visibility of how their data is managed. Through encryption and proper key management, data owners can retain their keys, protecting their data stored in Cloud. In this way, the data owner has the ultimate control over how their data sets are protected in the Cloud, how they are accessed and who may access it.

In the scenario described by this use case, a data owner has access to their files stored in the Cloud using an access point available on the client. The access may be done by using either a web portal accessible via a browser or a standalone executable installed on the client. A mobile application has also been developed to provide easy access from mobile devices. The access point will open a secure communication channel between the data owner and the CSP, allowing the data owner to manage their account and data stored in the Cloud. The storage backend of the service makes use of elastic Cloud storage technologies to enable the selection of a service configuration that meets the requirements of the data owner. A detailed description of the functional requirements of the use case are provided in Table 4.1.

Req. Ref.	Description
REQ-UC4-AC-1	Each user should have a unique username and password to enter in the system, granting them access their files.
REQ-UC4-AC-2	Authentication agent should remember user credentials after initial login.
REQ-UC4-AC-3	The middleware will control what data users can access and what operations the user can perform on that data.
REQ-UC4-AC-4	Only the data owner should be able to modify access control rules for other users.
REQ-UC4-AC-5	Only the administrator of the system (the ESCUDO-CLOUD middleware) should be able to enable the access of a locked user.
REQ-UC4-AC-6	Each user should have a limit of attempts to access the system.
REQ-UC4-SS-1	The Cloud (or storage service) should have capacity enough to store the data files of the users.
REQ-UC4-SS-2	The storage services should be accessible for the end users through ESCUDO-CLOUD middleware.
REQ-UC4-SS-3	By using an elastic Cloud should be possible to adapt the capacity of the Cloud to the user requirements.
REQ-UC4-SS-4	The storage service should comply with "EU Directive 95/46/EC - The Data Protection Directive"

REQ-UC4-SS-5	The CSP should implement appropriate technical and organizational measures to protect personal data against accidental or unlawful destruction or accidental loss, alteration, unauthorized disclosure or access, in particular where the processing involves the transmission of data over a network, and against all other unlawful forms of processing.
REQ-UC4-DE-1	The files should be stored encrypted. Only the user with the correct privileges will be able to decrypt the information.
REQ-UC4-DE-2	The data files should remain encrypted on the CSP until the user accesses them through a web browser.
REQ-UC4-DE-3	The data files should remain encrypted on the CSP until the user synchronizes the server with the agent installed on his device.
REQ-UC4-DE-4	An end user should be able to securely download a data file from the CSP through the web browser.

Table 4.1: UC4 Requirements

The core objectives of the architecture defined and implemented for UC4 is to offer certifiable secure data management in the Cloud by including the following main features:

- Encryption of the communication channel between the client and the CSP
- Encryption features for file storage in the Cloud
- Access management through definition of clear policies and roles
- Secure remote file access
- Distributed key management

4.1.2 Status of Implementation

Figure 4.2 shows the general architecture used for the implementation of the Elastic Secure Cloud Provider. There exist three stakeholders in this scenario, namely: the end-user, the agent and the CSP. The client-side encryption model enables ESCUDO-CLOUD to put the trust boundary as close as possible to the data owner (and authorized users). The communication between these actors is always the same: the user (data owner) will be able to manage her files hosted in the Cloud through the ESCUDO-CLOUD middleware. In this architecture the data is protected by encryption on the client side, and the data owner is the only entity authorized to access and manipulate the security policies enforced on the data. The CSP therefore has no access or influence over the encryption process, nor does it have access or influence over the security policies. Once the user is logged in via the ESCUDO-CLOUD middleware, this encryption is performed through the agent allowing users access to their data in a transparent way, without the necessity of installing additional software.

Implementing this elastic Cloud architecture, a service provider can offer file storage options including local and external storage resources to extend the storage capacity on the demand, and a secret storage container to secure user data. At this stage, the prototype is able to exploit the

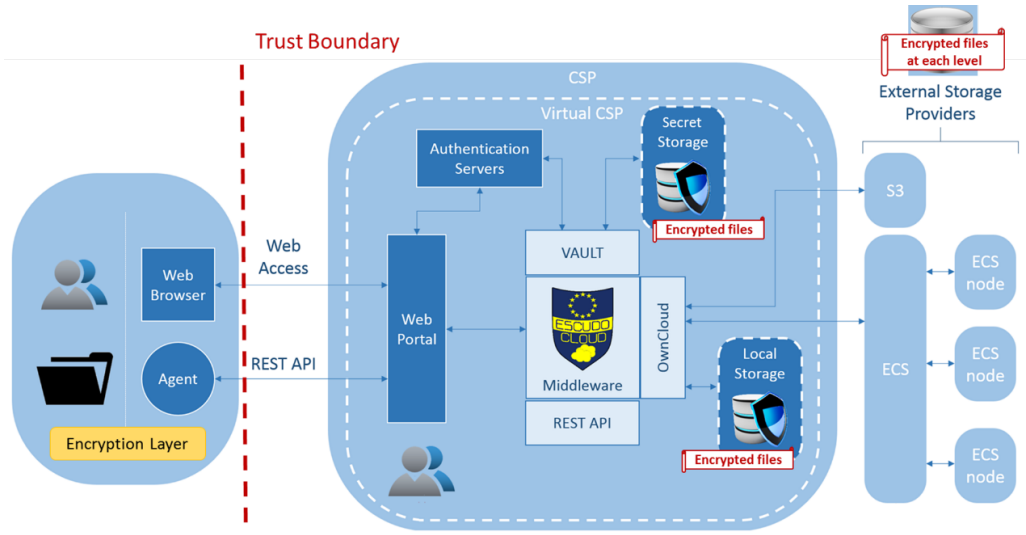

Figure 4.2: Elastic secure cloud storage architecture

infrastructure of different providers: Google Drive, Dropbox and EMC-ECS to achieve the elastic aspect. The technical objectives of the file storage are to store encrypted files, to provide an API to configure external file stores, and to provide an API to manage data backups for redundancy. The implementation of local file storage and the APIs facilitating the use of external file storage leverages an Owncloud server that is configured and the deployed within the UC4 reference architecture for ESCUDO-CLOUD. Briefly, Owncloud (https://owncloud.org/) is an open source, self-hosted file sync and share app platform. Several external storage providers have been integrated into the UC4 implementation. The technologies employed for its implementation are Owncloud v9.0.1 for the file storage and Postgres v9.5 for the management of database. Public providers such as AWS S3 were adopted in early prototypes, while EMC's Elastic Cloud Storage (ECS) platform provides a hybrid-Cloud solution that can be deployed in a private or public Cloud configuration (i.e., ECS can be configured to manage local file storage on existing infrastructure, and remote file storage at third-party datacenters).

As noted in the objectives of UC4 previously, key distribution is an essential component of a secure architecture. It is crucial that keys are protected from the CSP hosting the data in order to maintain the confidentiality guarantees on the user data from the CSP. This is achieved by storing the cryptographic keys and related key material in a secret storage container. The technologies used for its implementation mainly are Vault v0.5.2 (https://vaultproject.io) for the Secret storage manager and Consul v0.6.3 (https://www.consul.io/) for the Physical secret storage backend. Very shortly, Vault is a tool for securely accessing secrets. A secret is anything that you want to tightly control access to, such as API keys, passwords, certificates, and more and consul is a system for service discovery, monitoring, and configuration that is distributed and highly available. In UC4, WT uses Consul as a backend to Vault leading to get the best of both; Consul is used for durable storage of encrypted data at rest and provides coordination so that Vault can be highly available and fault tolerant.

4.2 Technical Evaluation

4.2.1 Functional Requirements

Table 4.2 maps the functional requirements defined in D1.2 (summarized in Figure 4.1) to the components identified in the architecture in Figure 4.2. D1.2 grouped the functional requirements of UC4 into three ESCUDO-CLOUD dimensions, *Access Control (AC)*, *Storage Service*, and *Date Encryption (DE)*. The implementation of features to address AC requirements are distributed across the architecture, providing multiple points where the access policies are enforced. Notably, it is the ESCUDO-CLOUD middleware that provides the authentication framework. This is supported by Vault for the protection of cryptographic keys, and Owncloud and ECS for access to the files. SS requirements are primarily handled by the Owncloud instance orchestrating local and external file storage. DE for UC4 is implemented on the client side but is supported by the ECS component that can provide an additional layer of encryption (compatible with the over encryption mechanisms reported in D2.2). The table represents the complete set of the relevant requirements for UC4, and also shows the status of the relevant component for each requirement.

Requirement Reference	**Requirement Description**	**Covered by Component**	**Implementation Status**
REQ-UC4-AC-1	Access Control to the web portal	Middleware (authentication framework)	Prototype
REQ-UC4-AC-2	Save credentials	Client; Vault;	Prototype
REQ-UC4-AC-3	Access control to middleware	Owncloud (through middleware)	Prototype
REQ-UC4-AC-4	Access to shared files	Owncloud; ECS as an external storage;	Prototype
REQ-UC4-AC-5	Access grant by administrator for locked users	Middleware; ECS;	Prototype
REQ-UC4-AC-6	Limit failed access attempts	Middleware	Prototype
REQ-UC4-SS-1	Manage Cloud storage usage/capacity	Owncloud; ECS;	Prototype
REQ-UC4-SS-2	Access control to storage	Middleware; Owncloud; ECS	Prototype
REQ-UC4-SS-3	Responsive elastic Cloud storage	Owncloud; ECS;	Prototype
REQ-UC4-SS-4	Comply with data protection directive	Middleware	Prototype
REQ-UC4-SS-5	Data recovery control	Owncloud (through middleware); ECS;	Prototype
REQ-UC4-DE-1	User encrypts/decrypts data control	Client; ECS;	Prototype

REQ-UC4-DE-2	Encrypted data cloud storage	Owncloud; ECS;	Prototype
REQ-UC4-DE-3	Ensure server synchronization	Middleware	Prototype
REQ-UC4-DE-4	Secure file download	Client	Prototype

Table 4.2: Requirements for UC4 and Evaluation

Navigation action	Response Time
Login	74 ms
Create a folder	118 ms
Open a folder	91 ms
Delete a folder	91 ms
Delete a device	91 ms

Table 4.3: Samples of response times for UC4 web navigation actions

During this project, the implementation has been carried out on the basis of the functional and security requirements defined at the beginning of the project as reported in D1.1 and D1.2. In the context of UC4, the research results from the topic areas studied under WP2 and security metrics (T4.1) have been used to build the first prototype for this use case meeting those requirements and achieving above objectives.

UC4 security is mainly characterized by authentication for the user session management, client-side encryption and Cloud storage including file storage and secret storage and a key management service. The implementation of the security features are detailed in D1.5. For the final prototype, file access can be achieved through a web portal, an agent, or a mobile application. WT targets the two current most popular OS (Android and iOS) in order to develop the mobile application of the UC4 prototype. The mobile application implementation has been tested against Android OS (version 7.1), and a solution has been designed for iOS 10.3. However, it is worth noting that WT has performed the implementation with a view to providing a multi-platform solution by using react active, and this brings sustainability to the ESCUDO-CLOUD solution.

4.2.2 Non-functional Requirements

Two main types of non-functional requirements were defined in D1.2: the performance requirements to obtain a responsive system; and the compliance requirements related to security aspects.

Regarding the authentication process and enumeration of files and/or folders, it is a question of web navigation activity. UC4 prototype presents an average response time of 93ms for web navigation actions. According to the KPI's related to web navigation, it is estimated a responsive time of 200ms, or less than 500ms, is acceptable and it is generally the threshold used to assess the performance. The UC4 prototype has been tested and the response time satisfies this threshold as shown in Table 4.3 giving samples of response times for the main UC4 web navigation actions.

Regarding uploading (i.e., encryption) and downloading (i.e., decryption), it is important to take into account the processing time. The response time is the sum of the latency and the processing time. Encryption and decryption are two time consuming processes that introduce additional latency. The performance of uploads (incorporating encryption) for a set of applications

have been tested [HAK17], including spiderOak and Viivo: solutions that were taken into account in our market analysis (see D5.3). In this article, the response times of spiderOak and Viivo are examined with different file sizes of 1 MB, 3MB and 5MB. The first results of the ESCUDO-CLOUD UC4 prototype measure favorably in comparison as shown in the below tables giving samples of uploading (encryption)/ downloading (decryption) response times.

Product	File size	Action	Response Time
Viivo	1MB	Encrypt	30.27 sec
Viivo	3MB	Encrypt	44.13 sec
Viivo	5MB	Encrypt	71.53 sec

Table 4.4: Viivo response times

Product	File size	Action	Response Time
SpiderOak	1MB	Encrypt	34.05 sec
SpiderOak	3MB	Encrypt	48.67 sec
SpiderOak	5MB	Encrypt	77.45 sec

Table 4.5: SpiderOak response times

Prototype	File size	Action	Response Time
UC4	1,7MB	Encrypt	8.39 sec (from submit to success notification)
UC4	3,66MB	Decrypt	17.31 sec
UC4	10,09MB	Decrypt	60.47 sec

Table 4.6: UC4 response times

For the security requirements that aim to provide a secure system, there are four specific objectives:

- The encryption of files on a Cloud provider's storage should be implement strong cryptography according to industry recommendations. Similarly, the protection of the Data Encryption Keys (DEK) used to protect user data by ESCUDO-CLOUD should follow industry recommendations for encryption. The UC4 architecture implements client-side encryption to encrypt/decrypt files, and to generate keys to encrypt/decrypt (technically discussed in the D2.1). UC4 uses cryptographic techniques for data confidentiality as addressed in WP2. The technologies in UC4 are PBKDF2-SHA512 for the generation of a 512-bit key, AES-256-GCM, Vault with Consul backend for the storage of encrypted keys, and the Crypto JS and SJCL Stanford Javascript crypto Library [1].

- An efficient protocol for the management of user keys and file/disk key should be in place. The UC4 prototype generates user keys on the user device. They are encrypted and then sent

[1] http://bitwiseshiftleft.github.io/sjcl/

to the server to be stored in a secret storage container. Key management mainly encompasses the generation of a derived key and an asymmetric key pair for each user. When a user logs into the client, a new valid key to decrypt the private key is generated from the user password and the stored metadata. A DEK is used to encrypt and decrypt the file contents and when a user shares a file, the client re-encrypts DEK with target user's public key. TLS is used to secure all communications between the client and the server of the application.

- A secure process to manage unexpected termination of the transaction by the user. An unexpected termination of transaction could be the result of a session timeout or a loss of connectivity. WT has developed a solution for the UC4 prototype enabling automatic reconfiguration of keys when the service has been interrupted in order to reduce the security vulnerability generated by an unexpected interruption.
- The user should have the possibility to ask for secure erasure of data. The implementation meets this requirement and features a *device wiping* management solution. The user is able to delete one of their own devices used to access this application in case it is stolen or otherwise compromised. This in effect blocks access to the data through that device.

4.3 Further Development

As the previous sections have elaborated on, the UC4 architecture has achieved its objectives in providing an elastic Cloud storage service that emphasizes the critical importance of data protection mechanisms to protect the confidentiality of user data. This was done in a way such that the risk of exposure of data to the host storing the data is mitigated. On top of the implementation of essential base components for such a service, WT and EMC have explored the potential for several innovations from the technical work packages of ESCUDO-CLOUD (namely, WP2, WP3 and WP4).

In particular, EMC focused attention on the Shuffle Index research initiated in D2.1. This work led to an implementation of the Shuffle Index on the ECS component of the UC4 architecture, presented in D4.3. The prototype proved the potential for the Shuffle Index to provide access pattern confidentiality in a multi-provider environment. Having proved the technology, the challenge now exists to identify the applications where it can have the greatest impact in providing enhanced confidentiality. While the Shuffle Index demonstrated that it performed well compared to alternative approaches with the same objective of protecting access pattern confidentiality, further analysis and development of the mechanism will be required for a practical implementation on large data sets. Currently the overhead of additional data transfers, though competitive with alternatives, is a barrier to adoption where the cost of such transfer outweighs the need for such high security.

Another promising aspect of WP2, is the development of the over-encryption mechanism in D2.2 (further detailed and implemented in D2.5 and D2.6). This mechanism provides both a significant performance improvement for the revocation of user access to data, and as a consequence, greater data protection in the event of a single layer of encryption becoming compromised (e.g, through exposure of cryptographic key). Given the existing independent encryption capability of ECS, the mechanism can be readily deployed such that client-side encryption in conjunction with ECS encryption would provide the over-encryption required to reduce the overhead of re-encrypting data as users access rights are revoked.

In addition, outcomes from WP3 have not been identified as critical requirements for UC4, however they do provide useful information and guidance in further enhancing the security offer-

ings of the use case. One aspect of UC4 is the capability to allow multiple users to access the data stored securely in the Cloud. This is best achieved through the use of efficient key management strategies. WT has developed an API to provide remote device wiping and foresees the implementation of features that would enable the granting and revocation of access rights. D3.1 proposes solutions for selective encryption to manage access control to different sets of data. This approach uses different keys to encrypt different tuples and then selectively distributes the keys to authorized users according to the policies defined by the data owner.

Another area of interest for UC4, but not a critical requirement, is that of application security. D3.2 provides comprehensive guidance on tools and recommendations for ensuring that the applications and virtual resources such as containers are verified and running in a known, trusted state. These approaches are applicable to UC4 and would provide an additional layer of security leading to generate trust in components of the infrastructure.

Another promising aspect of WP4, is the Service Level Agreement study describing in D4.1; the main interesting outcome is the work providing a SLA hierarchy focusing on security features (secSLA). The implementation a new back-end component based on secSLA and processing those data to improve the UC4 prototype by enabling the automatic selection of the third party providers. This will directly benefit a CSP and in particular a Cloud broker.

4.4 Summary

UC4 contributes to the key dimensions of the project, i.e. Cloud architecture, security, sharing and access. In the context of the development of the UC4, the integration of functionalities from WP2 such as key management, and security aspects from WP4 allows to achieve a trusted elastic cloud service. WT and EMC in charge of UC4, are also able to adequately address the security implications of this architecture by offering certifiable secure data management in the cloud. Therefore, the UC4 presents key innovative aspects providing a differential offer such as secure file sharing in multi-Cloud framework, secret storage, devices and sessions management enabling the provision of a competitive product in the current market (further detailed in D5.3).

By implementing the client-side encryption in the elastic cloud service use case (UC4), the trust boundary is as closely as possible to the user i.e. the data owner because the user does not have the necessity to put trust into any other party. Indeed and as described in D1.3, through encryption and proper key management, data owners can retain their keys protecting their data (files and sensitive data) that is stored in Cloud (resp. file storage and secret storage). Consequently, the data owner has the ultimate control over how their data is protected in the Cloud, how it is accessed and who may access it.

In addition and regarding Cloud architecture, the ECS solution developed in UC4 by WT and EMC, addresses the hybrid Cloud storage provisioning challenge by enabling the IT department to securely provision storage resources from an untrusted third party cloud provider. One of the primary objectives of a hybrid cloud solution is to enable the IT department (virtual CSP) to act as a trusted broker of cloud services sourced from both on-premise private clouds and off-premise public clouds to meet the rapidly growing demands of the business. The provision of the trust boundary is a major challenge and obstacle for organizations who want to consume public cloud resources as often the public cloud service providers do not offer the required security and privacy assurance levels.

5. Conclusion

The implementation of UCs 1-4 has played a role in the exploitation of the technology provided in the WPs 2-4. This report contains references to where this technology has been transfered into Open Source projects, and into productions. The objectives highlighted into the different UCs relate to user control and privacy, information sharing, and improved functionalities relating to the protection of information, and the management of encrypted documents. These objectives and the associated requirements have led to implementation applications which provides support to end users and provide better compliance with regards to existing and future European legislation.

The research result from UC1 on protection of data-at-rest, and key management solution has provided solutions developed by IBM research in the main code distribution of OpenStack Swift (release 2.9.0). Advanced key management have provided a contribution into the IBM Barbican external key management system. With these solutions users are able to manage keys used for encryption of data at rest instead of relying on a single key owned by the system. This is also being used in the context of the object storage platform Swift in OpenStack. The multi-party scenario with a mutual trust model functionality referred to as VICOS is still at an experimental phase, and not yet considered for production yet. There has been a strong transfer of knowledge from research to production, and there is still a pipeline of research ideas which will be implemented at a later stage.

UC2 which has a very clearly defined use case in the aviation industry, and specifically the MRO (Maintenance Repair, and Overhaul). The end result allows an organisation to perform encrypted querying on an encrypted database. SAP has provided the functionality of the search including the key management and the order preserving querying as an extension called SEEED (Search over Encrypted Data). This research is targeted to be deployed in the live SAP HANA database application.

UC3 is providing data storage encryption in the Cloud with the following properties. Users benefit from a combined user management and key management solution which allows the creation of encryption policies. This allows the creation of policies for specific users as well as application. The challenge has been to adapt the technology which was designed for volume encryption. The objective has been to provide this type of encryption policy management for object storage, and big data. Both additional use cases have been met. BT is currently exploiting volume encryption in the Cloud. The research work from ESCUDO-CLOUD is providing new solutions for BT Cloud Compute.

UC4 has WT and EMC working together to leverage a data encryption scenario with a shuffle index that obfuscates statistical access analysis by a third party Cloud host in order to conceal data owner access patterns. Various contributions are made including Cloud architecture, security, sharing and access. The scenario tested demonstrates achieve a trusted elastic cloud service. WT and EMC hence provide secure data management in the cloud. Innovative contributions are found in multi-Cloud framework, secret storage, devices and sessions management which create exploitation possibilities (described in D5.3). One of the achievements of UC4 is the ability to

securely broker hybrid cloud storage scenarios. The cross cloud trust model provides a solution to a common data storage scenario already commonly deployed in many organisations.

Bibliography

[BFT17] Daniel Bernau, Andreas Fischer, and Anselme Tueno. D1.5 - Use Case Prototypes. ESCUDO-CLOUD Deliverable, ESCUDO-CLOUD Consortium, December 2017.

[BHKR13] Mihir Bellare, Viet Tung Hoang, Sriram Keelveedhi, and Phillip Rogaway. Efficient garbling from a fixed-key blockcipher. In *2013 IEEE Symposium on Security and Privacy*, SP '13, pages 478–492, Washington, DC, USA, 2013. IEEE Computer Society.

[BK15] David Bowden and Florian Kerschbaum. D1.2 - Requirements from the Use Cases. ESCUDO-CLOUD Deliverable, ESCUDO-CLOUD Consortium, December 2015.

[DGK07] Ivan Damgård, Martin Geisler, and Mikkel Krøigaard. Efficient and secure comparison for on-line auctions. In *Proceedings of the 12th Australasian Conference on Information Security and Privacy*, ACISP'07, pages 416–430, Berlin, Heidelberg, 2007. Springer-Verlag.

[GHH+14] Patrick Grofig, Martin Haerterich, Isabelle Hang, Florian Kerschbaum, Mathias Kohler, Andreas Schaad, Axel Schroepfer, and Walter Tighzert. Experiences and observations on the industrial implementation of a system to search over outsourced encrypted data. In *Proc. of Sicherheit 2014*, Vienna, Austria, March 2014.

[HAK17] Md. Alam Hossain, Ahsan-Ul Ambia, Al-Amin , and Rahamatullah Khondoker. Measuring interpretation and evaluation of client side encryption tools in cloud computing. In *Security, Privacy and Reliability in Computer Communications and Networks*, pages 207–239, 01 2017.

[Ker15] Florian Kerschbaum. Frequency-hiding order-preserving encryption. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 656–667, New York, NY, USA, 2015. ACM.

[KS08] Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 486–498, 2008.

[KS14] Florian Kerschbaum and Axel Schröpfer. Optimal average-complexity ideal-security order-preserving encryption. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 275–286, 2014.

[KSS09] Vladimir Kolesnikov, Ahmad-Reza Sadeghi, and Thomas Schneider. Improved garbled circuit building blocks and applications to auctions and computing minima. In *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings*, pages 1–20, 2009.

[NIS01] NIST. Fips pub 140-2. *Security Requirements for Cryptographic Modules*, 25, 2001.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.

[PLZ13] Raluca Ada Popa, Frank H. Li, and Nickolai Zeldovich. An ideal-security protocol for order-preserving encoding. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, SP '13, pages 463–477, Washington, DC, USA, 2013. IEEE Computer Society.

[PRZB11] Raluca Ada Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 85–100, New York, NY, USA, 2011. ACM.

[PSDC15] Pramod Pawar, Ali Sajjad, Theo Dimitrakos, and David W. Chadwick. Security-as-a-Service in Multi-cloud and Federated Cloud Environments. In *IFIP Advances in Information and Communication Technology*, Trust Management IX. Springer, 2015.

[SDR15] Ali Sajjad, Theo Dimitrakos, and Rob Rowlingson. D1.1 - First version of requirements from the use cases. ESCUDO-CLOUD Deliverable, ESCUDO-CLOUD Consortium, June 2015.

[WFNL16] David J. Wu, Tony Feng, Michael Naehrig, and Kristin Lauter. Privately evaluating decision trees and random forests. *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2016(4), 2016.