Otto-von-Guericke-University Magdeburg

Faculty for Computer Science
Department of Technical and Business Information Systems

Master Thesis

A Survey and Classification of Privacy-Preservation Mechanisms for Cloud Data Management

Author:

Chunguang Hao

April 2, 2014

Supervisor:

Dr.-Ing. Eike Schallehn

University Magdeburg
Faculty for Computer Science
P.O.Box 4120, D–39016 Magdeburg
Germany

Hao, Chunguang
A Survey and Classification of Privacy-Preservation Mechanisms for Cloud Data Management
Master Thesis, Otto von Guericke Universitaet. Magdeburg, 2014.

Acknowledgements

I want to thank my supervisor Dr.-Ing. Eike Schallehn for all efforts and time that he spent on this thesis. He gives me many useful advices of how to write and organize my content. I also want to thank M.Sc. Stefan Barthel for giving me the original direction of this thesis. Last but not least, many thanks to my family and friends for their support and encouragement during this thesis.

Contents

List of Figures

List of Tables

; List of Abbreviations

API	Application Interface	9
ACL	Access Control List	
ACID	Atomicity, Consistency, Isolation, Durability	20
AWS	Amazon Web Service	5
BIFS	Bit-Interleaving File System	iv
BPOS	Business Productivity Online Suite	33
CRUD	Create Read Update Delete	11
CSP	Cloud Service Provider	6
CSA	Cloud Security Alliance	9
CAP	Consistency Availability Partition tolerance	20
CRM	Customer Relation management	9
DESX	Data Encryption Standard eXtended	12
DBMS	Database Management System	
DSP	Database service Provider	43
DAC	Discretionary Access Control	16
DTD	Document Type Definition	64
DaaS	Database as a Service	18
EFS	Encrypting File System	12
EC2	Elastic Compute Cloud	
HSM	Hardware Security Module	47
IAM	Identity and Access Management	14

IND-CKA	Semantic Security Against Adaptive Chosen Keyword Attack	54
IaaS	Infrastructure as a Service	6
IP	Internet Protocol	29
MAC	Mandatory Access Control	16
NSA	National security Agency	30
NTFS	New Technology File System	12
NTLM	NT LAN Manager	12
JSON	JavaScript Object Notation	
PII	Personal Identifiable Information	28
OS	Operating System	6
OLTP	On-Line Transaction Processing	
PaaS	Platform as a Service	8
PEKS	Public key Encryption with Keyword Search	50
QPS	Query Per Second	20
REST	Representational State Transfer	5
RBAC	Role Based Access Control	16
RDS	Relational Database service	
RDBMS	Relational Database Management System	15
RSA	cryptosystem designed by Ron Rivest, Adi Shamir, and Leonard Adleman	12
SASL	Simple Authentication and Security Layer	22
SLA	Service Level Agreement	31
SSLA	Security Service Level Agreement	
S3	Simple Storage Service	
SaaS	Software as a Service	6
SOAP	Simple Object Access Protocol	5

SAaaS	Security Audit as a Service	vi
TDE	Transparent Data Encryption	16
TCG	Trusted Computing Group	74
TPM	Trusted Platform Module	69
TXT	Trusted Execution Technology	
TTP	Trust Third Party	64
TF-IDF	Term Frequency Inverse Document Frequency	55
TVMM	Trusted Virtual Machine Monitor	77
TPA	Third Party Auditor	70
TCCP	Trusted Cloud Computing Platform	3
TC	Trusted Coordinator	
VMM	Virtual Machine Monitor	76
VM	Virtual Machine	6
XML	Extensible Markup Language	

Chapter 1

Introduction

Privacy preservation is always one of the major concerns when deploying IT technologies. Researchers have developed a set of frameworks and principles that effectively protect privacy in traditional IT systems. As the cloud is becoming the new tendency of building IT systems, old frameworks and principles face new challenges to provide sufficient protection of privacy. In this essay, I will have a survey and classification of privacy preservation mechanisms in the cloud environment. The motivation, the goal and the basic structure of my essay are stated as follows.

1.1 Motivation

Cloud is under great development in the IT industry. The novel approach of building the IT infrastructure gets much attention because of its great advantages. These advantages include: a large cost reduction of building new IT infrastructure, easier and cheaper maintenance of IT services, quick expandable IT architecture and the possible benefits of mobilization of IT resources.

From the perspective of IT service providers, the basic transformation is to provide IT services directly from their cloud to all the users instead of providing hardware and software modules to users which they had to build their own IT service infrastructure for. This transformation process is similar, if we compare cloud services to electric power service. Users can get electricity from a power station with little effort, they do not need to buy an electric generator and fuel for it. The billing system in the electricity industry is also similarly deployed in the cloud, which is called pay-per-use model. Users just need to pay for IT services in proportion to what they use.

From the perspective of IT service users, they can enjoy instant access to IT services without complex and time-consuming IT infrastructure building procedures. Considering the other advantages mentioned above, many more companies begin to move to the cloud. Struggling to enjoy all these benefits, privacy and security concerns become the major obstacle for them to deploy Cloud services [AAW13]. These major concerns are caused by losing control of the network, infrastructure and the most important value: data. In different business scenarios, when users deploy the cloud to get access to services, they have to outsource their sensitive data to the cloud. Traditional privacy preservation frameworks and security management architectures are not suited for the cloud model, they are initially designed to protect local IT systems. Without control of the IT infrastructure, these traditional architectures failed to provide sufficient protection of privacy and security. Possible damage to users may range from the least harmful but annoying spam emails or advertisements to disasters like personal bank account theft or even personal safety threats.

Yet without solid proof that security is assured and privacy preserved in the cloud, worries and concerns like data leakage and data confidentiality violation will greatly affect the trust and confidence in the cloud. The future development of the cloud will face great obstacles.

1.2 Goal

The goal of my essay is to give an overview of the existing privacy preserving mechanisms which may be implemented in Cloud computing environment.

Traditionally, preserving privacy can be considered as a part of information management. Information management is a long developed science in the IT branch. Researchers deploy the data life cycle model to define each phase of data management. Using information management as a reference, the data life cycle management is also applied in analysis of mechanisms of privacy preservation. I will assign these mechanisms to different phases of data life circle management according to their characteristics. Then the development of these mechanisms will be stated, the main approaches of these mechanisms will be researched and their advantages and disadvantages will be discussed.

1.3 Structure

The basic structure of this essay is as follows:

Background: This part will introduce the background information to my thesis, including two chapters separately talking about cloud issues and privacy and security issues. In the cloud chapter, the general cloud development history and the service model will be stated, then I will focus on the data management system in traditional local systems and in the cloud. In the second chapter, the privacy definition will be given, major privacy challenges in the cloud will be analyzed. Then according to different privacy challenges, different count-measures will be introduced. In summary, four general mechanisms are deployed in the cloud. They are: Anonymity based approaches, Encryption based approaches, Distribution based approaches and Trust based approaches. Anonymity based approaches will be generally discussed, but they are not discussed in detail due to their limited application scenarios.

Encryption based approaches: This chapter will have a research on the development of encryption based mechanisms. Different encryption algorithms will be stated, their restrictions and constraints will be analyzed. Also new designs of combination of these encryption algorithms will be stated. Combined with data life cycle phase assignments and various business scenarios, these approaches are classified into different categories.

Distribution based approaches: The general idea of this mechanism is to store data in multiple parties or separate service into multiple parts. The separated part of data or service is designed to be privacy preserving. Three approaches will be discussed in this chapter, they are: Bit-Interleaving File System (BIFS)[SMGL11], XML model[GWD14] and separating duties[HHK^{+}].

Trust based approaches: In this chapter, two mechanisms that aim to improve user trust on the cloud are discussed, Cloud audit and Trusted computing. Cloud audit approaches are separated into two categories, integrity audit and general security audit. At first, the trusted computing section will introduce the general background of trusted computing and then discuss two approaches, Terra [GPC^{+}03]and Trusted Cloud Computing Platform (TCCP)[SGR09]. Terra provides privacy preserving computation environment on single host and TCCP provides a privacy preserving computation pool in the cloud.

Conclusion and future work: In this chapter, general analysis of these privacy preserving approaches in the cloud is given. Future direction of preserving user privacy data in the cloud is discussed.

Chapter 2

Cloud Related Issues

In this chapter, the general development history of the cloud will be described, the cloud deploy model and service model will be introduced. Then the focus of the Cloud issue, Cloud data management, will be researched. At the end, the data life cycle model will be introduced and mapped to Cloud data management. This model will help us analyze the data management process.

2.1 Cloud Development

2.1.1 Cloud History

In 2006, Amazon began to launch a new product called Amazon Web Service (AWS). This new product is not like physical objects or digital objects sold on the Amazon website, like books or CDs. This new product is the computation service. Amazon Web Service (AWS) is a collection of remote computing services (also called web services) that together make up a Cloud computing platform. AWS customers can get access to services over HTTP, using Representational State Transfer (REST) or Simple Object Access Protocol (SOAP) protocols [Wik14c]. This novel approach attracts much attention from IT enterprises. Later this approach is named and marketed as Cloud. IT enterprises treat the cloud as the next IT transformation, and the concept of providing services from the cloud becomes the new tendency.

The Cloud concept is the natural evolution of Internet development. The early stage of IT development is that mainframes are the center nodes of the internet and limited users get access to computation services via terminal. Mainframes are so expensive and complex that only large research institutions are able to build and maintain them. Accompanied with the technological progress, the computation power of personal computers is strengthened and the cost is reduced. The development of the internet is hence sped up. At the same time, IT enterprises try to build their IT infrastructure with large amounts of cheap computers instead of using expensive mainframes. Despite of the strong reliability and stability of mainframes, the maintenance complexity and the difficulty of building elastic IT infrastructure limits its use. Combined with other technologies, like grid computing and fault-tolerant computation architecture, IT enterprises successfully built IT infrastructure, which we call the cloud nowadays. Besides of their own use of services provided by these IT infrastructures, they are also able to provide services to other users and make profit of it. Amazon puts forward the first step in this new market and other IT companies quickly follow. Traditional internet giants like Google and Microsoft have their own Cloud services for customers. Because the infrastructures from different IT enterprises are similar but not the same, services provided from these Cloud Service Providers (CSPs) also vary because of their different purposes of building their own infrastructures. Considering the service differences provided by Cloud Service Providers (CSPs), the definition of the cloud has no standard. The definition we usually use is from NIST [MG11]:

> Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Under this definition, Cloud computing should have these five essential characteristics:

- on-demand self-service
- broad network access
- resource pooling
- rapid elasticity
- measured service

To provide these essential characteristics, one of the fundamental technologies deployed by CSP is virtualization. Virtualization means the abstraction of physical resources, including network, storage, memory and CPU. Through the virtualization of these resources, CSPs will get better logic isolation to the actual complex management of hardware. With this logic isolation, the cloud infrastructure management system controls and schedules sets of Virtual Machines (VMs), i.e., pools of physical resources. One example of these management systems is called Sun xVM hypervisor. Infrastructure as a Service (IaaS) providers including Amazon (EC2), ServePath (GoGrid), and Sun Cloud employ this type of virtualization systems, which enables customers to run instances of various operating system flavors in a public Cloud [MKL09]. The structure of the system is described in Figure2.1.

In Figure2.1, it is clear that the hypervisor plays a similar role like Operating System (OS) in the traditional computer system. It is in charge of hosting guest OSs and coordinating the smooth switch between these OSs. Different types of OSs such as UNIX, Linux or Windows could run in this hypervisor environment like normal application softwares run in traditional OSs. Other applications remain unmodified and full functional running on guest OS. Instead of assigning physical machines to different users, these OSs represent the basic computation control and schedule units. They are also named as Virtual Machines (VMs). It is needed to specify that, Virtual Machines (VMs) are assigned to users when users deploy the Infrastructure as a Service (IaaS) model. When users deploy other service models, for example, the Software as a Service (SaaS) model, software instances instead of VMsare assigned to users. In the next section, different service models and also Cloud development models are described.

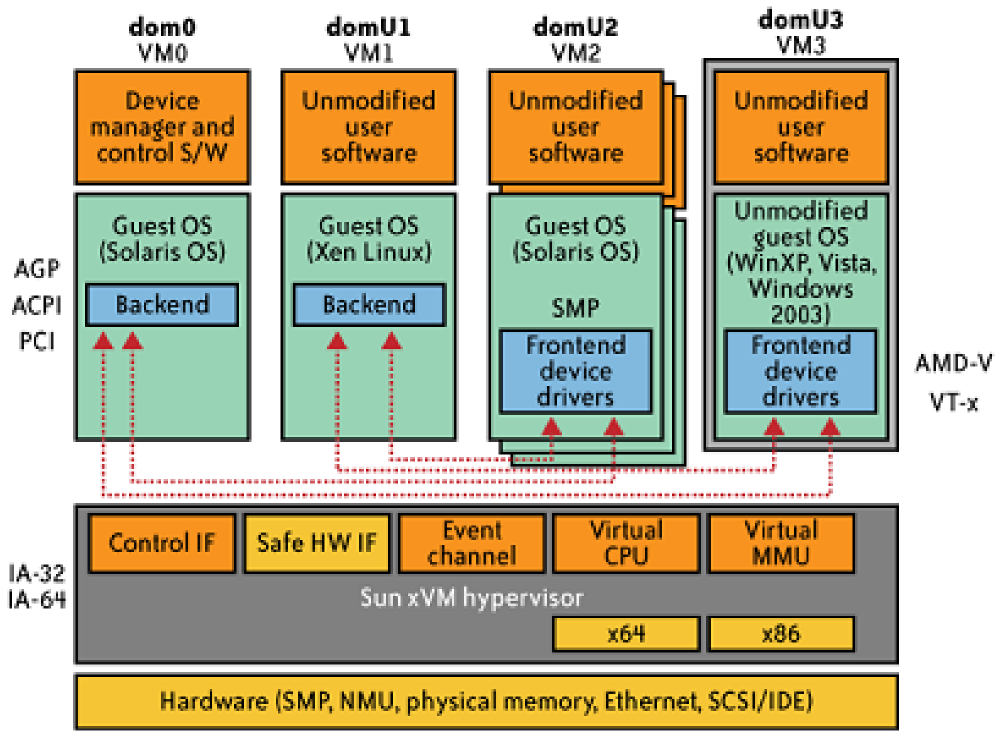


Figure 2.1: Sun xVM hypervisor environment [MKL09]

2.1.2 Cloud Deploy Model and Service Model

By deploying virtualization and other technologies, institutions and organizations could also build their Cloud. According to the difference between development purposes, Cloud development models include [MG11]:

- Private Cloud: The Cloud is built and used only inside of enterprises or institutions, the main purpose is to achieve rapid elasticity and convenient management and maintenance of IT infrastructure. This is exactly what Amazon and Google did before they provided services to the public.

- Community Cloud: The Cloud is built and used inside of a community. Enterprises or institutions share the same demands of Cloud services.

- Public Cloud: This is the most common deployment model of the cloud. Cloud infrastructure and resources are provided by third parties, resources are shared between many users. Enterprises or institutions can get access to Cloud services over the internet, they have little or no control of Cloud infrastructures. This model causes major concerns to enterprises and institutions. And most of privacy preservation mechanisms are researched in this model.

- Hybrid Cloud: This is the combination of private Cloud and public Cloud. Users Cloud store sensitive data and applications in private Cloud to achieve better control and security, less sensitive data and applications could be stored in the public Cloud to reduce costs and minimize the management efforts.

When enterprises and institutions deploy the public Cloud model, different service models are also provided. They have to decide which service model to employ, it is also normal to employ different service models from the same CSP or different CSPs.

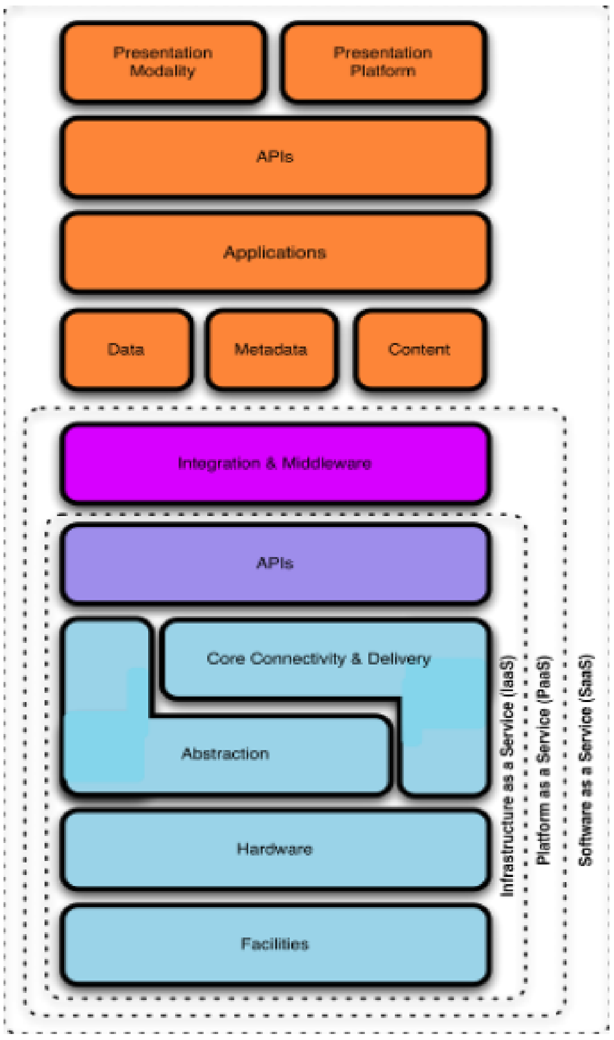


Figure 2.2: Cloud architecture and services [BM+11]

In Figure 2.2, the relation between different service models is described. Generally, the cloud services are classified into three models based on a principle similar to how we separate the application software, system software and system itself. Other books may call this Cloud service classification as service delivery model. These three levels of services are:

- Infrastructure as a Service (IaaS) : representative examples: Amazon EC2, S3, etc. Using S3, the user could store data in data centers of Amazon. This data could be accessed conveniently via the internet, other management efforts are not needed.

- Platform as a Service (PaaS) : representative examples: Google App Engine, Microsoft Azure, etc. Google App Engine provides a standard platform for developers to build and run applications. They do not need to put much effort to build and maintain the running environment, they share the same infrastructure on which

Google runs its various applications and services. Thus there is no need to worry about the reliability and availability. Using Google App Engine, they could focus on the application logical design and data processing, this greatly improves the develop efficiency.

- Software as a Service (SaaS) : representative examples: Google docs, Gmail, etc. Google docs allows users to create documents via the internet browser and store documents in Google Cloud. Google Cloud will automatically backup all the documents. Users do not need to buy expensive office software and take care of the documents storage and backup.

It is clear that, Figure 2.2 IaaS is at the bottom of architecture, users have no direct control of physical machines, instead they have full control of VMs. These VMsare assigned by hypervisor, which is described in Figure 2.1. PaaS is the mid-level of Cloud architecture. In this environment, platform means the software development platform, including storage, development tools, libraries and so on. SaaS is the top level of Cloud architecture, users do not need to install software on their local computers, they could get software services via Application Interfaces (APIs) or HTTP protocol. In this service model, besides limited operations allowed and provided by CSPs, users have no other control of Cloud resources. CSPs may deliver services under all three models or focus on one model as their market strategy differs. Because of the dependency hierarchy, CSPs who can provide IaaS are also available to deliver PaaS and SaaS models. For example, with Google, we can get access to Google drive, Google App Engine and Google mails at the same time. After years of development, the cloud ecosystem is becoming mature,SaaSCSPs could also be customers of IaaS providers. The Cloud market is also getting more subdivided. According to the survey from Cloud Security Alliance (CSA), until 2011, the representative Cloud services classification is demonstrated in Figure 2.3. Among all of these services, the data storage and management services are the main focuses of this essay. From the taxonomy, typical related services are selected and stated as follows:

Cloud software: Mango DB, Appache CouchDB, Cassandra, Berkeley DB and etc.

Infrastructure services: Amazon S3 and EBS, Rackspace Cloud Files and etc.

Platform services: Amazon simpleDB, Amazon RDS and etc.

Software services: In this category, almost all the services are integrated with data storage, especially when users use billing services, social networks, Customer Relation management (CRM) or Financials from CSP. Data storage of these services is not local, it is stored in the cloud using Cloud databases or the combination of Cloud storage and Cloud database.

The classification above is driven from the perspective of the business model. From the perspective of the internal logical connection and data storage types, these services connected with storage are re-summarized as follows:

- Structured Relational Data Storage: such as Amazon RDS(various database instances available, include MySQL, Oracle, PostgreSQL, SQL server), Azure Cloud

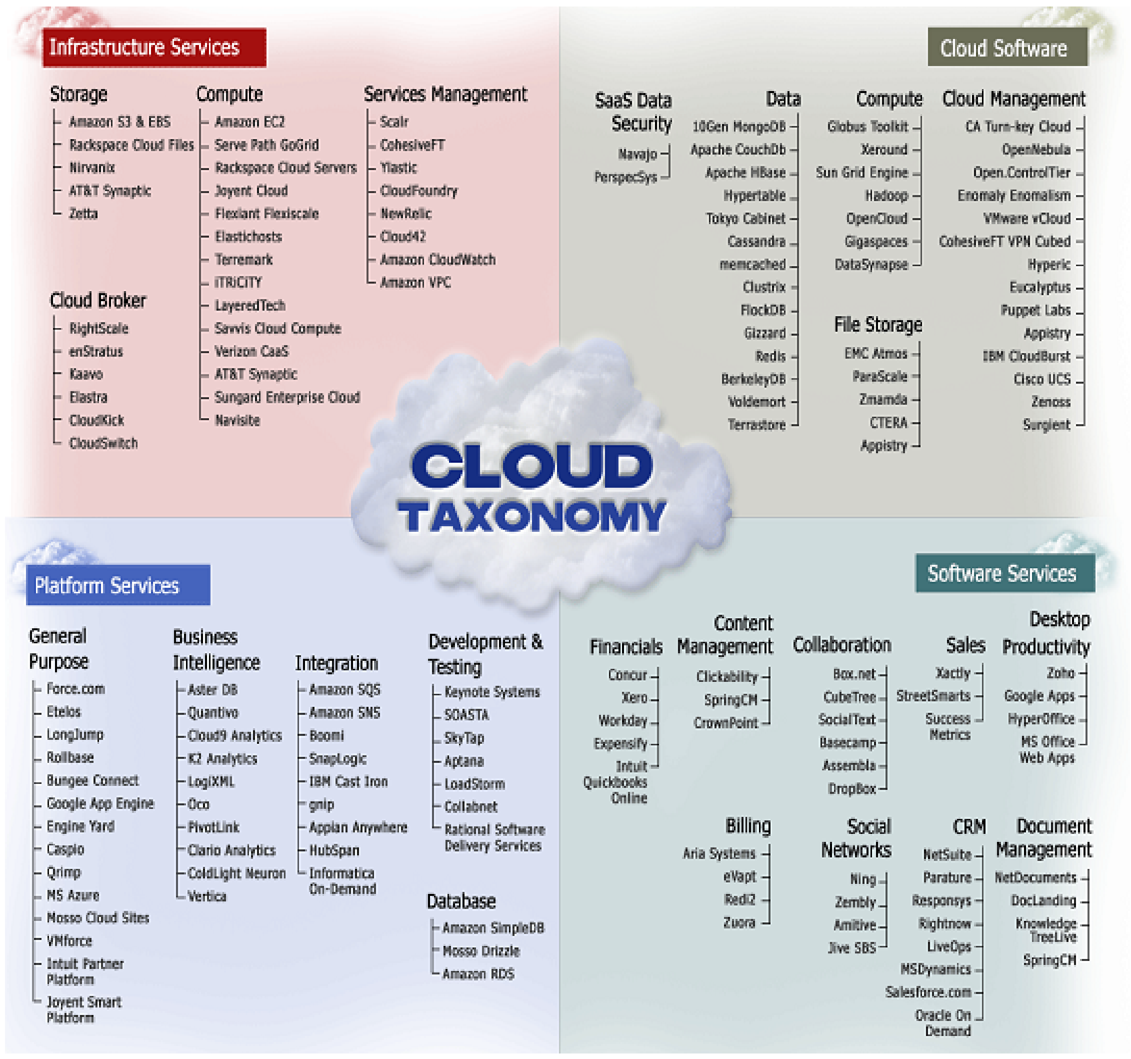


Figure 2.3: Open Cloud taxonomy [BM+11]

SQL database (using SQLserver instance), Google Cloud SQL (using MySQL instance).

- Structured Non-relational data storage: such as Hbase, Mango DB, Cassandra and Appache CouchDB.

- Simple Object Storage or File Storage: such as AmazonS3, Google drive, Microsoft skydrive and Dropbox.

According to the summarized Cloud services, these three types of Cloud data management will be researched in the next section.

2.2 Data Management in Cloud and Traditional System

Data stored in the cloud is categorized under three types. The traditional data stored in a local network is classified into two main categories: database data, and file storage. There are many types of databases. Since relational databases are the most widely used, they will be the focus of this thesis. This is also the reason why structured relational data is classified under a separate category in 2.1.2. These two kinds of data storage could always be easily mapped to simple object storage, and structured relational data storage in the cloud. The other type of structured non-relational data is then individually specified. The comparison of each type of data storage is described in section three.

2.2.1 Simple Object Storage

In the text bellow, the connection and the comparison between traditional local storage and Cloud storage is described. The main comparison focuses on the basic operations and access control for these two kinds of storage.

File storage includes all kinds of file types, such as .doc, .mp3, .mp4, .mpeg, .avi, .txt, etc. All of these files share common attributes, including, but not limited to: file name, file type, file date, and file author. These attributes could also be summarized as meta-data. Files stored in local disks or local networks could be separated into two parts: the file contents, and the file description information or meta-data. In OS or local networks, file systems are responsible for file storage management. There are many file systems designed and deployed in different operating systems. According to their design principles, or the focus of the application, the functionalities provided also vary. Most widely used OS file systems include:

- Windows: NTFS, FAT32, FAT16, etc.
- MAC OS X: HFS
- Linux: ReiserFS, ext, ext2, ext3, ext4, XFS, NTFS, HPFS, NFS, etc.
- UNIX: JFS (IBM AIX), ZFS (solaris), etc.

Among all of these file systems, four basic file operations namely, Create Read Update Delete (CRUD) are supported [BB08]. Because of the fundamentality of these operations, they are supported in almost all of the software applications. These four operations could be easily mapped to SQL instructions and HTTP commands. The table below shows the detailed map for each operation.

Besides these basic operations, another important part about file storage is the security control of the file storage system. General approaches deployed are: access control policies, Additional passwords and encryptions.

The fundamental security control approach is supported by the file access control system. Most of the access control policies are based on ACL or capability-based security. Capability-based security is the following concept: use capability (named in some systems as key) as a communicable, unforgeable token of authority. This token represents a

Operation	SQL	HTTP	Amazon S3
Create	INSERT	PUT/POST	POST/PUT
Read	SELECT	GET	GET
Update	UPDATE	PUT/PATCH	PUT Object (Copy)
Delete	DELETE	DELETE	DELETE

Table 2.1: CRUD mapping to SQL, HTTP and Amazon S3

value that references an object and the associated set of access policies. Other users or programs should use the corresponding capability to access an object. But this security control policy is not supported by most commercial products.ACL is implanted and supported in most commercial products. The code below shows an example of setting an ACL on a file *test.doc* in Linux system.

```
$ setfacl -s user::rw-,group::r–,other:—,mask:rw-,user:Guang:rw- test.doc
```

By executing the command showed above, the file owner permissions are set to read-/write, file group permissions to read only, and other permissions to none on the test.doc file. In addition, the user Guang is given *read/write* permissions, and the ACL mask permissions are set to *read/write*, which means that no user or group can have execute permissions on the test.doc file.

From the example above, it is better to understand the ACL policies. Access control policies are assigned for user groups or a single user by specifying the allowed operations on each file object. ACL is able to achieve file security to a certain extent.

The second approach is to encrypt data before storing it in a file system, users could encrypt data by using their own third party software and tools, or they could use encryption function supported by the file system. What is critical for users deploying encryption measures is the key management of the encryption. If only few data objects need to be encrypted, it would not be difficult or complex to use third party encryption software. For a large amount of data objects, users need to build their own key management system locally. This task is beyond of capabilities of most normal users. Hence using encryption functions provided by the file system is a much easier choice. Take New Technology File System (NTFS) as an example, the Encrypting File System (EFS) feature is provided since NTFS version 3.0. This feature enables file-system encryptions [BGG+01], hence the encryption/decryption is transparent for the application. The basic process is described in figure 2.4 [Mic14a].

Plain text is encrypted with the Data Encryption Standard eXtended (DESX) algorithm, using symmetry encryption key. After plain text encrypted with this file encryption key, then this encryption key is encrypted with cryptosystem designed by Ron Rivest, Adi Shamir, and Leonard Adleman (RSA) public key produced by using user certificate. A file encryption key is encrypted by RSA public key and then stored together with encrypted file data. At the same time, RSA private key that used to decrypt file encryption key is produced. The user's RSA private key is encrypted using a hash of the user's NT LAN Manager (NTLM) password hash plus the user name. When users log into the OS, by providing the correct password, the RSA private key is then automatically activated to decrypt encrypted file encryption key. Original plain text could be recovered by decrypting encrypted data using file encryption key. Users do not

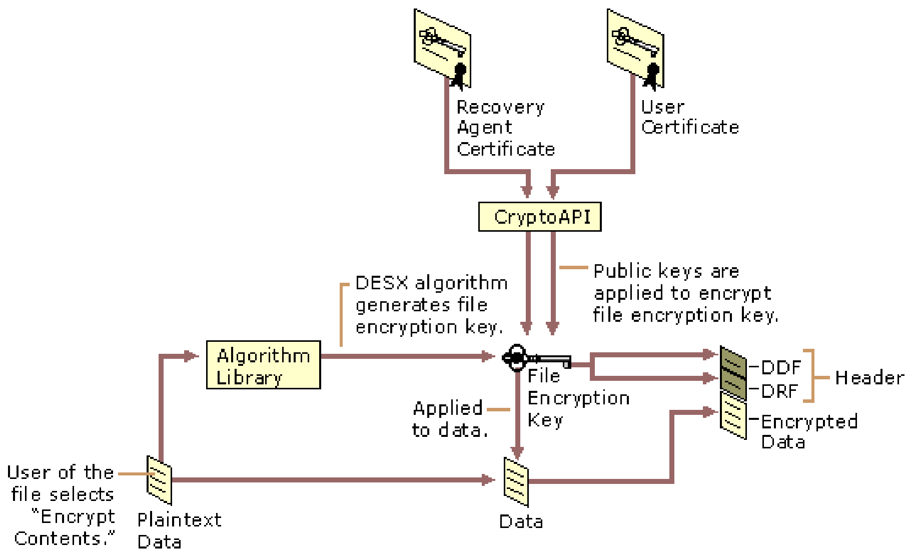

Recovery Agent Certificate
User Certificate
CryptoAPI
DESX algorithm generates file encryption key.
Public keys are applied to encrypt file encryption key.
Algorithm Library
File Encryption Key
DDF
DRF
Header
Encrypted Data
Applied to data.
User of the file selects "Encrypt Contents."
Plaintext Data
Data

Figure 2.4: Generation of an Encrypted Data File from [Mic14a]

need to provide private key by their own. For a better use of this system, recover agent certificate is set up to provide emergency recover of encrypted data. The other details of EFS could refer to [Mic14a].

Another alternative is to use password. It could be considered as a simplified and weaken version of encryption. This is useful for some certain file types, such as Microsoft word document and .wmv video. For these file formats, other users have to input the correct password to allow the correspondent software to open the file and to read the content. This is especially useful when sharing data with other users. Only users who have correct password could read the content. Again, for large amounts of documents that need to be protected, a proper password management system should be built.

In the cloud environment, simple object storage is similar to file storage in the local system, they share many commons. A representative example is Amazon S3. Taking file storage in the local OS as a reference, the comparison of data object operations and security control measures in S3 is described in the following text.

Amazon S3 is a highly reliable, scalable, secure and fast accessed storage infrastructure. It is provided both internal use to Amazon itself and external utilization to customers. The basic data model in Amazon S3 is described as follows:

> "Object is the fundamental entity that can be stored in S3. It contains the object data and its metadata. Metadata is a set of name-value pairs that describes the object. . . . Each object must be contained in one Bucket. . . . So an object is uniquely identified within a bucket by a key (name) and the version ID. Objects can be addressed by a combination of bucket name, key, and optionally version ID." [Moh11].

Together with Amazon simpleDB which indexes metadata of S3 objects, users get a powerful and convenient Cloud storage service.

Basically CSP try to provide a similar environment for users to store data in the cloud as they manage files in local systems. Basic operations of S3 mapping to CRUD is described in 2.1. Create, read, delete is supported by direct instructions. For update operation in S3, there is not direct instruction provided. To update an object that stored in Amazon S3, users can either delete the old object and then put a new object or direct put a new version of the object to the cloud like normal put new objects operation. It will overwrite the existing object and then update meta-data of the object. There are other instructions provided by Amazon S3, including [Ama14b]:

- Instructions that used to easy the basic data operations: Delete Multiple Objects, GET Object torrent, POST Object restore, PUT Object - Copy, Initiate Multipart Upload, Upload Part, Upload Part-Copy, Complete Multipart Upload, Abort Multipart Upload, List Parts.

- Instructions that used to control data access: GET Object ACL, HEAD Object, OPTIONS object, PUT Object ACL.

The first category of additional instructions is mainly designed to provide a convenient data management. The second category is instructions that designed to provide access control management of data objects. For example, the HEAD operation retrieves metadata from an object without returning the object itself. And a browser can send OPTIONS/ Objectname request to Amazon S3 to determine if it is allowed to send an actual request with the specific origin, HTTP method, and headers [Ama14b].

For different CSPs, because of the non-standardized Application Interfaces (APIs), instruction names of these CRUD operations may be different, but the actual effects of operations on the data object are the same. According to the diversity of system design requirements, additional operations that supported by different CSPs are also different.

Another data management feature that supported by almost all the CSPs is the security control mechanisms. Similar to the security control mechanism of file storage in the local OS, security control mechanisms for simple object storage in the cloud are also composed of different approaches. Again, taking Amazon S3 as an example, it provides four different access control mechanisms. Which includes: Identity and Access Management (IAM),ACL, bucket policies, and query string authentication. "IAM enables organizations with multiple employees to create and manage multiple users under a single AWS account. With IAM policies, you can grant IAM users fine-grained control to your Amazon S3 bucket or objects. Amazon S3 Bucket Policies can be used to add or deny permissions across some or all of the objects within a single bucket. With Query string authentication, you have the ability to share Amazon S3 objects through URLs that are valid for a predefined expiration time" [Ama14b].

ACL approach deployed in S3 has some differences comparing to ACL approach in the traditional file system. There are only five operations defined: Read, Write, ReadACP, WriteACP, FullControl [Ama14b]. Among these operations, *ReadACP* allows users to list the ACL of the buckets and objects; *WriteACP* allows users to write the ACL of the buckets and objects; *FullControl* allows users to have full control of the buckets and objects.

Bucket Policies: A Bucket is a container for objects. Each object must be contained in one Bucket. An object is uniquely identified within a bucket by a key (name) and the version ID. Objects can be addressed by a combination of bucket name, key, and optionally version ID.

Query String Authentication: It can be used when a third party browser needs to get access to resources on S3. The owner of these resources must specify an expiration date of the query, add a signature, place the data in a HTTP request and then distribute it to other users or embed it in a webpage.

Another security mechanism that deployed in S3 is encryption. S3 enables both server-side and client-side encryption. Encryption is a standard approach to protect data confidentiality and preserve privacy in the cloud.

This mechanism will be detail discussed in chapter 4.

2.2.2 Relational Data In The Cloud

In this section, structured relational data is researched and compared between traditional and Cloud database services. The comparison focuses on the database security management.

Relational database is the major database type in the business. Relational data model is a mature model, in which the basic operations defined are based on relational algebra. Because of its solid mathematical foundation, relational database model is a highly formatted and strict logical system. A typical relational table is structured as a set of attributes, this set is also named as a relation. An example of a relation table is described as follows:

Primary Key	Attribute1	Attribute2	Attribute3	..	..	Foreign Key
key value	value	value	value	..	..	key value

Table 2.2: Typical relational table structure

To improve database efficiency, users can build indexes on attributes, including primary key and foreign key. The most common relational data operations in relational database are listed as follows:

- Select: syntax like: SELECT (Attributes) FROM (Tables) WHERE (Conditions)
- Insert: INSERT INTO (Table) VALUES (Attributes values)
- Update: UPDATE (Table) SET (Attribute=value) WHERE (Conditions)
- Delete: DELETE FROM (Table) WHERE (Conditions)

In the *WHERE* clause, users can specify *Conditions* by using relational operators, such as $=, <, >, \geq$ and $\leq$ to define the scope of data operations.

ThemMost widely used Relational Database Management Systems (RDBMSs) include standard business RDBMSs, such as DB2 from IBM, Oracle series from Sun, SQLserver from Microsoft and open source databases, such as MySQL, Postgre SQL, SQLite. Most of these databases provide following security control mechanisms:

- Access control: It manages the rights assigned to different users, and control their accesses to data stored in DBMS. Access control system is integrated with identity control system.
- Encryption of data: This is an additional security measure to prevent improper accesses to sensitive data.

- Audit: Another mechanism that supports many functionalities, including security control, system monitoring, static research and so on.

The first mechanism: Access control is the most important mechanism to keep database protected against malicious attack or unauthorized access. There are three access control model most widely used, they are described as follows [San96]:

Mandatory Access Control (MAC): This model bases on the idea that: users are assigned with different access tags, objects are also assigned different access tags. Only users who have higher access tags than the access tag of an object are permitted to access this object. This model enforces strict one-direction information flow and hence achieves great security protection level.

Discretionary Access Control (DAC): The basic idea of this model is: the owner of an object has full access control of the object. The owner could authorize other users to get permissions to operate this object. Considering the inherent weakness that information could be copied, and the whole security is based on the correct decisions and behaviors of a single user, this model is not deployed by most DBMS

Role Based Access Control (RBAC): This model is deployed by most DBMS. The key idea is to create roles as sets of access control policies. Each role is created based on the specific task partition, for example, system administrator, system user, security administrator. Users are assigned with different roles; one user could have more than one role. The flexibility and convenience of database management are the main advantages of this model.

To prevent improper accesses to sensitive data, such as medical production secret, access control system is not full qualified. Data encryption is a standard solution to keep data confidentiality. There are normally three layers of encryption in traditional DBMS:

OS level encryption: At this level, database files are encrypted. For example, when using encryption API provided by the file system, database files are considered as normal files and encrypted. Because the inner logic relations between database files could not be recognized, encryption at this layer results in a complex key produce and management system.

DBMS kernel encryption: This approach is mostly developed by the developers of the DBMS product. Before storing data to physical storages or retrieving data from them, inner DBMS encrypt engine is responsible to encrypt/decrypt data. This approach is the best solution for normal users. It could be integrated with the access control mechanism to achieve better security. Applications could use data transparently. This is called Transparent Data Encryption (TDE). Different database products have different supports on this feature. Oracle TDE will be discussed and analyzed as an example.

DBMS tools encryption: Using tools or softwares to encrypt data means to store encrypted data as attribute value in database. This approach has the same problem, which is that additional key produce and management has to be developed and supported. Another problem of this approach is that, data encryption granularity is limited in attribute level. Metadata, logs and indexes could not be encrypted. Data constrains have to be checked before encryption, this is complex and difficult to implant.

From the above description, it is clear that DBMS kernel encryption is the best way to protect data confidentiality. Encryption is not well supported in all the DBMSs, MySQL and DB2 provide the encryption functions as internal SQL functions. In DB2 version(9.7), encryption and decryption function include: ENCRYPT,$DECRYPT\_BIN$,$DECRYPT\_CHAR$, GETHINT. Users can only encrypt data types such as*CHAR, VARCHAR, VARCHAR FOR BIT DATA*. The encryption key is a password set by user, which users have to manage it himself. A useful help is users can set password hint and by using *GETHINT* function, users could get hint of the password [IBM14]. This system is simple and rough, it is not convenient to manage data encryption in it.

Oracle has provided another encryption solution comparing to other databases. The oracle database encryption feature is developed from the basic encryption API to the Transparent Data Encryption (TDE); the newest feature supports TDE table space encryption [Ora14]. The following graph is the system overview of oracle TDE:

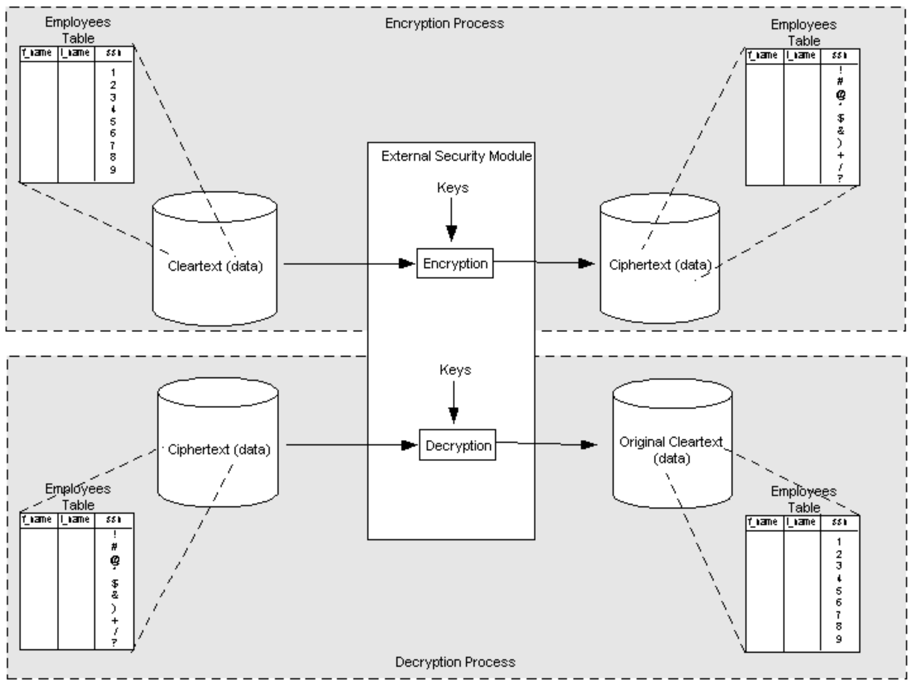


Figure 2.5: Oracle TDE overview from [Ora14]

Oracle *11g* supports attribute and table space encryption. These two types of encryption are not the same mechanism. Attribute encryption happens during the SQL call, and the table space encryption happens before storing data to disks [Ora14]. To use encryption feature in Oracle *11g*, users need to open the encryption wallet. Oracle wallet is an external security module that used to manage the master encryption key and column encryption keys. Encryption keys are automatically managed without user intervention; users just need to specify the password to open the wallet. The detail setting process is described in [Ora14].

Beyond of these benefits, encryption is still not fully matured. There are some weaknesses and limitations to be mentioned, for example, the potential performance reduction due to additional encryption and decryption processes. Other limitations include: encryption could not be performed on the index attribute and foreign key, range query and other queries are not available. Considering the limitations and weakness of these DBMS encryption, new encryption algorithms are being developed and also other mechanisms are being researched. Other solutions trying to break these limitations are being researched, details will be specified in chapter 4.

Audit is the third approach deployed in the database security management. This approach aims to monitor and record selected database actions. Based on the differences of which object or what kind of actions is monitored, audit is classified into several types. In oracle system, different auditing types are described as follows [Ora14]:

Statement Auditing This enables user to audit SQL statements according to the type of the statement. For example, *AUDIT TABLE* will audit all the *CREATE* and *DROP TABLE* statements.

Privilege Auditing This enables user to audit the use of system privileges and corresponding actions.

Schema Object Auditing This enables user to audit statements on a particular schema object.

Fine-Grained Auditing This enables user to audit the most granular level of SQL statements, data accesses and actions. For example, if an attribute value is changed beyond the allowed range, audit system will record related information in logs.

With all of these types of audit, audit system is typically used to investigate suspicious activities produced by users or occurred on particular data (sensitive data). It helps system administrators to detect problems regarding improper implementation of access control system. Another function of audit system is to gather statistic data of database activities and then improve database design and efficiency.

After a general illustration of security control mechanisms in local DBMS, relational data storage in the cloud is discussed in the following text. The mature business model to provide relational database services is to implant existing database in the cloud infrastructure. For open research uses, there are also newRDBMSs being developed to utilizing the benefits of the cloud. This new type of RDBMS is aiming to provide better performance, scalability and availability. Some prototypes are developed, for example H-store and VoltDB [ES12]. The focus of this thesis is mature Cloud database services. There are some basic service models to discuss. The first is IaaS model, under this model, users get a VM instance. Then users could install RDBMS in this VM instance as they used to do locally. The second service model is PaaS, under this model, users can choose DBMSs according to their requirements, CSPs provide the chosen database instance for users. For example, users could choose MySQL instance, Oracle instance, PostgreSQl instance or SQLserver instance from Amazon RDS. This service model is also named as Database as a Service (DaaS). The main difference between these two service models is: under IaaS model, users have to manage the complex maintenance and backup tasks and under PaaS model, these complex jobs are done by CSP.

Despite the differences illustrated above, from the user perspective, using Cloud database services is not different with using the local database services. From the CSP perspective, to provide database services is much more different and challenging. Figure 2.5 described the system overview of Amazon RDS. To provide reliable services, automated backup and load balance, Cloud relational database works together with other services including Amazon EC2 and amazon S3.

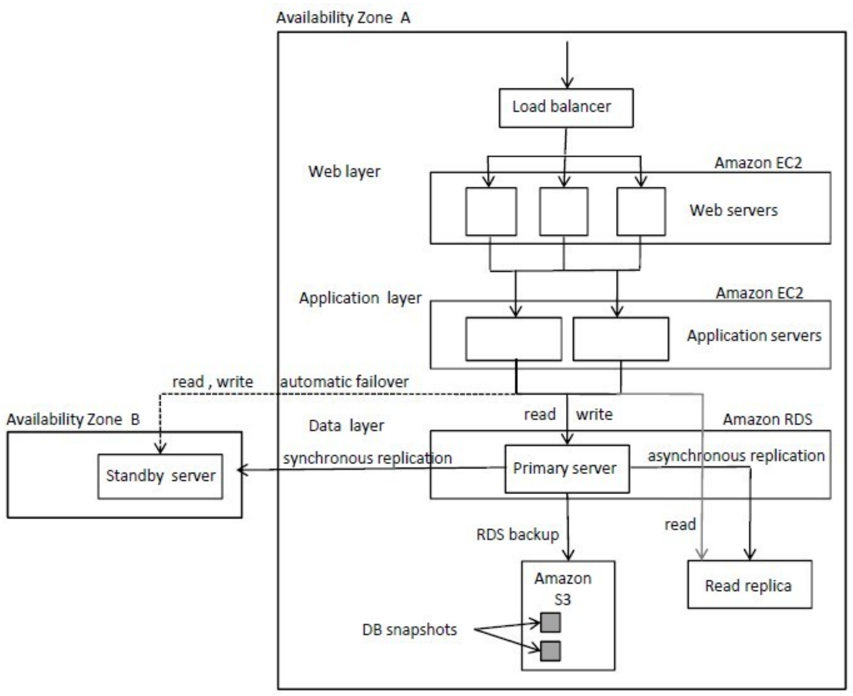


Figure 2.6: Amazon RDS architecture [Moh11]

In the architecture described in Figure 2.6, it is clear that to provide RDS, CSP Amazon has to combine all of its infrastructures.

Amazon EC2 is used in web servers and application servers to provide high availability and performance. At the back-end, Amazon S3 is used to provide support to backup function of databases. For the other services provided by other CSPs, similar structure is also deployed.

When comparing the security control mechanisms of structured relational data in the cloud to traditional relational databases in the local network, as it illustrated above, the DBMS core is not modified under both service models. Hence the access control mechanisms, data encryption and audit could be the same with traditional DBMS. These mechanisms that deployed in the cloud cause concerns similar to traditional inside security threats. For example, even encrypted data is stored in secret form on Cloud disks, data remains plain text in the cloud machine memory. Audit system is supposed to record unauthorized accesses to sensitive data, but potential attackers may be able to alter DBMS audit logs to avoid triggering security alerts.

When CSP is not trustworthy, how to achieve data confidentiality and protect privacy data. This problem will be researched in chapter 4.

2.2.3 NO-relational Database in The Cloud

The third type of data storage is structured non-relational data storage. This kind of data storage is developed due to the requirement of big data processing and real-time web. For IT companies who have critical demands to process PB level data, such as Google, to provide high availability and performance to their applications, traditional data models and data management systems are not suitable [ES12]. Strict relational model has limited performance in these application scenarios. Though relational database model could achieve high Atomicity, Consistency, Isolation, Durability (ACID) properties, it results in great sacrifice on availability and scalability. For some particular applications, for example, web pages storage in Google, the update of data is achieved by adding new data versions which are identified by timestamp, the consistency of these data is not strong demanded Other characteristics of these particular application scenarios include: simple and flexible data model, extreme big table size, extreme high Query Per Second (QPS) or high operations on rows. To provide high performance in these scenarios, distributed data replication and partition combined with distributed parallel processing is developed. Consistency Availability Partition tolerance (CAP) model presented by Eric Brewer shows that, in distributed systems, only two properties could be achieved at the same time [Moh11]. There are mainly three approaches to deploy this kind of data management services, including Google approach, Amazon approach and Hadoop approach [ES12]. Google has developed a new architecture of distributed parallel computing. Similar architecture have been developed and deployed by other companies like Amazon, which has similar requirements to provide high-performance web services. The basic structure of Google approach is described in Figure 2.7. Structured data systems

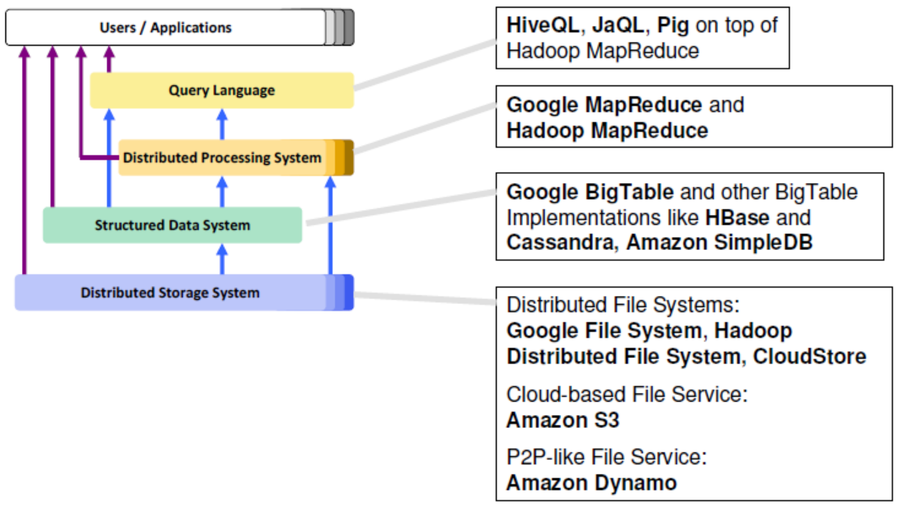

Users / Applications
Query Language
Distributed Processing System
Structured Data System
Distributed Storage System
HiveQL, JaQL, Pig on top of Hadoop MapReduce
Google MapReduce and Hadoop MapReduce
Google BigTable and other BigTable Implementations like HBase and Cassandra, Amazon SimpleDB
Distributed File Systems: Google File System, Hadoop Distributed File System, CloudStore
Cloud-based File Service: Amazon S3
P2P-like File Service: Amazon Dynamo

Figure 2.7: Structure overview of the cloud and component example from slides [ES12]

are at the second level of this structure. Similar to the relational database discussed above, a detail analysis is given in the following text. The most famous system is Bigtable from Google. Bigtable is a high performance data storage system that initially used by Google internally. After Google releases a series of papers describing the system design, Hbase is developed as an open sourced version of Bigtable. It is supported by Apache Software Foundation. Hbase is a column based real-time database system that provides high reliability, scalability, high performance [Apa14]. Hbase is built on hdfs. It is part of Hadoop, an open source project that aims to build a reliable and scalable distributed computing framework. The framework architecture is described in Figure 2.8.

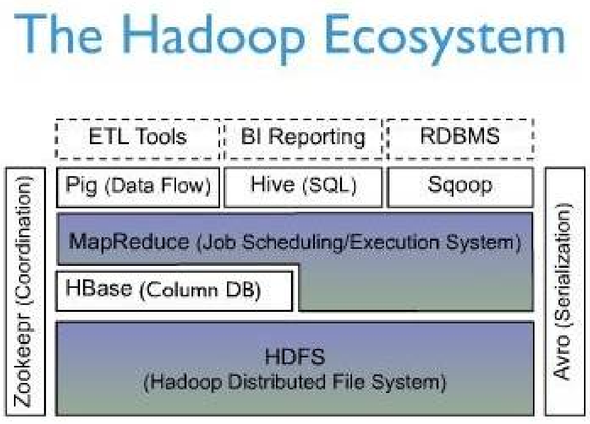


Figure 2.8: Hadoop system [had14]

In this framework, HBase acts as an interface layer between applications and file storage system. HBase is a distributed scalable big data store, which stores very large tables. It is non-relational and part of NoSQL database. The data model in HBase is after Google Bigtable designed. Data stored in HBase is organized into tables, in each table, *row* is the basic logical unit. It is identified by *row key.* In each row, there are *column families* defined in the table schemas before creating tables. In each column families, various columns could be defined. A special column family is *time stamp.* For HBase, the smallest accessible unit is *cell*, it is identified and located with a combination of *row key, column(family + label), time stamp.* An example of this data model is described in Table 2.3.

Row Key	Time Stamp	ColumnFamily contents	ColumnFamily anchor
com.cnn.www	t9		anchor:cnnsi.com = ”CNN”
com.cnn.www	t8		anchor:my.look.ca = ”CNN.com”
com.cnn.www	t6	contents:html =	
com.cnn.www	t5	contents:html =	
com.cnn.www	t3	contents:html =	

Table 2.3: Data model of Hbase

There are other similar Cloud data systems, such as Amazon Dynamo, Apache Cassandra, Yahoo! PNUTS, etc. These systems deploy the similar key-value storage architecture. The main difference between these systems is the row keys and values (or contents) stored in columns. The detail of other systems will not be illustrated in this thesis, in the following text, Hbase is taken as an example to analyze the security control mechanisms.

HBase is designed as an open sourced version of Bigtable and Bigtable is initially used by Google internally. Considering the application scenarios that processing big data and the fact that Hbase is still under development, security control mechanisms are not as mature as in traditional RDBMS. But some basic measures are deployed. In the HBase documentation [Apa14], it describes several mechanisms including secure authentication, ACL, visibility label, TDE.

Authentication options include two protocol, Kerberos and Simple Authentication and Security Layer (SASL). Kerberos is a computer network authentication protocol. It deploys the idea of ticket or certificate to allow nodes communication over a non-secure network. Different nodes can prove their identity to others in a secure manner [wik14a]. SASL is a framework for identity authentication and data security in Internet protocols. SASL is provided only in version newer than version 0.92.

ACL has been discussed above in Section 2.2.2. HBase implementation approximates current convention to adjust to the feature that in HBase, create a new record and update a record is the same operation. The newest version to read is identified by time stamp.

The visibility Label mechanism could be considered as approximating MAC. This feature provides cell level security. What is new in HBase is that label is not simply numbered and then used to grant access based on the simple comparison. In the label set, logical expressions '&', '|' and '!' can be used combined with labels. An example in [Apa14] shows the possible use:

> Consider the label set confidential, secret, topsecret, probationary ,... If a cell is stored with this visibility expression: (secret |topsecret)!probationary). Then any user associated with the secret or topsecret label will be able to view the cell, as long as the user is not also associated with the probationary label.

TDE is also enabled in HBase. HBase provides transparent server side encryption to protect HFile and WAL data at rest, using a two-tier key architecture for flexible and non-intrusive key rotation [Apa14]. The first tier of key architecture is the cluster master key, which is used to encrypt the encryption keys of HBase schema and column families. The second tier of key architecture is the encryption keys that deployed on HBase table schema level or column families. After the encryption process, two attributes are added to the column descriptor. One attribute value specifies the encryption algorithm deployed, currently supported algorithm is only AES. The other attributes value is the encryption key wrapped with cluster master key. The key management is integrated with the Java KeyStore API. To deploy other key management systems is also possible in HBase. Details of the key management process can refer to [Apa14].

HBase has provided a fairly complete security control system, other systems provide similar mechanisms. Users can deploy third party data management tools like IBM infosphere to gain these features as well.

This kind of storages is widely used in the cloud, due to its nature of flexibility and Scalability. In many situations, non-relational model is considered as Nosql. But Nosql does not mean that SQL is not supported, it should be considered as Not-Only-SQL [Sto10]

Despite of all these security mechanisms, services and data in the cloud is not risk free. These mechanisms may have a good effect on protect against outside attacks, they are still vulnerable against inside attacks. For example, because of the nature of the server side encryption mechanism, keys have to be stored in the memory. Possible inside-attackers will be able to retrieve these keys and get access to sensitive data. Another problem is the trust on CSP. This kind of storage faces same problems with relational data stored in the cloud.

2.3 Cloud Security Challenges

AS it is specified above, moving to the cloud and deploying Cloud services is convenient from the perspective of customers. Together with other benefits, moving to the cloud is becoming a tendency. Customers do not need to build their own IT infrastructures, this is normally complex and time consuming. The brand new price model of pay-per-use has got a lot of attentions, it will significantly reduce the costs and at the same time avoid enormous investment into IT resources before the market prospects is becoming clear. With the flexibility and potentiality of saving cost, the idea of the cloud attracts also more developers and innovation software companies to join in this new market. From the perspective of software developers and companies, developing and deploying software services on the cloud platform has a better protection of software copyright. Through cloud services, software developers have a much close connection with customers. This in return will produce higher revenue for the software developers. Despite of all these

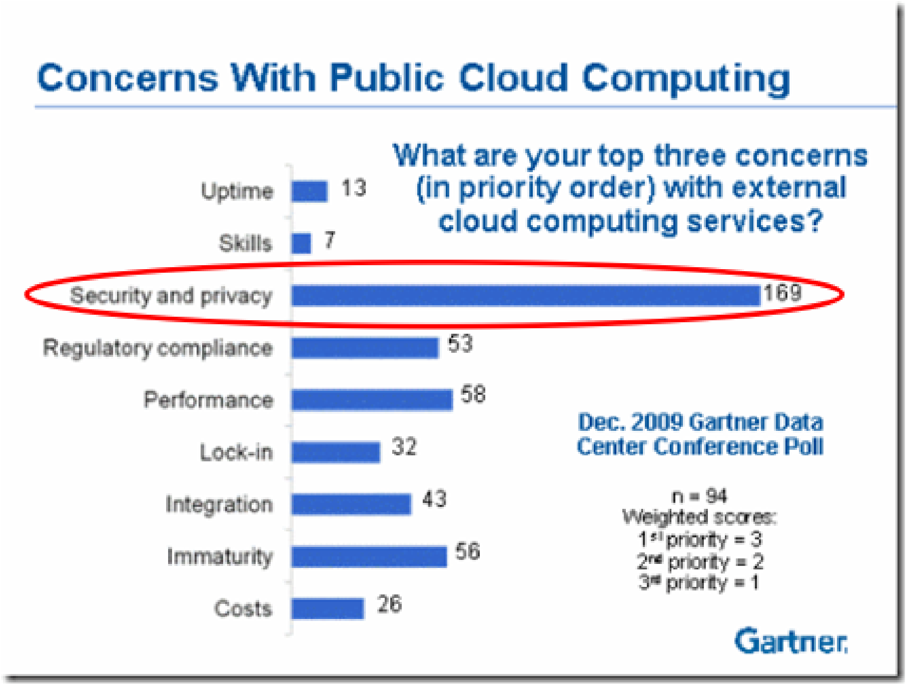


Figure 2.9: Survey of major concerns deploying external Cloud services

benefits, there are also obstacles for users to deploy Cloud services. The Cloud is still under development. The immaturity of the cloud raises concerns of the sustainability of Cloud services. Once users are already moved to the cloud and the cloud service provider they deployed gets into operational difficulties, it is difficult to recover to full functional by rebuilding their own IT infrastructure or moving to other CSPs. The main reason of this concern is that, the cloud standard is not set up. Data models and APIs depend on CSPs. Building their own Infrastructure or moving to other CSPs may be time and monetary consuming and hence it is not acceptable.

Another result of lacking interoperability is CSP dependency. This makes it harder to deploy the most optimal solution by integrating various best approaches. CSPs provides services usually in a standard form. Choices of solutions are limited to choose to achieve the best result.

Among the barriers of deploying Cloud services, a survey from Gartner [AAW13]in Figure 2.9 shows which is considered by customers as the top concern. It is clear that the security and privacy is the top barrier deploying Cloud computing services. The Cloud system complexity brings new challenges to comply with security and privacy principles. Main security and privacy concerns about the cloud include:

- Loss data access control: Because of the great cost-effectiveness, attacks that aiming at CSPs are supposed to be more frequent. The great value of data stored in the cloud may appeal attacker from both inside and outside. Data stored in the cloud faces great risks against these attacks, because once data is stolen or malicious altered, value loss is not evaluable.

- Inadequate data deletion: Because of data replication and distribution in Cloud, to make sure that all the data replicas are convinced deleted and not recoverable is difficult. Another situation is that users want to recover the accidentally deleted data. These two situations are in contradiction and hence it is becoming more difficult for cloud service design.

- Data backup vulnerabilities: Comparing to original data, data backup is normally in lower level protected. Making data replication to provide better protection against storage device failures or disasters also makes it vulnerable again attackers.

- Isolation failure: Users share the same IT infrastructure with each other, once the isolation is failed, data may leaked to the other users. Attackers can pretend to be normal users to gain privilege in other application systems by using isolation vulnerabilities.

- loss of transparency: As the basic infrastructure of the cloud may be clear to users or researchers, the detail of the cloud architecture remains unclear. For different CSPs, service design architectures are also different. When a VM or a database instance is assigned to a user, it is not transparent where the physical host is located, how the VM is migrated and recovered, etc.

The above discussion shows the general concerns of privacy and security in the cloud. Detail privacy challenges in the cloud will be introduced in section 3.2.

2.4 Data Life Cycle Management

Information management is a pretty mature area in the computer science. Information is stored as written material or recorded on tapes, these two approaches still play an important role in our society. But more generally, information is represented as digital data to transfer and store. Information demands of a full life control. As information is specified as data in this paper, we will use the term data life cycle management instead of information life cycle management. In the discussion of privacy preserving approaches in the cloud environment we focus in this thesis, data life cycle management works as a reference to have a better understanding of privacy control process in the cloud. It is also a foundation of how to classify privacy preserving mechanisms in the conclusion chapter. These mechanisms are not just specified or designed to be implanted in a particular

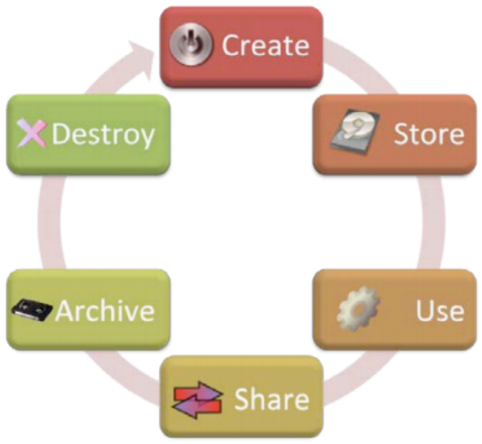


Figure 2.10: Data life cycle management from [BM+11]

phase, they may be deployed in several phases due to mechanism characteristics. In Figure 2-10, the data life circle management process is stated. The original description are listed as follows [BM+11]:

- Create. Creation is the generation of new digital content, or the alteration/updating/modifying of existing content.
- Store. Storing is the act committing the digital data to some sort of storage repository and typically occurs nearly simultaneously with creation.
- Use. Data is viewed, processed, or otherwise used in some sort of activity, not including modification.
- Share. Information is made accessible to others, such as between users, to customers, and to partners.
- Archive. Data leaves active use and enters long-term storage.
- Destroy. Data is permanently destroyed using physical or digital means (e.g.,cryptoshredding)

When we map this life cycle management graph to the privacy data management in the cloud, the meaning of each phase is a little different. Each phase of privacy data life cycle in the cloud is specified as follows:

- Create: Data is prepared to be outsourced and stored in the cloud. Detail processes include properly classification of privacy data, preprocessing of privacy data, deciding the control policy for data sets.

- Storage: Users purchase data storage capacity and transfer data to Cloud storage. Data storage should take the backup strategy into consideration. If it is not automatically controlled by CSP and defined by the customers, customers should have a strategy to back up their data. Data transfer should also consider the corresponding security control mechanisms. For example, if large data sets need to be transferred, transfer over internet or transfer hard disk per post should be decided according to the time costs and other factors. Another work to do is the migration of the old data model to Cloud storage model.

- Use: Users get access to data over Internet. This is similar to getting access to local stored data, querying data and the management of data. The latency due to internet connection should be concerned during use phase. Also what is different is the access control mechanism. Google storage service provides ACL based object control; the use of data is through JSON or XML APIs. But this is not suitable when processing traditional relational data. Google Cloud SQL provides the same security control of its own application and Cloud data from users. The detail is not published and then essential trust on Google is needed.

- Share: Users share sensitive data with others. In most of the researches, this phase is similar to the data publishing or data recreation. Privacy preserving mechanisms in this phase are similar to data creation phase. In the future discussion, the classification of mechanisms in Create and Share phase is distinguished. For example, sharing privacy data means that decryption of encrypted data, and then securely transferred to third party. The third party may deploy another mechanism to protect the privacy, which is similar to mechanisms deployed in data Create phase.

- Archive: Data is stored as zipped file or other formats and then transferred to long term storage. For Cloud users, the general purpose in this phase is to back up the data. A local archive system moved to the cloud is implanted as a mirror, for example DNAnexus use Google Cloud storage to get a comprehensive Cloud-based DNA archive. [gooa]

- Destroy: Data is securely deleted from Cloud. The focus of this phase is to make sure that no information could be recovered and accessed by any means. The major difficulty is how to convince users that data is properly post-processed. This goal can either be achieved by technical solutions or regulation solutions. Technical solutions is for example, protocol based rewrite over deleted object. Google proposed that, once the data is deleted, the corresponding disk sector is only write operation granted; the read command of this area is banned. Another solution is the self-encryption drives, once it is power off, it is locked and encrypted automatically [Mol]. But this may be not practical, because the cloud data is multi-times duplicated, and the locations are not predictable. To enable this feature, CSPs have to deploy all this kinds of drivers in their data centers and the cost may be not acceptable.

Chapter 3

Privacy Issue

This chapter discusses the privacy issues. At first, the definition of privacy is given and the basic privacy protection principles are listed and explained. Then the challenges of privacy in the cloud are discussed and the general privacy preservation mechanisms are introduced and classified.

3.1 Privacy Definition and Protection

Privacy is a broad concept in both legislations and the daily life. Privacy in the daily life means the fundamental human right that the personal spaces will not be intruded. By personal space we mean that people have some spaces beyond the observation of third parties. For example, people do not want to be observed when using the bathroom or doing private and sensitive activities. The improper government surveillance are seen as part of the privacy threats. This kind of privacy could be named or classified as physical privacy.

Another kind of privacy is the information privacy or soft privacy. A normal citizen has various types of privacy information. The basic identification information is stored in a specific department of the country; the other information, like health records and tax records are also stored in the correspondent departments of the country. Normally we assume that privacy information stored in these official organizations are properly protected and will not be misused. Besides these essential information that are allowed by legislations to be collected and stored, other privacy information are not allowed to be collected and stored even by the official organizations. In the daily life, for example, when we register to a social network website, we have to read and sign a privacy policy agreement from this website to allow it to collect and process our privacy data.

In different lands or regions, privacy information have varies definitions and protection requirements. The terms they use that connected with privacy may be different, EU use a similar concept personal data as privacy information we talked above. The current European Union (EU) definition of personal data is that [PY13]:

> "personal" data shall mean any information relating to an identified or identifiable natural person ("data subject"); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological,

> mental, economic, cultural or social identity;

Another similar concept is Personal Identifiable Information (PII). This term is generally used by USA. It is defined as follows [PY13]:

> information that can be traced to a particular individual and include such things as name, address, phone number, social security or national identity number, credit card number, email address, passwords and date of birth.

Though terms used in different areas or countries may vary, the actual meaning (scope) of the privacy is almost the same. The privacy definition of USA is a general concept without exact classifications. Privacy definition from EU has a much better scope statement. In general, personal privacy information is categorized under following categories:

- Physical: The height and weight information, healthy information, address etc.
- Physiological: Gender, age, DNA, blood type etc.
- mental: Most of this category is academic achievements or test scores.
- Economic: Financial information including tax records, salary, credit, debt etc.
- Cultural or social identity: Nationality, religion, political preference etc.

Most privacy information is nowadays digitized. According to the difference of storage formats, privacy information could be categorized into two categories:

- Structured data: Data are stored in different types of data management systems. This category is corresponding to relational data in the cloud.
- Unstructured data: These data may be images, video or audio files, or simple formatted document file. This category is corresponding to simple object classification in the cloud.

Storage and processing of privacy information has to comply with laws, regulations and rules. Depends on the region and location difference, these laws and regulations also vary. In EU, the most important law is European Union (EU) Directive 95/46/EC. It sets up a regulatory framework which seeks to strike a balance between a high level of protection for the privacy of individuals and the free movement of personal data within the European Union [DAT95]. This framework have some main principles list as follows [PY13]:

- Data collection limitation: Data should be collected legally with the consent of the data subject where appropriate and should be limited to the data that is needed.
- Data quality: Data should be relevant and kept accurate.
- Purpose specification: The purpose should be stated at the time of data collection.
- Use limitation: Personal data should not be used for other purposes unless with the consent of the individuals.

- Security: Personal data should be protected with a reasonable degree of security.
- Openness: Individuals should be able to find out what personal data is held and how it is used by an organization.
- Individual participation: Individuals should be able to obtain details of all information about them that held by a data controller and challenge it if incorrect.
- Accountability: The data controller should be accountable for complying with these principles.

These principles are also widely deployed by other countries and regions. Beside these main principles, EU has also adopted data transfer restrictions in this framework. USA has similar restrictions on data transfer, they are specified in the Safe Harbor agreement. By adding other restrictions, other countries and regions can build their own privacy regulation system to match with their particular requirements.

Although these principles should be adopt in the practice of privacy information management, because of the compliance complexity and the lacking of experts, privacy violations are often occurred. For example, many websites request users to register and then submit their profile. Connected with the Internet Protocol (IP) address and user browsing records, websites are able to analyze the user habits and other preferences. Form the early stage of direct marketing and telemarketing to nowadays data mining aided marketing, websites and other business institutions are getting a better market aiming strategy. Since the great development of online social networks, more and more privacy data are now being collected and analyzed. These data are also exchanged and circulated among websites, consultancy and advisory companies, marketing companies and so on. Because of their potential profit value, these data are also the target of many attackers. Once this privacy information is leaked, the possible harmful result may range from spam mail to online identity theft and even possible finical loss or even criminal damage.

3.2 Privacy Challenges and Privacy Preservation Mechanisms Overview

Traditional privacy protections face great challenges in practice. The tendency that more and more organizations and institutions are moving to the cloud brings new challenges to the traditional privacy protection framework. Ali Gholami, Ake Edlund and Erwin Laure promoted Cloud privacy threat modelling to enforce privacy requirements of EU Directive 95/46/EC on Data Protection [GEL]. Using the classification of this model as a reference, Cloud privacy threats are described according to two categories: regulation challenges and technical challenges.

Regulation Challenges

The first category of challenges is the regulation compliance complexity. Because of the cloud nature, privacy data is duplicated in multiple data centers. Those data centers are located in different areas or lands. As we discussed above, various privacy definitions and scopes also bring up various regulations. Sensitive data needs different protection requirements in each location. Another problem is the data flow restriction; this is a part of the regulation complexity. This restriction should not only be considered when transferring privacy data to CSP, it needs to be taken into account during the data duplications. For example, when transferring privacy data among all countries with national privacy protection legislation, which include EU and European Economic Area (EEA) countries, Argentina, Australia, Canada, Hong Kong and New Zealand, data transfer restriction must be complied with. From EU/EEA countries, personal information can be transferred to countries that have "adequate protection", namely, all other EU/EEA member states and also Switzerland, Canada, Argentina and Israel (since all have regulations deemed adequate by the EU) [PY13]. When choosing CSPs, privacy data from these countries and areas should also consider if their Cloud data centers are allocated in proper geographical locations. If this requirement is not satisfied, there are also alternatives to choose. One such mechanism is that information can be transferred from an EU country to the USA if the receiving entity has joined the US Safe Harbor agreement [PY13]. The problem caused by the duplication of privacy data seems to be easy to solve. Some CSPs provide the option for user to choose Cloud data center locations, and the duplication of privacy data are also stored under users' restrictions. But the choice is limited, because some CSPs may only have one data center in this area, and the location do not comply with the data transfer restriction. Another possible situation is that, data storage location complies with the restriction, but the essential backup locations do not comply. The purpose of backup is to provide better availability, performance and disaster recovery, and the backup strategy is initially designed and it is impossible or difficult to change. Customers have to decide the balance between potential violation of regulations and privacy data security according to their individual requirements.

Another challenge has strong connection with the regulations compliance complexity. The main reason of this challenge is losing data control. If privacy data are stored in data centers located in USA, privacy information could face great threats. The Patriot Act, a US federal law that can compel the legal request of customer and employee privacy information, causes fears about transferring information to the USA particularly [PY13]. This law allows the US government to get accesses to privacy data without informing the privacy data owner. And we can see from the report that the patriot law is abused [Sen]. Another situation is that: a CSP may be forced to hand over data stored in the cloud. As it is illustrated by the US vs. Weaver case, where Microsoft was requested via a trial subpoena rather than a warrant to provide emails transferred by their Hotmail service [PY13]. What is more terrifying is the National security Agency (NSA) scandal [Mey]. It is reported that NSA has been continually collection privacy data over internet. There was a list showing IT companies that involved in NSA spy project. In the list, we can see some familiar names, Google, Microsoft, IBM and Amazon. They are the most valued IT companies which have in almost all the IT areas significant market share, also in the cloud service market. Without the cooperation of those companies, NSA secretly adds back-doors to IT hardware and software. All of these examples and evidences make it hard to trust the CSP and then deploying Cloud services.

There are no perfect countermeasures to solve these two challenges. When customers decide to deploy Cloud services, they have to carefully choose the CSP and make sure to address these challenges in Service Level Agreement (SLA).

Technical Challenges and Protection Mechanisms

Among those principles described in the EU privacy regulatory framework, which are mentioned in 3.1, Data collection limitation and Data quality principle remains not challenged when deploying Cloud services. To follow these two principles is still the responsibility of the cloud customer, who wants to deploy Cloud services. The other principles are challenged due to loss control of privacy data, multi-tenancy, complex Cloud architecture and so on. The violations of the other privacy preservation principles are summarized as three categories of challenges, the trust and transparency challenges, the information disclosure challenges and security challenges.

Once data is transferred to and stored in the cloud, as the CSP has the root control of the entire Cloud infrastructure, theoretically CSP could do anything with the data stored in the cloud. Considering this concern and the potential possibility of making profit of privacy data, it is highly possible that inside attacks happen. Privacy data may be stolen and sold by malicious Cloud administrators. A worse situation is that CSP may make improper secondary use of data stored in the cloud without permissions from data owners. In both situation, privacy preservation principle of keeping initial purpose of privacy data collected is tampered, the principle of purpose specification and privacy data use limitation are also violated. The difficulty to identify the ownership of the privacy data makes this problem much more complex.

Although CSP like Google described their approaches [goob] to protect IT security and try to convince customers that data is properly protected in the cloud, inside attack problems remain unsolved. In the white paper of Google's Approach to IT Security [goob], several standard protection mechanisms are described. These mechanisms mainly aim at providing physical security and disaster recovery ability. To prevent inside attacks and thefts, mechanisms that Google employs are strict access control policies and other management policies. For the cloud user, it is not sufficient to build trust on the morality of companies and employees. The abuse of laws and regulations from law enforcements and the possible cooperation between CSPs and other official departments reinforces the distrust on CSPs.

Current standard solution to solve these problems is to address the data privacy preservation in SLA. But the fact that, CSPs are responsible for the enforcement of SLA. This does not necessarily improve the trust on CSPs. When privacy protection violation happens, CSPs are highly motivated to hide the evidences and conceal the accidents due to economic benefits. Because of lacking transparency, for the normal Cloud users, it is neither practical nor available to monitor the privacy data processing in the cloud. CSPs may not be willing to allow user to monitor the complete processing procedure regarding of security controls and business secrets.

All of those issues discussed above are just parts of the trust and transparency challenges. Cloud audit has been proposed to solve the transparency problem [Gro]. A step further, third party auditing [SBM$^+$07] and public auditing are promoted to address this problem [WRLL10]. Another approach is deploy the idea of trust computing [Pea02]. By providing VM level [GPC$^+$03] and computing pool level security control [SGR09], Cloud users could get better control over the computing environment. Hence the trust of the cloud could be improved.

The second privacy technical challenge exists before deploying Cloud services. It is information disclosure challenges. In Cloud privacy threat modelling, this challenge is also named as data minimisation. Before storing data in the cloud or publishing data to the public, essential preprocessing of data is deployed. The main purpose of these procedures is to reduce the risk of privacy leakages. Data minimisation including anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management [PH10]. Researchers have found that, removing the identity is not enough to preserve privacy. By connecting with background information and even alone with the combination of published attributes, privacy information could be recovered. To solve this problem, anonymity base mechanisms, such as k-anonymity [SS98b], [Swe02a], [Swe02b] , [EED08] has been promoted. The definition of k-anonymity and related quasi-identifier are specified as follows:
Quasi identifier definition [Swe02b]:

> Given a population of entities $\mathbf{U}$, an entity-specific table $T(A_1, A_2, \ldots, A_n)$, $f_c : \mathbf{U} \rightarrow \mathbf{T}$ and $f_g : \mathbf{T} \rightarrow \mathbf{U'}$ where $\mathbf{U} \subseteq \mathbf{U'}$. A quasi-identifier of $\mathbf{T}$, written Q_T, is a set of attributes $\{A_i, \ldots, A_j\} \subseteq \{A_1, \ldots, A_n\}$ where $\exists p_i \in \mathbf{U}$ such that $f_g(f_c(p_i[Q_T])) = p_i$.

k-anonymity definition [Swe02b]:

> Let $\mathbf{RT}(A_1, \ldots, A_n)$ be a table and QI_{RT} be the quasi-identifier associated with it. $\mathbf{RT}$ is said to satisfy k -anonymity if and only if each sequence of values in $\mathbf{RT}[QI_{RT}]$ appears with at least k occurrences in $\mathbf{RT}[QI_{RT}]$.

Since normal database tables do not preserve the k-anonymity property, the general approach to achieve k-anonymity is generalizing (for record linkage mitigation) and suppressing [SS98a]. Generalizing data means attribute values are only specified as in certain range, accurate values are not stored in the table. For example, salary value could be stored as $25 * *$ instead of 2535. Suppressing data means related value are completely deleted from the table, which will be outsourced or published.

Furthermore, based on the analysis of the disadvantages and possible threats, i-diversity[LLV07],(for record and attribute linkage mitigation), t-closeness [LLV07] (for probabilistic attacks mitigation) and differential privacy [YL12](for table linkage mitigation) are promoted to prevent privacy leakage from different types of attacks. The Differential privacy ensures data protection through a negligible difference between prior and posterior adversary's knowledge against a participating data [GEL]. paper [SZ09] has summarized the privacy preservation information disclosure approaches.

But since this kind of mechanism demands data pre-processing before storing data in the cloud, it is used in several specific areas, such as data mining and data publishing. Considering of its limited use in the cloud, detail discussion will not be addressed in this thesis.

The last challenge deploying Cloud is security. Security has four aspects, they are listed as follows [Jan]:

- Confidentiality: Keeping data and resources hidden
- Integrity: Data integrity
- Origin integrity: Data origin authenticity and Entity authenticity. These two terms refer to identity management and access control.
- Availability: Enabling access to data and resources

When deploying Cloud services, there is no guarantee for security and privacy concerns such as confidentiality, integrity and accountability in their Service Level Agreements (SLAs) [GEL]. One of the most attractive benefits provided by CSP is that high availability of Cloud services. The fact is not as optimistic as it should be.

- Amazon S3 service twice crashed in February and July 2009. Reason: overload of user requests.
- Amazon data center, large area crash happened at April 22th 2011.
- Microsoft Cloud platform Azure crash at March 17th 2009,
- Microsoft Business Productivity Online Suite (BPOS) service breakdown in September 2010
- Rackspace Cloud service breakdown in June and November 2009,
- Salesforce.com service breakdown in January 2010,
- Terremark service breakdown in March 2010
- Intuit online financial accounting and develop service breakdown in June 2010.

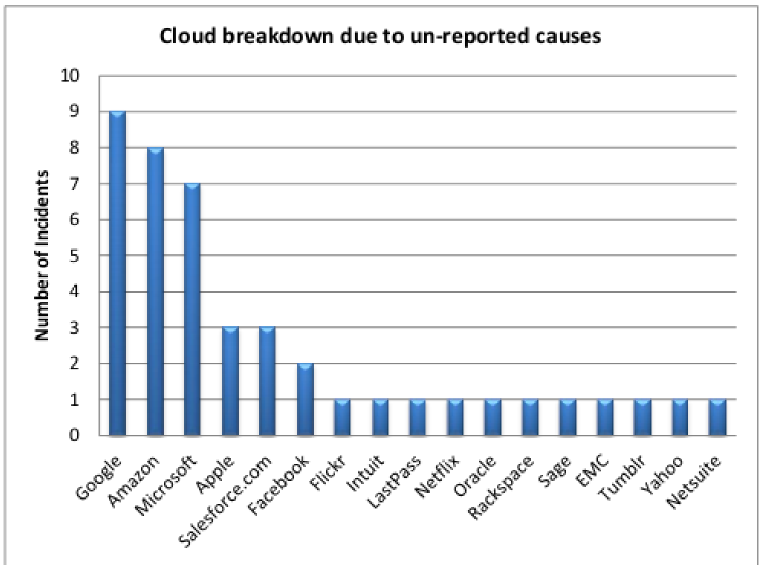


Figure 3.1: Number of incidents reported by Cloud service providers [CSA13]

The accidents survey showed in Figure 3.1 are availability accidents due to CSP failures or unidentified causes. Availability problem is not the research focus of this thesis. Most of the security researches in the cloud focus on the confidentiality and access control of privacy data. They are direct connected with the privacy preservation in the cloud. The above availability accidents act as evidences to demonstrate that security faces great challenges in the cloud.

For other aspects of security, the data security threats analysis in [GEL] focuses on data confidentiality and integrity. Possible attacks may be run-time memory access by malicious insiders of a Cloud infrastructure, eavesdropping, side-channel attacks, elevation of privileges, and transmission of personal data out of allowed area [GEL].

Based on the security attributes, a threat modeling is given in [XX13]. In this Cloud security and privacy threat model, security aspects are not exactly the same as they are defined in [Jan]. Accountability is considered instead of origin integrity. Accountability threats in [XX13] are summarized as: SLA violation, inaccurate billing of resource consumption, etc.

In this thesis, accountability threats are considered as trust challenges. The defense strategies promoted in [XX13] also deploy the idea of the cloud audit and trusted computing. Since the focus of this thesis is privacy and privacy is tight connected with data confidentiality and integrity, following Cloud threats are summarized based on the analysis described in [XX13].

- Confidentiality concerns: Cross-VM attacks via side channels and malicious sysadmin. Cross-VM attacks could be performed by inside or outside attackers. The major privacy concerns of this thesis come from malicious sysadmin.

- Integrity concerns: Data loss or data manipulation. Data loss could be partly solved via data backups and data duplication, this is not the major privacy concern in this thesis. Data manipulation could be considered as the data access control concern, which refers to origin integrity defined in [Jan]. As access control mechanisms are deeply researched and mature science, they are implanted by the cloud (please see the discussion in 2.2). Based on former experiences, we assume that, after correct implementation and configuration, privacy data access violations from outside could be prevented. The major concern comes back to inside attacks, which malicious sysadmin is a great threat.

To solve data confidentiality and integrity problems discussed above, the general approach is to encrypt sensitive data before store in the cloud [AEAW12], [NS13]. Based on the analysis of deploying different types of encryption, onion encryption is promoted to achieve better performance [AEKR]. Related topics, query on encrypted SQL data and encrypted based access control are described in [NGSN10], [MKL09], [HILM02].

Besides the encryption approaches, a novel approach is to protect privacy data by data distribution. This means splitting data into small data groups, only contents are stored in the cloud storage, the control information or data object arrangement information is stored locally or on trusted third party. Even malicious attackers could get access to data split pieces, without the reconstruction information, the reconstruction process is time consuming and hence these data may be meaningless. Since CSP or attackers only knows meaningless data splits, data confidentiality is preserved and privacy could be preserved.

Among these four major classifications of privacy preservation mechanisms: trust based approaches, encryption based approaches and data distribution based approaches will be detail described and analyzed in the next separating chapters. Anonymity based approaches are not addressed due to their limited application scenarios.

Chapter 4

Encryption-based Approaches

Encryption of sensitive data is a natural and standard approach to preserve privacy in Cloud. Encryption based approaches are classified into two general types: client side encryption, server side encryption. For each type of encryption, a survey result and typical examples are given according to a simple classification of simple object storage and relational, semi-relational (non-relational in chapter 2) data storage in the cloud. At each end of discussion of two encryption types, the analysis of advantages and disadvantages from four aspects will be given. Four aspects are: 1. Performance 2. Technological maturity and complexity 3. CSP support 4. System and application modification.

4.1 Client-side Encryption

For client side encryption, approaches are described according to the former data categorization, but relational and semi-relational data will be discussed together. That is simple object storage and relational data storage.

4.1.1 Simple Storage Client-side Encryption

For simple data object storage in the cloud, there are mainly two approaches deployed. The first approach is Using encryption API provided by CSP, for example, amazon S3 storage service provides client side encryption. The second approach is using local encryption system or third party encryption software.

When using Amazon S3, this function is provided in AWS SDK, available programming languages are Java and Ruby. The encryption process when using Java in AWS SDK is called *envelope encryption.* Based on the introduction of [Ama14b], the encryption process is described as follows:

1. The encryption client generates a one-time-use symmetric key, it is called envelope symmetric key. This key is used to encrypt data locally before uploading data to Cloud.

2. The Assignment of a unique encryption key. This key is could be symmetric key or asymmetric key (public/private key). It is used to encrypt the envelope symmetric

key. This key should be properly stored and managed, backup should be made in case of losing or storage media damage.

3. Data is encrypted with the envelope symmetric key. Encrypted data and the encrypted envelope symmetric key are uploaded to Cloud. The encrypted envelope symmetric key is stored as part of the object metadata, called x-amz-meta-x-amz-key.

Decryption of data stored in S3 is the reversed process. At first, user retrieves encrypted data and encrypted envelope key together from Cloud. Then encrypted envelope symmetric key is decrypted using private key that managed by user himself. At last, encrypted data could be decrypted using envelope symmetric key. As we can see, the client encryption using AWS SDK is a typical two-tier system. Private key assigned by the user acts as a master key and envelope symmetric keys act as slave keys. This will save great efforts for users to manage large amounts of encryption keys. If users want to use a third party encryption process, it is also supported by S3. The detail of configuration could be found in [Ama14b].

Client side encryption APIs are not yet supported by Google Drive and Microsoft SkyDrive. Users should encrypt sensitive data using third party tools. Following discussion will be focused on using third party tools. There are many available choices for encrypting data locally. Depending on the sensitivity of data, users could simply add a password to protect sensitive files or use an encryption software. Media has promoted the five most useful encryption tools for cloud storage, they are: Viivo, Sookasa, DataLocker, Boxcryptor, Cloudfogger [enc14]. These five tools are described as follows:

- Viivo: It is developed by PKWARE. The encryption algorithm uses AES-256, and the encryption key is generated by using a secure hash function (PBKDF2 HMAC SHA256) on users passwords, which are assigned when the user accounts are registered [vii14]. All files are encrypted using the same encryption key. Viivo is easy to use, but it has a downside that it only supports Dropbox.

- Sookasa: It is designed for enterprise use. The encryption algorithm also uses AES-256. Besides the normal encryption and decryption function, Sookasa provides an additional security feature that users could set key expiration times. This feature provides a better key security management strategy. Similar to Viivo, Sookasa supports only Dropbox, which limits its use.

- DataLocker: It is another all-in-one encryption tool which allows users to easily encrypt sensitive information in Dropbox.

- Boxcryptor: It creates a cryptographic virtual hard disk. Once the data is moved to this virtual hard disk, it is automatically encrypted using the AES-256 standard. Comparing to the other three tools, Boxcryptor supports an easier management of sensitive data stored on Dropbox, SkyDrive, or Google Drive.

- Cloudfogger: It works the same way as Boxcryptor and Viivo. Users could assign a specified virtual file folder to automatically encrypt data in it, or encrypt a file manually using a corresponding command.

As it is described above, third party encryption tools have a similar encryption key management system. User passwords are used as a master key, and the encryption key (slave key) is generated by the master key. Compared to the encryption API provided in Amazon S3, there is a major drawback that the encryption key is permanent and all the files are encrypted using the same key in these tools, while encryption keys are one-time-use when using an AWS SDK envelope encryption process.

No matter using the APIs provides by CSPs or deploying tools described above, the general process of data encryption management is similar. The data life cycle of client side encrypted data is described as follows:

- Create: Encrypting data objects which are needed to be stored in the cloud. When using APIs provided by CSPs, users need to assign a master key. When using third party tools, users need to login and move the data to an encrypted file folder.
- Storage: When using client side APIs, users upload encrypted data and the encrypted envelope key together to the Cloud. The essential control policies for the data, such as ACL, bucket policies are also uploaded to the cloud. When using third party tools, this task is performed by them automatically. File folder synchronization is usually used during the upload process.
- Use: Encrypted data and additional metadata are downloaded from the Cloud. Users could decrypt the data manually or they are automatically decrypted by using third party tools.
- Share: Sharing encrypted data on the Cloud is not encouraged. Users have to send a corresponding master key to other users, which is not practical in most cases. The naive solution would be decrypting data locally and then send it to the other users. This solution works well only when a small amount of data needs to be shared.
- Archive: Archive is not the main purpose to deploy Cloud services. Simple storage on the Cloud normally acts as an archive system.
- Delete: Deleting data from the Cloud could be done by using a standard function provided by a service provider. When using third party tools, users need to delete data manually. The major problem with data deletion in the cloud is how to make sure that the deletion is successfully performed and no backups and duplicates remain in the cloud. This problem belongs to the trust challenges discussed in 3.2. While data is encrypted in the cloud, this problem is not the major concern of Cloud users. Encrypted data will not leak user privacy as long as it is not decrypted, which we assume that is reasonable.

According to the discussion above, the analysis of the client side encryption is described as follows:

- Performance: While data encryption and decryption processes cause nly little performance loss, encrypted data management becomes a major obstacle when using Cloud storage. When only a limited amount of encrypted files are stored on the Cloud, users could perform the management tasks manually. When the amount of encrypted files increases and the relation between these files getting complex, retrieving needed data among them becomes difficult. Current approach is assigning keywords to each file and storing encrypted files and keywords together on the Cloud. When searching for needed files, users could perform keyword searches on the cloud. Detail approaches are discussed in 4.3.

- Technological maturity and complexity: Encryption algorithms have been widely and deeply researched in the past. Corresponding standards and technologies are mature to use. While encryption algorithms are mature to use in third party tools and Cloud APIs, the related key management is not addressed in third party tools. The solution that encrypting all files using the same encryption key could potentially causes security weaknesses. This is a conflict between convenience and security assurance.

- CSP support: When using third party encryption tools, CSP support is not necessary. When using a client APIs, CSP support is essential. While not all CSPs provide similar APIs like Amazon S3, lacking CSP support may limit the utilization of other Cloud platforms.

- System or application modification: Applications implemented in local system do not need modifications, because encrypted data are decrypted locally and applications could perform operations on the decrypted data.

4.1.2 Relational and NoSQL Data Client-side Encryption

There are three approaches of client side encryption discussed in this section: 1. Encrypting data and storing it as attributes value. 2. Using local encryption engine. 3. Database re-encryption in the cloud. Because the architecture of the three approaches is relatively different, analysis of these approaches will be done separately.

The first approach is encrypting data and then storing the encrypted data as attribute values. The Encrypted data has no data types. It is a string of binary numbers. To store encrypted data as attribute values, the available attribute type for current database is *blob*. *Blob* is a binary large object that can hold a variable amount of data. This approach is highly limited. The data life cycle is similar to the simple object storage client-side encryption. Data has to be encrypted locally and then stored in the database. The processing of encrypted data requires users to retrieve it from a remote database at first. Unlike querying over encrypted data directly in Cloud storage, *blob* data could be retrieved by querying other attributes or keys stored together with it in the same table. For example, if a bank account of a user A is stored as encrypted data (blob type), user A could get the whole record by the following SQL command:

*select * from Table where id== '8765';*

Suppose *id* of user A is 8765, then the record could be transmitted to user A, and user A could decrypt the bank account respectively.

Another choice is using the encryption function provided by the database. For example, DB2 provide a built-in encryption/decryption function. Using built-in encryption functions is limited, only *CHAR, VARCHAR, VARCHAR FOR BIT DATA* data types could be encrypted, the encrypted result is stored as *VARCHAR FOR BIT DATA*, technical detail could refer to [IBM14].

When using these client side encryption approaches, the data life cycle is similar to a simple storage client side encryption. Sensitive data needs to be encrypted before inserting it into a database. It is decrypted after retrieving the corresponding record. The analysis of this approach is discussed as follows:

- Performance: Data encryption and decryption is complex. The data types that are available for encryption are limited. In general, this approach does not perform well in many application scenarios.
- Technology maturity or complexity: Users have to encrypt data manually. The management of the encryption keys or passwords is complex. Similar to the simple storage client encryption, indexes on encrypted data are not supported. Relational operations on encrypted attributes are also not supported. Current researches focus on performing SQL functions over encrypted data, this will be discussed in 4.3.
- CSP support: CSP support depends on the database instances that users choose. If the original database provided encryption and decryption functions, database instanced on the Cloud may provide similar functions. For example, MySQL instance in Amazon RDS provides the same function as MySQL implemented in local system [Ama14a]. But users need to check the related documentation for details when using these functions. Not all encryption features are supported on the Cloud.
- System or application modification: When applications interact with databases, if sensitive data need to be encrypted, this task should be done by the application logic.

The second approach is building a local encryption engine. Sensitive data is encrypted and stored in a Cloud database. The local encryption engine takes care of query rewriting and encryption/decryption tasks. There are several solutions promoted. The following text discusses the basic design architectures and working processes of local engines.

CloudSE [Liv] is a client database engine, which aims to provide the local encryption functions. Local database engines (cloud storage engine) take care of SQL queries and the encryption/decryption of sensitive data. Cloud storage acts as a remote file system to cloudSE. The Current version support system implantation on cloud storage provided by S3, GS (Google Cloud Storage), and Walrus (Eucalyptus Walrus). No advanced storage management functions are utilized. Remote servers store only the databases, where the sensitive data has already been encrypted. The structure of cloudSE is described in Figure 4.1: To enable this encryption feature of cloudSE,

> "The *clouse-cloud-data-encrypt-key* option can be used to specify the encryption key data in the form Algorithm:*Passphrase*. AES256 is the only algorithm that is currently supported. [Liv]".

Similar to the third party encryption tools discussed in4.1.1, *Passphrase* is the master key that is used to generate the encrypted key, which is easy for users to remember.

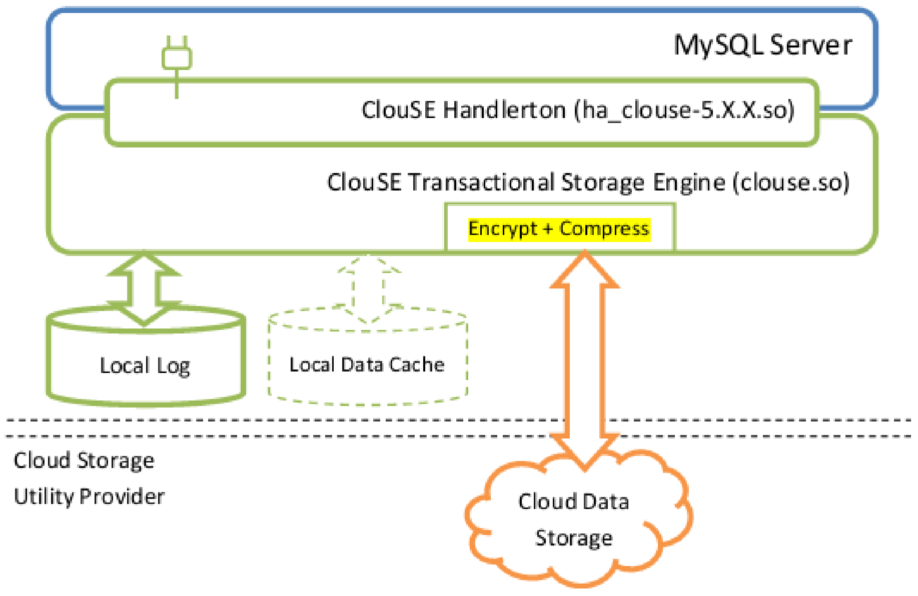


Figure 4.1: Structure design of cloudSE [Liv]

Proper backups of *Passphrase* should be done in case of key loss. There is no detailed explanation of the encryption process in cloudSE, it is reasonable to assume that the encryption granularity is at the level of database files based on the description in [Liv] and Figure4.1.

CryptDB [PRZB11] is another solution that built a web proxy as an engine to interact with applications and the database servers. The basic system architecture is described in Figure 4.2. In this architecture, a web proxy could be implemented on a trusted third

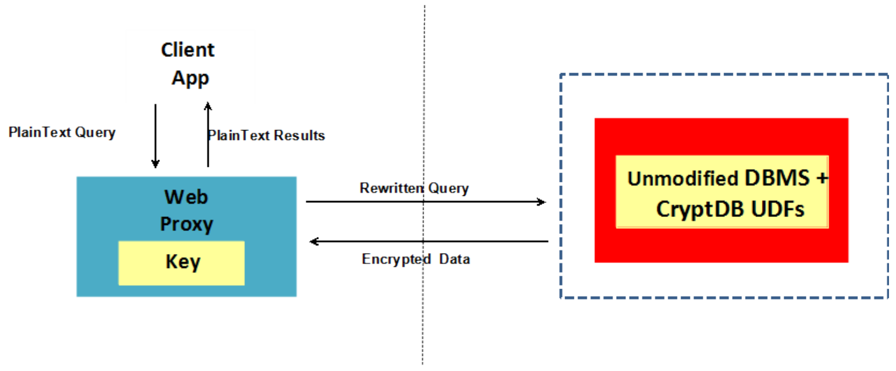


Figure 4.2: Structure design of CryptDB based on [AEKR]

party or in the local system. The main task of a web proxy is to rewrite queries and perform the final decryption of results returned from remote database servers. UDFs (User Defined Functions) are a set of encryption functions which encrypt and decrypt data dynamically to a certain level based on query types and attribute types. The general idea of CryptDB is to encrypt data using different encryption algorithms that support SQL queries over encrypted data. The detailed query process and data encryption model will be discussed in Section4.3.

An analysis of using local encryption engines is described as follows:

- Performance: In CloudSE, the encryption granularity is at the database file level. Retrieval of large tables needs to download the whole table from the Cloud. This is a significant performance bottleneck. In CryptDB, data is stored using sets of encryption algorithms, storage space overheads is an obstacle [AEKR]. In the Cloud, more storage space and more computation mean that users need to pay more. Users need to reconsider whether it still is worthy to store data on the cloud when security investment significantly increases. Besides this obstacle, CryptDB can support operations over encrypted data for 99.5% of the 128,840 columns seen in the trace of 126 million SQL queries from a production MySQL server [PRZB11]. Compared to unmodified MySQL, CryptDB has a low overhead and it reduced throughput by 14.5% for phpBB, a web forum application, and by 26% for queries from TPC-C (Benchmark for OLTP) [PRZB11].

- Technological maturity and complexity: CloudSE and CryptDB were both successfully implemented and evaluated. The main limit for CryptDB is that, not all the SQL operations are supported [AEKR]. The potential disadvantage is that, these two engines works well for workload where the amount of data shipped is small [AEKR].

- CSP support: CloudSE does not need CSP support, because Cloud storage acts as a remote file system and all the data are encrypted in an encrypted form. CryptDB needs to implement CryptDB UDFs (User Defined Functions) component in the cloud, hence Amazon RDS does not support cryptDB. When deploying IaaS service model, users could implement the database and the CryptDB component in VM.

- System or application modification: Both systems do not need application modification. For the system modification aspect, CloudSE is integrated with the MySQL engine, which means CloudSE needs to modify the MySQL engine while CryptDB needs to add additional components to the database server, but the original database does not need to be modified [PRZB11].

The third approach is Database service Provider (DSP) re-encryption, the following discussions are based on the description in [TWZ09]. The System architecture is described in Figure 4.3. Abbreviation used in this system is specified as follows:

- DO: Data Owner
- DR: Data Requester
- RE: Re-Encryption .
- DSP:Database Service Provider
- PKG: Private Key Generator
- REKG: Re-Encryption Key Generator
- ACAT: access Control Authorization Tables

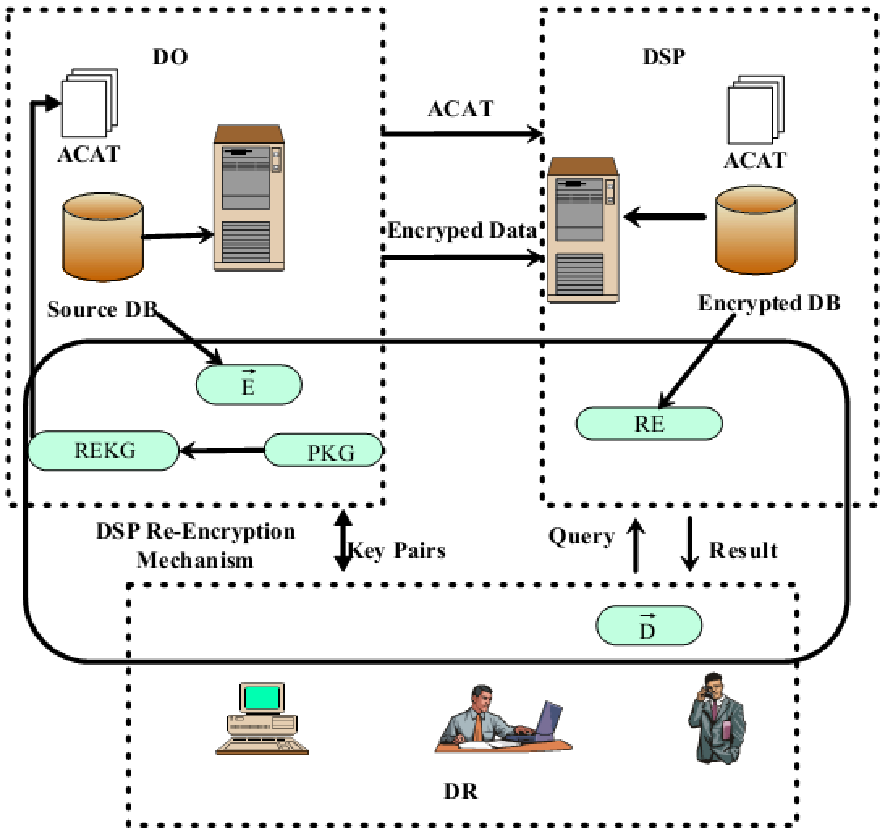


Figure 4.3: Structure design of DSP re-encryption [TWZ09]

The General process of the DSP re-encryption is specified as follows: At first, the client side (Data owner) uses public key PK1 to encrypt the tuples in the source database and then store them on the Cloud as encrypted tuples with additional information, such as index on tuple-id. Then ACAT is created by the data owner and uploaded to a remote server. In ACAT, access specification and re-encryption key (PK_user) are stored for each user. When users retrieve tuples from the database, the remote server needs to check the ACAT to decide whether users have authorized access. If a user is authorized, the remote server re-encrypts the result using the PK_user assigned by the data owner. Users could decrypt results using SK_user. SK_user is generated by the data owner and assigned to the data user.

In this approach, using DSP re-encryption as an access control for multi-user is a novel approach which is not addressed in other systems. But the disadvantage is obvious, re-encryption consumes too much computation. An analysis of this approach from four aspects is specified as follows:

- Performance: The original design focuses on the access control for multi-users in the cloud. Query operations are performed on tuples in the cloud database; record retrieval is performed by using index on the table primary key (ID). Since the encryption granularity is at the tuple level, many SQL operations are not well supported[TWZ09].

- Technological maturity and complexity: This system is a prototype, which needs re-enforcement and improvement. Experiments done in [TWZ09] focus on user control and tuple-level insertion and deletion, which are only a part of the SQL operations.

- CSP support: In DaaS model, this system could not be supported unless CSP reconstruct their whole system architecture. In IaaS model, users need to build the whole system by themselves. They need experts and do much configuration and maintenance work.
- System or application modification: In the system design, users interact with a remote database server. When using applications to interact with a remote server, decryption should be performed by the applications. This requires modifications in the applications.

In this section, general client side encryption approaches are discussed. For these approaches, a common technique difficulty is to perform efficient query over encrypted data, this topic will be discussed in 4.3. In contrast to client side encryption, server side encryption also provides a certain level confidentiality assurance. This will be discussed in the next section.

4.2 Server-side Encryption

For server side encryption, approaches will also be described and discussed with a similar structure based on the classification of simple object storage and relational data storage. At the end of each approach, analysis will be given.

4.2.1 Simple Storage Server-side Encryption

CSPs provide transparent server side encryption for simple storage object. For example, Google automatically encrypts Google storage objects when writing them back to unstructured storage [Bar]. The encryption algorithm is using AES-128, and the encryption system is a two tier architecture. Each object is encrypted individually and the encryption key is encrypted by a regularly rotated set of master keys [Bar]. Amazon provide a similar encryption feature for S3. The encryption system is also designed as two tier architecture. Which is the same with the system design in Google, the difference is that Amazon uses AES-256 to encrypt data before writing it back to disks. This encryption feature is not automatically enabled in Amazon S3, users could specify server side encryption using REST API or using AWS SDK, multiple programming languages are available, detail could refer to [Ama14b].

Server side encryption frees the complex encryption/decryption management. This approach could successful prevent attacks from outside or inside disk theft. With additional access control policies, it is much more difficult for a malicious administrator to get access to the users privacy data. This conclusion is derived based on the assumption that cloud security administrator and system administrator are separated and the malicious system administrator has no access to the encryption keys. Since the encryption keys and master key are managed by the CSP, this may still cause concerns about sensitive data privacy in Cloud. When the CSP wants to get access to sensitive data, server side encryption has no effect on privacy preservation. Analysis of this approach is given in the following text.

- Performance: Since the encryption/decryption process is done in the cloud, performance loss is supposed to be less than in local system. Comparing to local system, Cloud has much more computation power.
- Technological maturity or complexity: This technology is not for Amazon or Google challenge. For small CSPs, this may be complex. Considering the fact that the user prefers choosing leading CSPs in the market, there should not be much concerned.
- CSP support: As it is specified above, server side encryption is widely supported in the cloud.
- System or application modification: Applications and systems do not need to be modified.

4.2.2 Relational Data Server-side Encryption

In this part, there are two types of encryption: using encryption features provided by databases and full disk encryption from the CSP.

The first type, such as RDS from amazon, provides several database instances to use. Available choices are: Oracle, MySQL, SQLserver, PostgreSQL. Among these database instances, Oracle Transparent Data Encryption (TDE) and SQLserver TDE feature are supported by RDS. TDE in Oracle and SQLserver are not the same architecture, though their names are identical. Encryption features provided in RDS are based on the design of original database products.

The original Oracle TDE system overview has been discussed in Figure 2.5. There are two types of encryption enabled in Oracle TDE, column encryption and tablespace encryption. A detail discussion and comparison of them will be specified in the following according to [Ora14]:

Encryption Layer

- Column Encryption: SQL layser.
- Tablespace Encryption: Before read/write disks.

Encryption Scenarios

- Column Encryption: Single column sensitive.
- Tablespace Encryption: Multiple columns sensitive.

Storage overhead

- Column Encryption: Maximum of 52 bytes storage overhead for each encrypted value.
- Tablespace Encryption: No storage overhead. But users need to specify the maximum storage size of encrypted tablespace, for example, 150MB.

Create Encryption

- Column Encryption: Encryption existing column available, but encrypted column size is restricted by data types.
- Tablespace Encryption: Encryption of the existing tablespace is not available. New encrypted tablespaces should be created. No column size restriction.

Query Issues

- Column Encryption: Limited index type on encrypted column is available with encryption without salt. Index is created on the encrypted values. When query on encrypted column, index lookup using encrypted values.
- Tablespace Encryption: Index is unlimited as in normal table. Range scan and other operations are supported.

Performance

- Column Encryption: Performance reduction only when retrieved from or inserted into encrypted column
- Tablespace Encryption: Performance impact varies depending on the actual application. The overhead is roughly estimated to be between 5% and 8%.

To enable Oracle TDE in RDS, DB instance need to be associated with an option group in which TDE option is enabled, user could create a new option group or add TDE option to the existing option group [Ama14a]. When a DB instance is enabled to use TDE, it is not possible to disable this feature. Not all features are supported in RDS, for example data pump with TDE. Amazon RDS only supports the password encryption mode ($ENCRYPTION\_MODE = PASSWORD$) for Oracle Data Pump. TDE encryption mode is not supported [Ama14a]. Another feature that is not supported by Oracle instance in Amazon RDS is theHardware Security Module (HSM). HSM is a physical device that provides secure storage for encryption keys and also provides secure computational space for encryption and decryption operations [Ama14c]. In local Oracle databases, HSM could be used to store the TDE master key in tamper-proof storage and perform encryption/decryption operations [Ora14]. In Amazon, HSM is only supported when using Amazon Virtual Private Cloud (VPC) [Ama14c]. The oracle TDE master key and the related encryption/decryption process is performed by Amazon RDS. When IaaS model is deployed by user, then user could build a much more security database system in the cloud using oracle TDE together with HSM.

TDE architecture in SQLserver is different comparing to with TDE in Oracle. Amazon RDS supports the usage of transparent data encryption (TDE) to encrypt stored data for SQL Server 2008 R2 Enterprise Edition and SQL Server 2012 Enterprise Edition [Ama14a]. In the following, the encryption process are described according to [Mic14b]. In the description of Figure 4.4, it is clear that TDE is not an individual database component; it is tightly integrated with the security system in the SQL server. The root protection of the key hierarchy is based on the operating system level data protection API. TDE automatically encrypts data before it is written to storage and automatically decrypts data when the data is read from storage. This real-time I/O encryption and decryption is performed and scheduled by SQL server background threads [Mic14b].

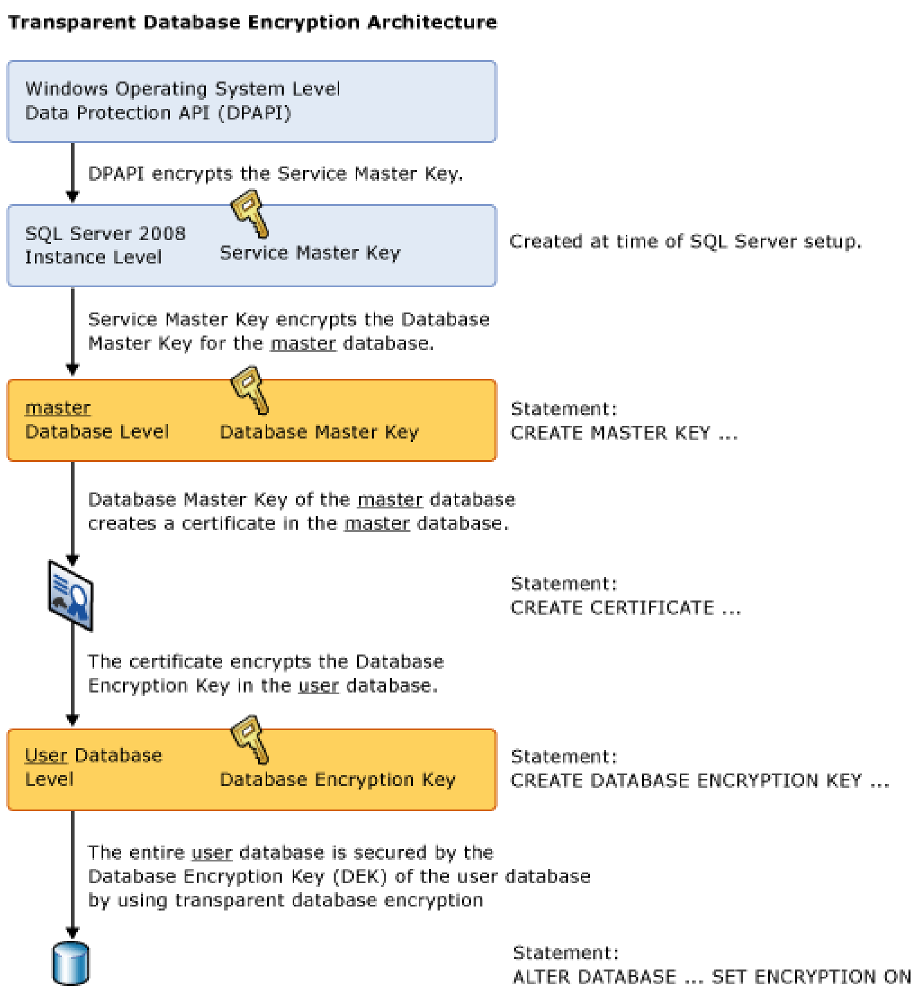

Transparent Database Encryption Architecture
Windows Operating System Level
Data Protection API (DPAPI)
DPAPI encrypts the Service Master Key.
SQL Server 2008
Instance Level
Service Master Key
Created at time of SQL Server setup.
Service Master Key encrypts the Database Master Key for the master database.
master
Database Level
Database Master Key
Statement:
CREATE MASTER KEY ...
Database Master Key of the master database creates a certificate in the master database.
Statement:
CREATE CERTIFICATE ...
The certificate encrypts the Database Encryption Key in the user database.
User Database
Level
Database Encryption Key
Statement:
CREATE DATABASE ENCRYPTION KEY ...
The entire user database is secured by the Database Encryption Key (DEK) of the user database by using transparent database encryption
Statement:
ALTER DATABASE ... SET ENCRYPTION ON

Figure 4.4: Structure design of SQLserver TDE [Mic14b]

Encryption of the database file is performed at the page level. Database pages in an encrypted database are encrypted before they are written to disk and decrypted when read into memory. This way, TDE does not increase the size of the encrypted database [Mic14b]. Since the encryption is not performed in the SQL layer, operations in database are not restricted. Although normal SQL functions are not affected by TDE, several operations are not allowed during initial database encryption, key change, or database decryption, such as taking the database offline, dropping the database. More restrictions when enabling SQL server TDE refer to [Mic14b].

To enable transparent data encryption for a DB instance that is running SQL Server, user need to specify the TDE option in an Amazon RDS option group that is associated with an SQLserver instance [Ama14a]. Encryption and decryption processes in Amazon RDS are identical with them in the local SQLserver, but the key management structure is modified to adjust to Cloud service model. According to [Ama14a], a two-tier key architecture is deployed to manage the encryption key; a certificate is used to protect the database encryption keys, which is generated by using the database master key. To provide better protection of database encryption key and hence comply with several security standards, Amazon is working on the implementation of automatic periodic master key rotation [Ama14a].

There are already multiple choices to store relational data in the cloud when deploying DaaS model. As it is illustrated above, TDE could protect data privacy at rest. For NoSQL database, for example, HBase, deploying IaaS model in the cloud is also attractive. TDE in Hadoop HBase is also supported. The general process has been discussed in chapter 2. When implementing HBase inEC2, combination of transparent key encryption and Hardware Security Module (HSM) could efficiently protect privacy data in the cloud.

The second approach of server side encryption is the system used by Google. All data that stored in Googles unstructured storage is automatically encrypted with AES-128. Key management and encryption/decryption processes are performed by Google [Bar]. When using Google Cloud SQL, a MySQL instance that lives in the cloud, Cloud SQL customer data is encrypted when on Google's internal networks and when stored in database tables and temporary files. This solution is not specified detail in [Bar], but it could be considered as a unique form of full disk encryption. This should prevent attacks from disk theft or some of the malware bugs. Privacy is protected to a certain level, but privacy concerns are not relieved unless the trust on CSP is built by using other approach. Cloud audit and Trusted computing mechanisms are used to build trust on CSP, this will discussed in Chapter six.

When using server side encryption, the data life cycle is similar to the data management in local systems, hence details of each phase are not discussed. The analysis of server side encryption is described as follows:

- Performance: When using server side encryption, performance reduction is acceptable. The advantages and disadvantages of deploying server side encryption should be weighed by users themselves, depending on the application scenarios [Ama14a].
- Technological maturity or complexity: For encryption features which are already provided by original database, they could be supported by the Cloud with little modification. Complex configurations are performed by the Cloud. Another approach that is similar to full disk encryption is original enabled to support Google inside applications. It is now available for other users.
- CSP support: For different CSPs, server side encryption features are provided according to their own infrastructure and service model. Key point is that, current encryption key management is performed by CSP. Though additional security features are provided, such as HSM in Amazon, they are not supported by all the service models. This matches the nature of cloud services, the higher level service is delivered, the less control the user has over Cloud resources.
- System or application modification: Applications above relational database or NoSQl database do not need to change.

4.3 Searching Encrypted Data

NEW CONTENT from here to the end of the chapter

When using client side encryption, data privacy could get properly preserved. While the main difficulty is query over encrypted data. When using server side encryption, such as Oracle tablespace TDE, performance reduction is limited. When using column encryption, there are many restrictions that greatly affect query performance, such as join and range query. Other problems of using column encryption including trigger, loses constraints integrity are open challenges [AEKR].

Searching encrypted data is a common obstacle when using Cloud. In this section, general approaches of query encrypted data are discussed. Theses approaches will be classified into two categories: query over encrypted data and query over index. Query over encrypted data is the initial approach to perform computation over encrypted data while query over index is a performance enhanced approach via querying over privacy preserving index.

4.3.1 Query over Encrypted Data

As it is described above, searching encrypted data is the main obstacle. Two general categories of approaches are addressed to solve this problem. The first category is keyword search over encrypted data and the second category is SQL query over encrypted data. Approaches in these two categories are discussed in the following text.

Keyword Query over Encrypted Data

Keyword word search over encrypted data means that direct comparing encrypted keyword with encrypted data in the cloud, this is obvious not practical. Even when small amounts of documents are encrypted in Cloud, this native solution will fail. An improvement to this approach is to assign keywords to documents, keywords are separately encrypted in order to perform search operation on them. Keywords in a document could be the title of the document plus a few user defined keywords.

According to the encryption algorithm difference, two general categories could be classified, symmetric encryption keyword query and asymmetric keyword query. Symmetric encryption keyword search suits for single user environment. Encryption keys are kept by a single user and there is no need to distribute encryption keys to the other users. In the Cloud environment, multi-user queries need to be enabled; hence asymmetric keyword query is widely used to address this challenge. In the following text, approaches in this category are discussed.

Asymmetric approaches: The first Public key Encryption with Keyword Search (PEKS) approach is promoted in [BDCOP04], this approach is based on the development of Identity Based Encryption (IBE). The first practical identity based encryption system [BF01] is proposed in 2001, it is constructed based on the assumption of a variant of the computational Diffie-Hellman problem. It uses bilinear maps between groups (Weil pairing on elliptic curves used in [BF01]) to construct the system. In [BDCOP04], the author demonstrated that PEKS implies Identity Based Encryption, and PEKS is a much harder problem. The initial purpose and the initial design goal of PEKS is to design a secure mail system that enables users to retrieve Emails from a server without leaking any information of queries and the email content to the remote server. This approach could be used in the cloud storage environment. The basic architecture of PEKS is specified in Figure 4.5. Based on the introduction in [BDCOP04], the general processing procedure of using PEKS in the cloud is specified as follows:

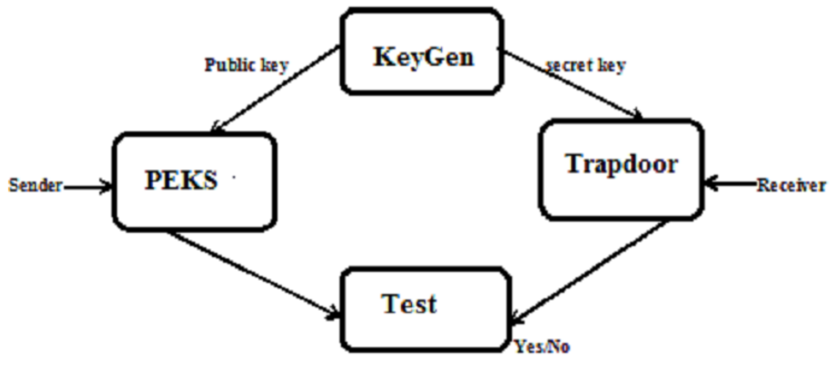


Figure 4.5: Structure design of PEKS adapt from [SS13]

- KeyGen: For different users, corresponding public/secret key pairs are generated. An open challenge is the access control policy designation, when documents are authorized for multi-users to use. Which means that secret keys must be distributed to multi-users.
- PEKS: Documents and keywords are encrypted using corresponding public key. An example of the encrypted result is:
  $[E\_A_{pub}[Document]; PEKS(A_{pub}; W_1); \ldots ; PEKS(A_{pub}; W_k)]$
- Trapdoor: Trapdoor is generated by authorized users and sent to Cloud. It is used to perform the keyword query without leaking the query information. An example of trapdoor of $keyword_x$ from usr A is like:
  Trapdoor $T_W = (A_{priv}, W_x)$
- Test: The Cloud performs the keyword search using a test program Test(A_{pub}, S, T_W). The test program uses a public key from user A, an encrypted keywords set of a document and the trapdoor of the queried keyword to get an output, which identifies if W_x match W_i from the encrypted keywords set. If keyword match is found, corresponding document is sent to user A.

Problems of deploying PEKS in the cloud are: Users who have corresponding secret keys could perform keyword query correctly, otherwise there will be no match for unauthorized users. But the document access control is complex, allowed access assignments are limited. Improper access policy may cause the crash of the system security mechanism. Another problem is that, CSP could build a mapping between a trapdoor and the encrypted keyword; this potentially reduces the system security strength.

The original design has also a few drawbacks that may reduce the system efficiency. Paper [BSNS08] has summarized three main problems: Trapdoor information could be stored for later use and then the remote server could get the correspondent keywords categories based on the statistic information. This is discussed above and it remains as an open challenge. Another problem is that PEKS uses a secure channel to secure the communication between users and the remote server, Paper [BSNS08] has promoted a design that without a secure channel, the system could still perform the keywords search securely. This problem is actually not necessary in the cloud environment. The normal service provider now suggests users to use a SSL connection to get access to data and perform operations on the data. The third problem is that, original PEKS is designed to be a one-time query system. In [BSNS08], A new schema that enables efficient multiple keywords search is promoted. The new schema could be implemented in the cloud environment.

Considering the limited computation power of a local client, [LWW09] promoted an enhanced schema that improves the system efficiency by allowing the service provider to participate in the decryption process and then return an intermediate result without knowing the plaintext. This enhancement suits for the Cloud environment very well, because clients of the Cloud have also limited computation power. It borrows the partial decipherment idea from [DLKY04] and proposes a new design. It utilizes the mathematical characteristics of PEKS to realize the system, detail design and mathematical proof of the system could refer to [LWW09].

There is a general survey of keywords search over encrypted data [SS13], in which more solutions that have different emphasizes, such as fuzzy keywords and access control, are briefly introduced. Despite those efforts, these approaches has an inherently disadvantage that the search result is not ranked. Hence when many documents share identical keywords, the result will be oversized. Local users may need to decrypt all the documents to find the accurate document that he needed. This results in a great internet bandwidth waste and monetary loss in the cloud pay-per-user price model [WCL$^+$10].

To enable ranked keyword search, many approaches are promoted, this will be discussed in Section 4.3.2.

SQL Query over Encrypted Data

To perform SQL queries over encrypted data, current approaches focus on the combination of different encryption algorithms. The native approach to enable SQL queries over encrypted data is to encrypted different data types using different encryption algorithms, such as deterministic encryption, order preserving encryption. CryptDB [PRZB11]is a successful design which utilizes multiple layers of encryptions to achieve high privacy protection level and efficient performance. The structure overview is described in Figure4.2. The most important contributions of CryptDB are SQL-aware encryption functions and the web proxy. To enable SQL operations over encrypted data, six types of encryption functions are used in CryptDB. According to the description in [PRZB11], these six types of encryption are specified as follows:

- Random (RND): Random means non-deterministic, which means that equal values are almost always encrypted into different cipher texts. When using this type encryption, no SQL operations could be performed over encrypted data. It is used on columns that are not needed to be queried, or it is used to encrypt the outermost layer of the encryption onion to provide highest security. In CryptDB, integer values are encrypted using Blowfish with 64-bit block size in CBC mode together with a random initialization vector to reduce cipher text length, other values are encrypted using AES similarly to Blowfish.

- Deterministic (DET): Deterministic means equal values are always encrypted in to identical cipher text. This kind of encryption is used to perform equal queries over encrypted data, which means that selection with equality predicates, equality join, GROUP BY etc. are supported [PRZB11]. Though deterministic encryption has a slightly weaker security, by adding other encryption layers, CryptDB provides sufficient guarantee on data confidentiality and privacy preservation.

- Order-preserving encryption (OPE): This means that, if the (P-value) plain text value has a relation that P-value-1 $>, or <$, P-value-2, cipher values of P-value-1 and P-value-2 remains the same relation. With this property, SQL comparison queries could be performed over encrypted data, which means that MIN, MAX, RANGE QUERY, SORT, etc are supported. Order preserving encryption has a weaker security comparing to Deterministic encryption. In CryptDB, columns that use order preserved encryption are revealed to the remote server only when necessary.
- Homomorphic encryption (HOM): Fully homomorphic encryption means that any operations of plain text values could be mapped into operations of cipher values without decryption, which is not yet practical [AEKR]. In CryptDB, partial homomorphic encryption that allows direct add(+) computation over encrypted values is deployed. This enables SQL operations such as SUM. The Detail of SQL operation process in CryptDB is described in [PRZB11].
- Join (JOIN and OPE-JOIN): Join operations could be performed when using deterministic encryption. In CryptDB, columns in different tables are encrypted using different keys. Hence a new encryption schema is designed in CryptDB to support join operations between different tables.
- Word search (SEARCH): This means keywords search over encrypted data. The cryptographic protocol that implemented in CryptDB is promoted in [SWP00]. An improved utilization of this protocol allows CryptDB to achieve better security guarantee.

To provide sufficient security guarantee and better performance, *Adjustable Query-based Encryption* is used to dynamically adjust the encryption layers. The native approach to provide this possibility that allows SQL queries over encrypted data is to predicate possible query types on certain columns. But this is not practical, because the accurate query set on certain columns is not predictable in advance. Thus CryptDB deploys the idea of onion-layer encryption to dynamically adjust encryption strategies. Examples of onion-layer encryption are showed in Figure 4.6

In the onion-layers encryption hierarchy, each layer supports certain SQL operations. Normally the inner layer has more support to SQl operations, the outer layer has less support to SQL functions and the outermost layer allows no SQL operations.

> "Multiple onions are needed in practice, both because the computations supported by different encryption schemes are not always strictly ordered, and because of performance considerations (size of cipher text and encryption time for nested onion layers)." [PRZB11]

To perform SQL queries in CryptDB, following process are proceeded: once the web proxy receives a SQL query, based on the operation types and current encryption layers

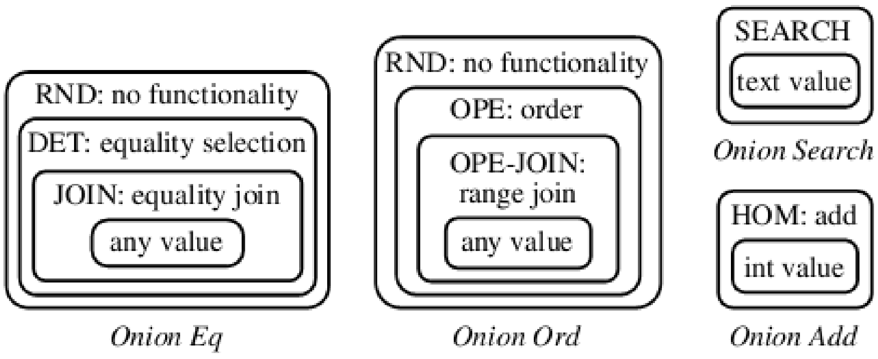


Figure 4.6: Onion-layer encryption examples and hierarchy adapt from [PRZB11]

of columns, certain decryption keys are sent to the DBMS server. The remote server could decrypt related columns to a certain layer to allow queries on them. The web proxy will replace related column names and query values with corresponding onion names and cipher values, the remote server could perform rewritten queries just like normal queries. There are many details of CryptDB architecture and query processing that not specified here, please refer to [PRZB11] for better understanding.

4.3.2 Query over Index

Instead of direct query over encrypted data, query over index is an alternative to use. Approaches that enable keyword search and SQL query have been promoted. In this section, these two categories of approaches are discussed.

Keyword Query over Index

Performing keyword query over index provides a better performance comparing to query over encrypted data. In order to prevent privacy leakage when outsourcing index to CSP, secure index is promoted in [G$^+$03]. In this paper, an index security model is formulated against adaptive chosen keyword attack, which is named as Semantic Security Against Adaptive Chosen Keyword Attack (IND-CKA). To build a secure index, four algorithms are used, they are: Keygen(s), Trapdoor, BuildIndex, SearchIndex. Furthermore, an index schema is promoted to fulfill acfIND-CKA security. This schema deploys Bloom Filter as the index to track keywords. To enable privacy preservation on the indexes, keywords in each document index are generated by using pseudo-random functions, which means CSP knows nothing about keywords. For each keyword in each document index, corresponding trapdoor is generated to allow query on them. Other details of the design of this schema and query processing procedure could refer to [G$^+$03]. Since other approaches that enable keyword and multi-keyword search has been discussed in Section 4.3.1. In this section, only ranked keyword search approaches are discussed.

To enable ranked keyword search, index that with additional weight information of documents should be constructed and stored in the cloud. Hence the classification of approaches that enable ranked keyword search will be based on the difference of calculating weight information and protecting index security and privacy in the cloud.

There are already many researches that discussed calculating document weight information in Information Retrieval(IR). Inverted index is widely used to map keywords to related file sets [WCL$^+$10]. When we assign each file with a related weight score, ranked search is enabled. Weight information could be calculated by using statistical measures, the most widely used measure is Term Frequency Inverse Document Frequency (TF-IDF), details of this measure could refer to [WCL$^+$10]. There are also other weight calculation functions, which will be discussed later.

Since the key designs of these approaches that enable ranked keyword search are index building and index security in Cloud, a short specification of these two designs for each approach are described as follows:

Secure Ranked Keyword Search [WCL$^+$10]:

- Build Index: Weight scores of documents are calculated using Term Frequency Inverse Document Frequency (TF-IDF), indexes are built use inverted index structure.
- Index Security: A special one-to-many order-preserving symmetric encryption named as *OPM* is used to encrypt weight scores. *OPM* incorporates the unique file IDs together with weight score to generate a random value in a certain range. Furthermore, different keys are used to encrypt different posting lists (*for each keyword, corresponding document locations and keyword locations with additional weight scores composed of such a posting list*) to make the weight score more randomized from the overview point of the CSP.
- Weakness: For a certain keyword, a different key is used to encrypt weight scores, this key is also sent to cloud. When related keyword posting list is found, weight scores could be decrypted and documents that being queried return in a ranked order. This means that CSP will constantly keep the related decryption key for each keyword. The result is that CSP knows document weight scores on each ciphertext of keywords, keywords and user document content is not exposed to the CSP.

Multi-keyword Ranked Search over Encrypted cloud data (MRSE) [Nin12]:

- Build Index: Weight information is generated using inner product similarity. When constructing indexes, a binary vector $D_i[j]$ is created for each document D_i. Which means that if the corresponding keyword W_j appears in document D_j [Nin12]. When performing queries over the index, a binary vector Q is also built, in which Q_j means if corresponding keyword W_j exists in the query or not [Nin12]. As a query is sent to CSP, weight scores is dynamically generated by using inner product of Q and D_i for each document.
- Index Security: If binary vector Q is constructed locally, then the privacy of keywords and documents remains preserved. Because the CSP only knows these binary vectors and has no idea of the actual keywords set and the order.

But to preserve privacy strictly, even data vectors, query vectors and the inner product of them are not exposed to CSP. The index privacy design deploys the idea of secure k-nearest neighbor(kNN) computation [WCKM09] and adjusts this idea to securely compute the inner-product similarity. Detail of the security proof could refer to [Nin12].

- Further Generalization: Similar techniques are used to search graph documents. Instead of keywords, feature information is queried. Through building feature based index for graph documents and using efficient inner product as weight information, an efficient privacy preserving graph search framework is promoted in [Nin12].

Most of the approaches discussed above did not address the access control problem. [LYCL11] discussed this problem and promoted an efficient schema that allows access assignment and revocation. It enriches the choices when designing secure and reliable cloud storage services.

With the discussion above, the efficiency and privacy preservation of keyword search over index is demonstrated to allow users to enjoy the benefits of Cloud. Another important application is database services in the cloud. The next section will discuss approaches that enable SQL queries over index.

SQL Query over Index

Encryption algorithms that allow SQL operations have been introduced in Section 4.3.1. Index is widely used to improve query performance in unencrypted databases. Similarly, constructing secure indexes that enable SQL queries is researched. An approach that deploys this idea is promoted in [HILM02]. This approach is used in a DaaS model.

The basic idea of this approach is to push as many queries as possible to the CSP side, local clients take care of the final decryption and query processing. According to the description in [HILM02], the general design of this approach is specified as follows: In this design, a database is encrypted in tuple level. Tuples are encrypted and augmented with additional information, i.e., secure indexes. The tuple encryption algorithm could be symmetric or asymmetric, to improve the database performance, symmetric encryption is preferred. Secure indexes are built only on attributes that involve in search or join predicates. A secure index is built based on the partitions of attribute values. An example of tuple stored in remote server is in the form of:
$emp(eTuple, eid^s, ename^s, esalary^s, eaddr^s)$ In this tuple, eid^s is the secure index value. This value is generated through two procedures. The first procedure is attribute value partition, the second procedure is a collision-free hash function that is used to identify the partition. In this way, index is supposed to have better security when comparing to Blob Store [AEKR]. It uses "fake" partitions to index blobs.

In general, eid^s is the result of the mapping function $ident_{id}(partitionx)$, which maps attribute values to index values. Two types of mapping function are discussed in [HILM02], random mapping function and order preserving mapping function. Definitions of these two types mapping function is similar to the definition of random encryption and order preserving encryption, which were discussed in Section 4.3.1. Order preserving mapping is used in this thesis to simplify the specification of the system design.

In SQL queries, query conditions are used to locate related tuples. In [HILM02], query conditions are classified into three types:

1. Condition $\leftarrow$ Attribute *op* value;
2. Condition $\leftarrow$ Attribute *op* Attribute;
3. Condition $\leftarrow$(Condition$\wedge$condition)$|$(Condition$\vee$condition)$|$($\neg$Condition)

   Allowed operations for *op* are$(=, <, >, \leq, \geq)$.

Mapping of these attribute conditions to indexes are listed as follows:

- Attribute = Value:
  $Map_{cond}(A_i = v) \Rightarrow (A_i)^s = Map_{A_i}(v)$
- Attribute < Value:
  $Map_{cond}(A_i < v) \Rightarrow (A_i)^s \leq Map_{A_i}(v)$
- Attribute > Value:
  $Map_{cond}(A_i < v) \Rightarrow (A_i)^s \geq Map_{A_i}(v)$
- Attribute = Attribute:
  $Map_{cond}(A_i = A_j) \Rightarrow \vee(A_i)^s = ident_{A_i}(p_k) \wedge A_j)^s = ident_{A_j}(p_l))$, where $p_k inpartition(A_i) and p_l inpartition(A_j$, $p_k \cap p_l$ not empty. "This means all possible all possible pairs of partitions of A_i and A_j that overlap" [HILM02].
- Attribute < Attribute:
  $Map_{cond}(A_i < A_j) \Rightarrow \vee(Map_{A_i}(p_k\_low) \wedge Map_{A_i}(p_l\_high))$, where $p_k inpartition(A_i)$and $p_l inpartition(A_j)$.
- Condition-1 *and* Condition-2,Condition-1 *or* Condition-2:
  $Map_{cond}(Condtion\_1) andMap_{cond}(Condtion\_2)$;
  $Map_{cond}(Condtion\_1) orMap_{cond}(Condtion\_2)$

The original design has discussed about mapping to index when use random map functions($ident_{id}(partitionx)$) as well, detail could refer to [HILM02]. With the mapping to index listed above, queries are performed on the remote database index. The intermediate result is returned by remote server, the final decryption and processing is performed locally.

This approach may result in building indexes for all the attributes, accurate predicates are not practical even with the help of statistical research. Another challenge is that the intermediate result depends on the partition granularity. The proper partition of attribute values should be chosen in order to balance the performance and security. Anyway, the experiment result given in [HILM02] demonstrated its partial efficiency. Two types of queries were performed in the experiment, selection and join. They are the most common SQL query operations.

Another approach that enables SQL query over index is promoted in [WAEA11]. In this approach, secure B+-tree indexes are built on frequently queried attributes. Then index related tuples are encoded and dispersed into the matrix by using salted IDA

(Information Dispersal Algorithm) [WAEA11]. In order to determine which attribute is to be indexed, statistical data is utilized. This means, when migrating existing databases to the Cloud, this approach would achieve a better performance. Other details of query processing and tuple organization could refer to [WAEA11].

4.4 Summary

Encrypting data in the cloud could efficiently protect data confidentiality and privacy. In this chapter, client side encryption and server side encryption are discussed. In related issues of deploying encryption, the focus of this thesis is to query over encrypted data. General approaches are discussed in Section 4.3. While client side encryption could achieve better confidentiality and privacy protection, data management and query are difficulties. Server side encryption could achieve a better performance. User data privacy is properly protected against outside attacks and inside attacks such as disk theft. The main problem is that key management and encryption/decryption process are performed by CSP, trust need to be improved to achieve better privacy protection in the cloud. Cloud audit and Trusted computing are proposed to address this problem, related topics are discussed in Chapter 6.

Figure 4.7 gives the classification of encryption based approaches to provide better overview.

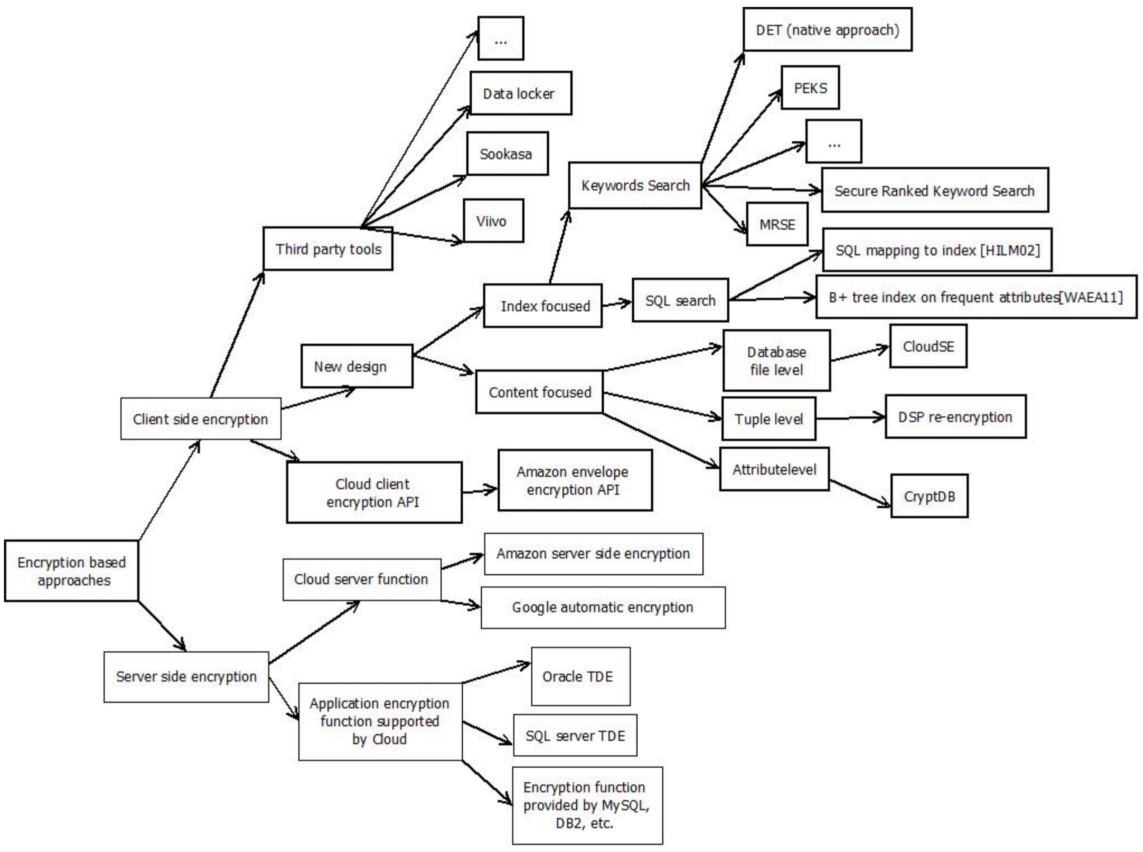

Encryption based approaches
Client side encryption
Server side encryption
Third party tools
...
Data locker
Sookasa
Viivo
New design
Index focused
Keywords Search
DET (native approach)
PEKS
...
Secure Ranked Keyword Search
MRSE
SQL search
SQL mapping to index [HILM02]
B+ tree index on frequent attributes[WAEA11]
Content focused
Database file level
CloudSE
Tuple level
DSP re-encryption
Attributelevel
CryptDB
Cloud client encryption API
Amazon envelope encryption API
Cloud server function
Amazon server side encryption
Google automatic encryption
Application encryption function supported by Cloud
Oracle TDE
SQL server TDE
Encryption function provided by MySQL, DB2, etc.

Figure 4.7: Overview of encryption based approaches

Chapter 5

Distribution-based Approaches

Privacy concerns are the major obstacle when a user deploys cloud services. To preserve user data privacy in the cloud, encrypting data to keep data confidential is a widely used approach. Similar to data encryption, as a novel approach, distributed file systems BIFS (Bit-Interleaving File System) [SMGL11] are being promoted by Zhonghua Sheng et al. from the University of Hongkong. This approach deploys Bit-Interleaving technology to disorganize data bits stored in the cloud. In this way, user data confidentiality is preserved and hence user privacy is efficiently protected. Other approaches employ a similar data disorganization principle to achieve privacy preservation in the cloud. The XML privacy preservation model [GWD14] promoted by Lihong Guo et al. from Nanjing is such an approach. In this approach, XML documents that contain user privacy data are protected by disorganizing privacy data elements stored in the cloud. The third approach is preserving privacy by employing a separation of duties [HHK$^+$]. User privacy data is stored in two separate CSPs. None of those two CSPs could successfully link related database records to a specific identity by itself.

Considering the common characteristic, that these approaches deploy the idea of data distribution to achieve privacy preservation in the cloud, they are classified as distribution-based approaches. The following sections are arranged according to the order of approaches described above. In each section, the general design of the approach is described and a analysis is given.

5.1 Bit-Interleaving File System (BIFS)

While encrypting user data and storing the cipher text as a unbroken logical unit in the cloud is efficient to protect user data privacy, the related overhead of decrypting computation may limit its use. An 30% overhead is demonstrated in [SMGL11], when adding an encryption layer above an ext4 file system. [SMGL11] promotes a novel file system, that provides efficient privacy protection without data encryption.

There are three general principles employed in [SMGL11] to design the system:

- Let the user handle data, and the infrastructure handle bits.
  This means that all data bits allocation information and data retrieval is controlled by the cloud user, cloud storage only acts as storage media without the global structure overview over the user data.

- Hide data by re-ordering, not substitution.
  This means data bits remain unmodified, but the order and connection between data bits is completely different. If cloud service providers or attackers try to read data bits just as reading a normal disk file, the result can not be recognized and is therefore meaningless. Instead of data encryption, the general process could be considered as multiple applications of bits substitution using specific mapping functions.

- Improve strength by distribution, not entropy.
  The security and privacy protection strength is improved by distributing data to multiple buckets. This means that without getting all the essential Bit-Interleaving fragments, data could not be decoded by attackers.

The basic file system architecture is described in Figure 5.1: In this architecture,

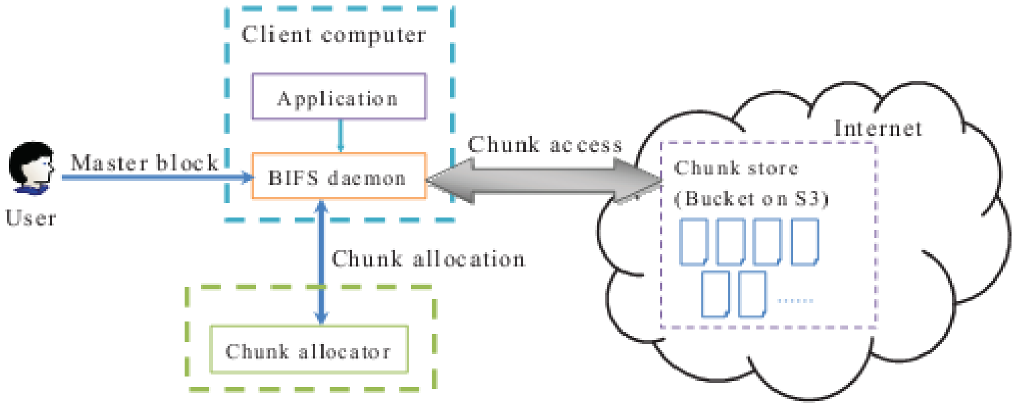


Figure 5.1: Structure design of Bit-Interleaving File System (BIFS) [SMGL11]

the most important components are BIFS daemon, chunk allocator and chunk store. BIFS daemon is responsible for converting files to slices and translating the PUT file operations (use REST in S3) to PUT slice operations. Reversely, file retrieval operations are translated to retrieve slices of file operations. And then those slices are reconstructed together by BIFS daemon. Chunk allocator is a performance optimization component which records the chunk allocation information to allow BIFS daemon faster access to chunks stored in Chunk store. Chunk store is a collection of chunks, which are stored in the cloud storage. The storage choice in [SMGL11] is Amazon S3. It is also available to deploy on other cloud storage services.

To store a specific user file, that contains privacy data in the cloud by using BIFS, the following steps are necessary: At first, a file is being divided into blocks of a predefined size. Then data bits in each file block are being reordered through Bit-Interleaving and divided into different slices with variable lengths. At last, those slices are allocated to random chunks in the chunk store by using a slice allocation algorithm. The logical view of such a file is described in Figure 5.2. Among these three steps, the second and the third step are the key design to provide privacy preservation in the cloud. An example that illustrates the working process of Bit-Interleaving is described in Figure 5.3. In this example, only three words are considered. In the actual system, data blocks are the basic input unit. Block size is fixed, that means the $\|d\|$ is fixed. System parameter m is set up in advance. According to the security analysis of this process, m should be set as larger or equal to 8 [SMGL11]. Since d and m is fixed, the whole matrix $D_{m,h}$ is fixed, $h = ceil[d/m]$. The rest of the matrix D is padded with bit 1. Slice split and allocation are using the same algorithm, which is being described in detail in [SMGL11]. The output of this algorithm is a block descriptor, which contains the corresponding slices allocation information. It contains a set of vectors of the form (chunk ID, slice ID). But the slice length and slice location within a chunk are not being specified in a block descriptor. Slice length is calculated by using a function, which contains three parameters: User credential (passwords or certificate), chunk ID and slice ID. The slice location is calculated by adding the lengths of slices which are being stored before this slice. This provides a better privacy protection, even when all the chunks are revealed to attackers.

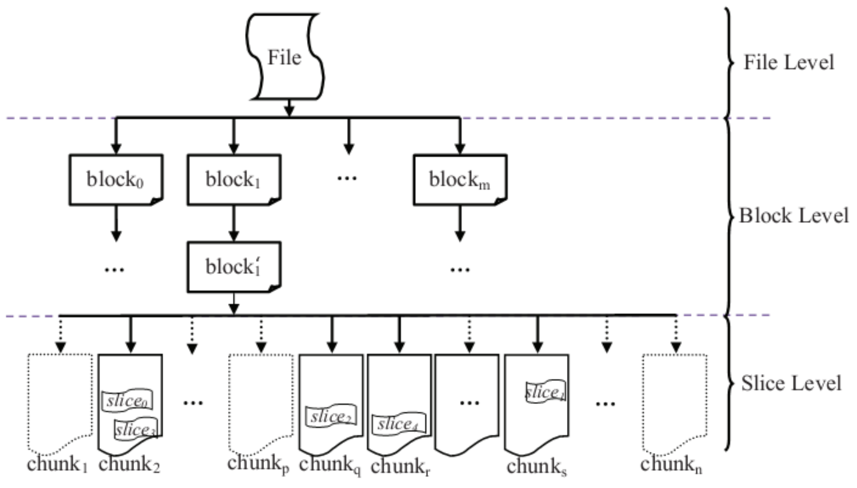


Figure 5.2: Logical view of Bit-Interleaving File System (BIFS) file adapt from [SMGL11]

The whole file system hierarchy is specified in Figure5.4. Master block is the entry of the whole file system here, which is a specific file descriptor. In this master block, sub-directory and file descriptor are stored as entries. Each file descriptor contains file names, file attributes and the root index block for block the descriptor. Following the block descriptor, slices could be retrieved from chunk storage in the cloud. It is clear that the master must be properly protected to preserve the whole system security and hence user privacy. Since the size of this master block is not big, it could be stored locally or stored on external storage devices. Whenever a user needs to use BIFS, the master block should be provided.

Besides the privacy preservation characteristics, system performance evaluations of the author show, that it achieves a very good score compared with other distributed file systems [SMGL11]. Related scores are listed in Table 5.1. When considering the influence of an unstable internet connection speed, these scores may vary dependend on multiple factors, such as location of client, time of using BIFS, internet service provider bandwidth limit. This means that to achieve privacy preservation, unstable performance has to be tolerated.

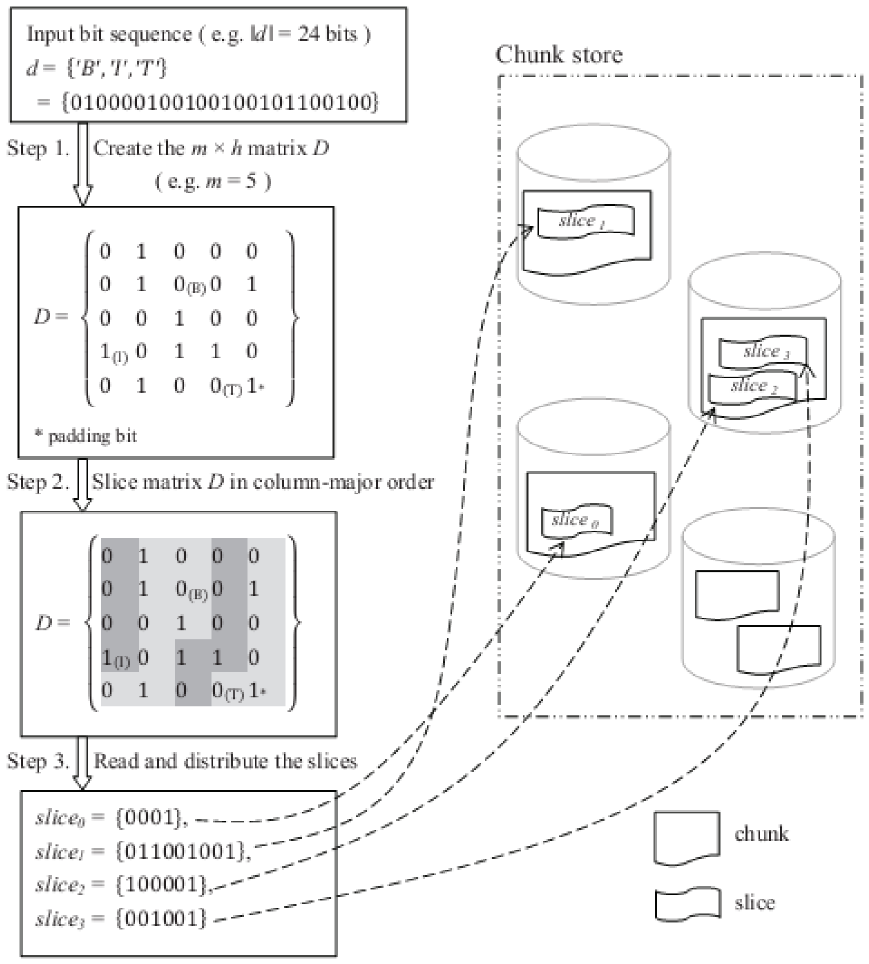


Figure 5.3: Bit-Interleaving process adapt from [SMGL11]

With the description above, the basic design of this system is specified. The analysis of its efficiency and other aspects are discussed in the following paragraph.

- Data types:
  BIFS is initially being used to store files, i.e. simple objects in the cloud. It is not suitable for relational and non-relational database services.

- Availability issue: The availability of BIFS is based on the availability of the cloud storage service. Since cloud storage services provide automatic backups and disaster recovery ability, BIFS enables user to enjoy the benefits of the high availability of the cloud. But from the inherent problem of the cloud, that services are only accessible via the internet, potential availability problems are unavoidable.

- Security and privacy:
  Comparing to date encryption, which data confidentiality is guaranteed by security proof of encryption algorithms, Bit-Interleaving security is not guaranteed. Though through data distribution, the security of system is improved, no formal attack model is built hence security proof model is available to demonstrate its efficiency. Nevertheless, [SMGL11] analyzed two attack scenarios: 1. Attacker

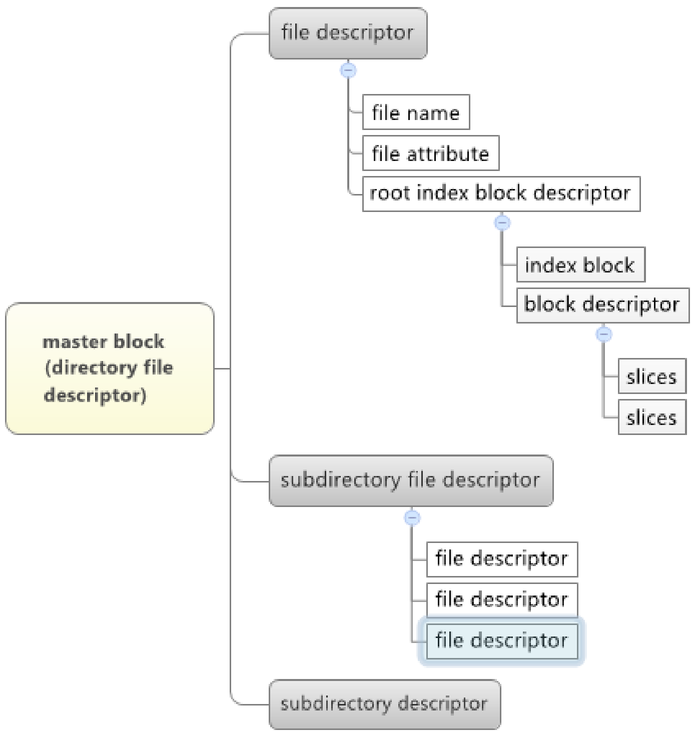


Figure 5.4: File System overview

Operation	Throughput
sequential read	2.3 MB/s
sequential write 18.3 MB/s	18.3 MB/s
file creation	4688 ops/s
file deletion	6052 ops/s

Table 5.1: BIFS evaluation data adapt from [SMGL11]

has obtained all the chunks that store user data. 2. A similar knowing-plaintext attack schema. Under both conditions, privacy could be efficiently preserved. Furthermore, [SMGL11] promoted, that by deploying two or more CSPs, privacy protection is enhanced.

5.2 XML Privacy Protection Model

XML privacy protection model is promoted to address XML document privacy preservation in the cloud. "...eXtensible Markup Language (XML) becomes a widespread data representation and exchange format for Web applications" [GWD14]. In the cloud, XML is also a standard exchange format. For example, Google provides an XML API for developers to communicate with Google storage and manage data in a programmatic way [Goo14]. Considering the wide use of XML documents to store user data, a novel model is promoted to protect user privacy. The basic idea promoted in [GWD14] is separating the content and XML structure of the document. In this model, document contents are stored in cloud storage as simple object, and the structure information is original designed to be managed by Trust Third Party (TTP). A local client could be built to achieve a higher security (privacy protection) level.

The model consists of the following three major components [GWD14]: user module, cloud module and TTP module. The TTP module is the most important module, its tasks are: Resolving the XML document and encoding the documents tree structure, creating the Document Type Definition (DTD) document and creating the sub DTD and corresponding document tree structure according to privacy control policies. A privacy policy defines, which nodes of the XML document are being accessible to which types of users and for which kinds of purposes. Details of access control are not discussed in this Thesis. The focus of this model is the structure encoding process of XML documents and the content storage architecture in the cloud.

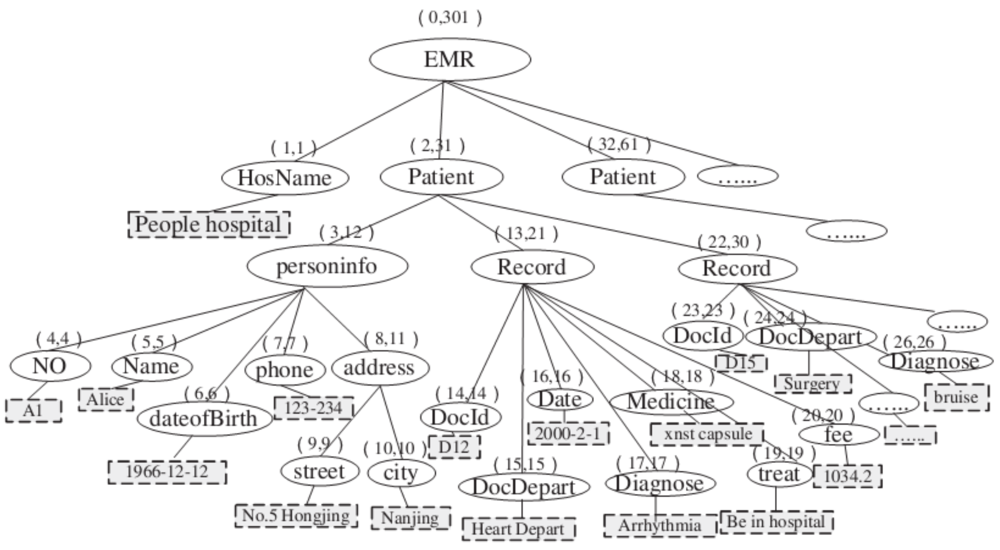


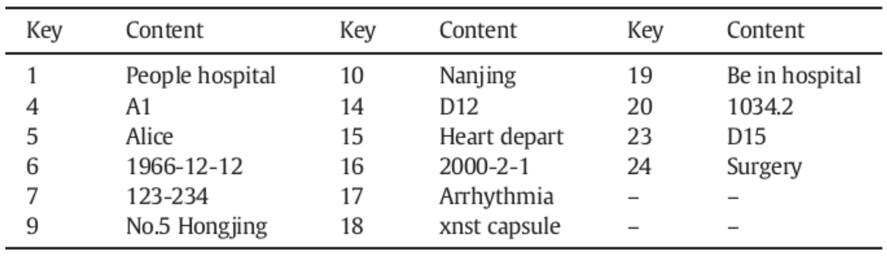
Figure 5.5: Health record XML document and its encoding adapt from [GWD14]

A XML document representing a health record is used as an example in [GWD14] to specify the process. The structure of this document and its corresponding node encoding are described in Figure 5.6. The tree structure of this document is easy to understand. The encoding of each node is using *(start,end)* tags. Node *start* tag in this document tree is assigned sequentially at each visit using depth-first-traversal, while for non-leaf nodes, *end* tags are assigned with the sequential number when revisiting these nodes; for leaf nodes, *start, end* tags are the same value.

Since the most valuable information is stored in content ,i.e. leaf nodes of the XML document tree, these nodes are stored in the cloud. The corresponding storage format is described in Figure 5.6. But in the actual application scenario, contents are protected by using cryptographic mechanisms.

When an authorized user request certain information from TTP, corresponding privacy policies are being checked. If privacy policies are not violated, corresponding DTD and content encoding keys are sent to the user, the user can get the content from the cloud by using encoding key, and then reconstruct the XML document respectively.

After the general introduction of this model, analysis of this model is given in the



Key	Content	Key	Content	Key	Content
1	People hospital	10	Nanjing	19	Be in hospital
4	A1	14	D12	20	1034.2
5	Alice	15	Heart depart	23	D15
6	1966-12-12	16	2000-2-1	24	Surgery
7	123-234	17	Arrhythmia	–	–
9	No.5 Hongjing	18	xnst capsule	–	–

Figure 5.6: Leaf nodes stored in the cloud [GWD14]

following paragraph:

- Advantages: A highlight of this model is, that it protects XML document privacy while at the same time, it provides a fine access control mechanism for the cloud environment. Compared to traditional XML document encryption standards, [SMGL11] formalized two problems: Encryption granularity selection difficulty and information disclosure between users. This model encrypts only leaf nodes (contents) in the cloud, with the access control performed by TTP, information disclosure between users is minimized.

- Disadvantage: While only leaf nodes are stored in the cloud, according to the encoding key, CSP could easily get the information structure based on the encoding key. For example, in Figure 5.6, encoding keys are plaintext and it is easy to find out that, content(1), content(4,...,10) and content(14,...,20) are sub-nodes of the parent node. If the encoding system is exposed to CSP for example, using depth-first-traversal, it is easy to rebuild the XML document structure. A possible enhancement to address this problem is to encrypt encoding keys using order-preserving encryption algorithms. When content is encrypted in the cloud, related encryption algorithms and key management is not addressed in [GWD14]. This problem needs to be solved to provide a fully functional privacy preservation system in the cloud.

5.3 Seperating Duties

In article [HHK+],[Hub10],[HHKMQ], researchers discussed a new approach to achieve trusted cloud computing. This approach deploys the idea of separating duties to database services. Separating duties means, "separating a service with respect to its algorithms and data structures and deploying each part on a different server "[Hub10]. In this way, each part of the service has only part control over the whole service, this will provide efficient protection against privileged administrators as long as separate services are not all compromised. There are two types of separating services, serial and parallel ones. When the separating service idea is applied on cloud database services, serial and parallel service separating may be the form described in Figure 5.7and Figure 5.8.

A practical example of serial cloud database service separation is CryptDB [PRZB11], which is discussed in Section 4.3.1. In CryptDB, a web proxy is the client database adaptor, it takes care of query translation and result decryption. The encrypted database

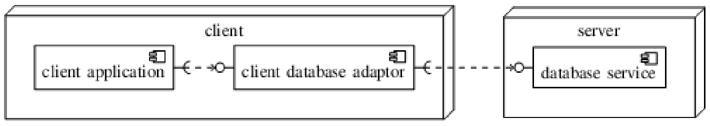


Figure 5.7: Serial database service separation [Hub10]

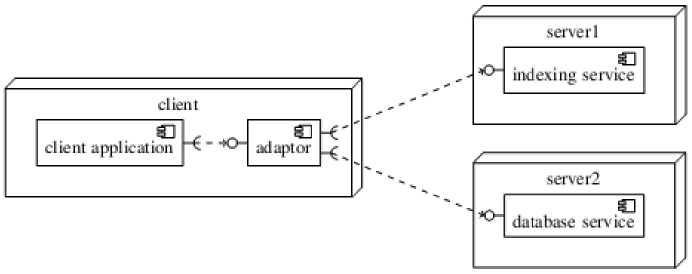
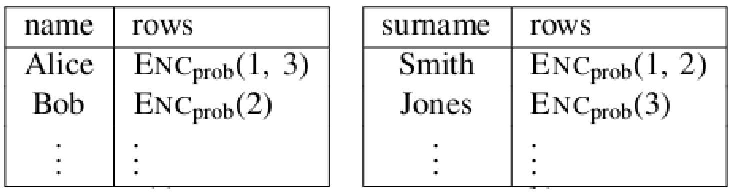


Figure 5.8: Parallel database service separation [Hub10]

is the server part of database services. Practical parallel database service separation is not yet discussed. In the following text, an example promoted in[Hub10]is used to specify the parallel service separating process.

Given a simple table *T*, *T(row-id, name, surname, bank account)*, *T* is separated in to three sub-tables, *T1(name,rows)*, *T2(surname,rows)*, *T3(row-id,*$ENC_{prob}(name, surname, bankaccount)$*)*. *T1* and *T2* are indexes built on *name* and *surname*, in *rows* column, corresponding *row-ids* are encrypted using a probabilistic encryption algorithm. In *T3*, other attributes values except *row-ids* are also encrypted using probabilistic encryption algorithms. These sub-tables are described in Figure 5.9 and 5.10.



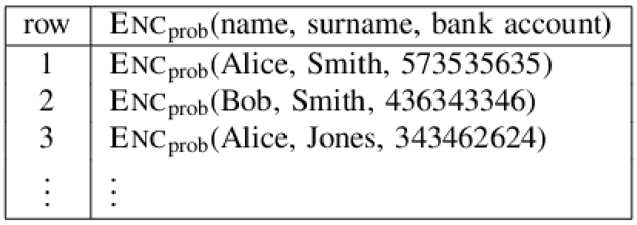
name	rows
Alice	$\text{ENC}_{\text{prob}}(1, 3)$
Bob	$\text{ENC}_{\text{prob}}(2)$
⋮	⋮

surname	rows
Smith	$\text{ENC}_{\text{prob}}(1, 2)$
Jones	$\text{ENC}_{\text{prob}}(3)$
⋮	⋮

Figure 5.9: Two index tables of separated database service [Hub10]

When a user needs to query, for example, the *bank account* of Alice Smith, two sub-queries are performed on *T1* and *T2*. After getting the results from *T1* and *T2*, the local client needs to decrypt the results and find the intersection, i.e., the *row-id* and then perform a query on *T3* using the intersection. Finally, *bank account* of Alice Smith could be queried without privacy leakage.

By encrypting *row-ids* in *T1, T2* and encrypting other attributes in *T3*, the above example provides additional privacy protection, but the available SQL queries in this example are limited. The author of [Hub10] plans to investigate more complex index structures to allow other SQL queries. Indexes that allow SQL queries have been discussed in 4.3.2, a combination of service separation and index technology could be a new



row	ENC_{prob}(name, surname, bank account)
1	ENC_{prob}(Alice, Smith, 573535635)
2	ENC_{prob}(Bob, Smith, 436343346)
3	ENC_{prob}(Alice, Jones, 343462624)
⋮	⋮

Figure 5.10: Content table of separated database service [Hub10]

direction to address cloud privacy concerns.

5.4 Summary

In this chapter, BIFS and the XML storage model were taken as practical examples of deploying data distribution ideas to preserve user privacy; Separating duties formulated the distribution idea into a formal security concept. A practical example of parallel separating tables into index and content is discussed in [Hub10]. It is clear, that data structure separation is the foundation to support separating duties. Hence, these three approaches have the same sprit of data distribution. Based on data distribution or separation, other practical solutions could be promoted to protect privacy in the cloud.

These three approaches share another common characteristic: One CSP has only partial control of the user data, the whole control is managed by the user. In this way, privacy of the user is efficiently protected. Although there are still many problems to solve, these approaches provide a novel idea to eliminate the security and privacy concerns when deploying cloud services.

The potential drawback may be, that deploying more than one CSP will increase the related data management complexity and probability of availability accidents. The overview and classification of distribution based approaches is described in Figure 5.11.

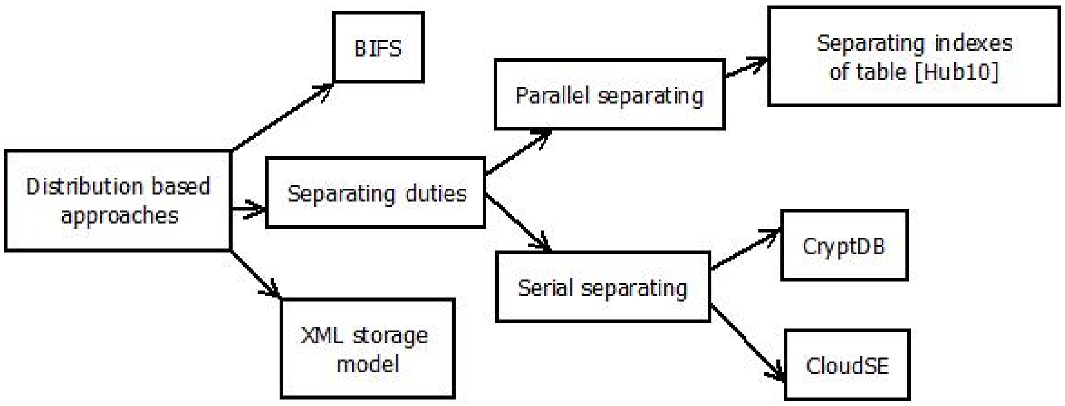


Figure 5.11: Overview of distribution based approaches

Chapter 6

Trust-based Approaches

The major obstacle for users to deploy Cloud services are security and privacy concerns. Challenges of deploying Cloud services have been discussed in Section 3.2. While users store data in the cloud, threats to privacy come from multiple aspects. Due to loss of data control and cloud transparency, privacy preservation in the cloud is a much harder task. Traditional data in a local network is protected by a set of security mechanisms, including firewalls, antivirus software, etc. There are already standards and frameworks, such as (ISO 27001:2005) and laws (e.g. Germanys Federal Data Protection Act) [PY13], that guide local IT infrastructure administrators to provide efficient security and privacy protection. Trust from user is built upon those security practices and full control of IT resources.

The infrastructure of some CSP may follow these standards, as for example AWS: "AWS has achieved ISO 27001 certification and has been validated as a Level 1 service provider under the Payment Card Industry (PCI) Data Security Standard (DSS)"[Ama14d]. The essential trust is though not completely built up. Third party audit is an efficient approach to enhance trust in the cloud. Though audit does not provide active control over the preservation of user privacy in the cloud environment to, it is a passive solution to ensure that privacy and security protection agreements promised by CSP are valid.

While users could put trust on third party auditing, an alternative for users is to put trust on hardware protection. By using Trusted Platform Module (TPM) to build a secure VM and further more a secure VM computing pool, users have more control over cloud computation environment. Hence efficient privacy preservation measures, such as encryption on server side, could achieve a higher security guarantee.

In this chapter, cloud audit and trust computing approaches are discussed in two sections. In each section, a classification of these approaches is given.

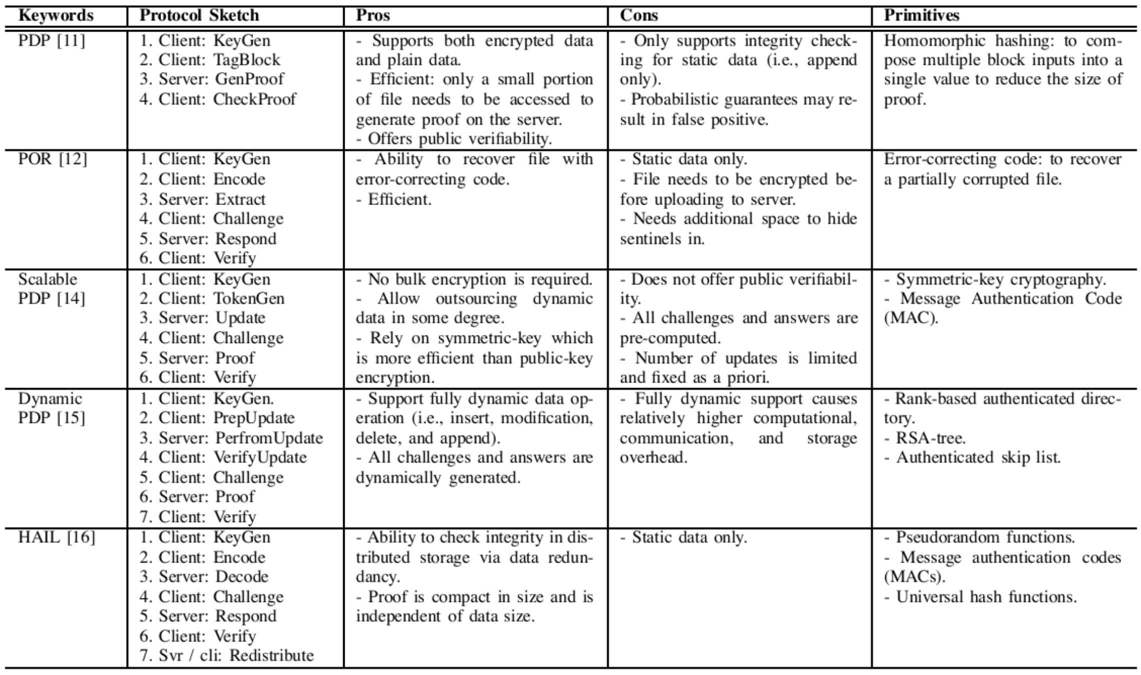
6.1 Cloud Audit

Cloud audit covers a wide range of activities, which also focus on the different aspects of cloud security. Current approaches could be classified into two categories: Integrity audit and security audit. Integrity audit focuses on privacy data integrity checks and security audit focuses on the monitoring of general security accidents.

6.1.1 Integrity Audit

Former approaches, such as data encryption and data distribution focus on the confidentiality of privacy. At the same time, integrity of privacy data is also of vital importance, because data loss and data manipulation in the cloud could result in great privacy damage.

The native approach that the user downloads the data and performs hash functions to check the data integrity is not acceptable. It is not only time and money consuming, furthermore is the complexity of such a task out of user tolerance. Remote integrity check has been deeply researched, several integrity check approaches has been discussed in [XX13]. A comparison table of these approaches is described as in Figure 6.1.



Keywords	Protocol Sketch	Pros	Cons	Primitives
PDP [11]	1. Client: KeyGen 2. Client: TagBlock 3. Server: GenProof 4. Client: CheckProof	- Supports both encrypted data and plain data. - Efficient: only a small portion of file needs to be accessed to generate proof on the server. - Offers public verifiability.	- Only supports integrity checking for static data (i.e., append only). - Probabilistic guarantees may result in false positive.	Homomorphic hashing: to compose multiple block inputs into a single value to reduce the size of proof.
POR [12]	1. Client: KeyGen 2. Client: Encode 3. Server: Extract 4. Client: Challenge 5. Server: Respond 6. Client: Verify	- Ability to recover file with error-correcting code. - Efficient.	- Static data only. - File needs to be encrypted before uploading to server. - Needs additional space to hide sentinels in.	Error-correcting code: to recover a partially corrupted file.
Scalable PDP [14]	1. Client: KeyGen 2. Client: TokenGen 3. Server: Update 4. Client: Challenge 5. Server: Proof 6. Client: Verify	- No bulk encryption is required. - Allow outsourcing dynamic data in some degree. - Rely on symmetric-key which is more efficient than public-key encryption.	- Does not offer public verifiability. - All challenges and answers are pre-computed. - Number of updates is limited and fixed as a priori.	- Symmetric-key cryptography. - Message Authentication Code (MAC).
Dynamic PDP [15]	1. Client: KeyGen. 2. Client: PrepUpdate 3. Server: PerfromUpdate 4. Client: VerifyUpdate 5. Client: Challenge 6. Server: Proof 7. Client: Verify	- Support fully dynamic data operation (i.e., insert, modification, delete, and append). - All challenges and answers are dynamically generated.	- Fully dynamic support causes relatively higher computational, communication, and storage overhead.	- Rank-based authenticated directory. - RSA-tree. - Authenticated skip list.
HAIL [16]	1. Client: KeyGen 2. Client: Encode 3. Server: Decode 4. Client: Challenge 5. Server: Respond 6. Client: Verify 7. Svr / cli: Redistribute	- Ability to check integrity in distributed storage via data redundancy. - Proof is compact in size and is independent of data size.	- Static data only.	- Pseudorandom functions. - Message authentication codes (MACs). - Universal hash functions.

Figure 6.1: Integrity check approaches comparison adapt from [XX13]

Approaches discussed in Figure 6.1 are performed by the user. When large numbers of documents and files are stored in the cloud, the related integrity checking task becomes a heavy burden to user. When users employ a third party to perform this integrity audit task, several new problems should be addressed. [WRLL10] points out that, "The Third Party Auditor (TPA) should be able to efficiently audit the cloud data storage without local copy of data and without any additional online burden for data owners. Besides, any possible leakage of an owners outsourced data toward a TPA through the auditing protocol should be prohibited". Considering the fact that Third Party Auditor (TPA) must provide data integrity auditing services in a multi-user environment and considering the dynamic nature of the cloud, requirements that Third Party Auditor (TPA) should fulfill are [WRLL10]:

- Minimize audit overhead: Performing an auditing task in the cloud should not cause much performance loss both for cloud users and the cloud provider.

- Protect data privacy: Data collected by Third Party Auditor (TPA) while performing audit should not cause user privacy leakage. And it should not introduce new vulnerabilities to the cloud environment[WRLL10].

- Support data dynamics: Audit protocol should allow data modification operations in the cloud. This feature may not be essential to certain users or application scenarios, but an audit protocol that supports cloud dynamic nature is of great flexibility.

- Support batch audit: Considering the fact, that a TPA has to provide audit services to multiple users, this feature will enhance the service efficiency and scalability.

An auditing service, that completely fulfills these requirements has not yet emerged. [WCW+13] promoted a *Privacy-Preserving Public Auditing system* to try to support these features. This approach deploys the integrity check solution promoted in [ABC+11], *Homomorphic Linear Authenticator*(HLA), which enables users to check data integrity without retrieving data copies from the cloud. Due to its mathematical characteristics, directly deploying HLA solutions may cause privacy leakages, detail of the analysis are covered in [WCW+13]. Hence, random masking technologies are being used to prevent privacy leakage to TPA. With the combination of other improvements of HLA as provable data processing (PDP), which are listed in the Figure 6.1, [WCW+13] it achieves to build an auditing system that fulfills the requirements discussed above. An preliminary experiment has been performed on Amazon EC2, the system efficiency is demonstrated by [WCW+13].

6.1.2 Security Audit

Most of the knowledge in section are acquired from book[PY13]. In this book, cloud audit and security aspects have been thoroughly discussed.

Comparing to integrity audits, security auditing monitors a large scope of security aspects. Security auditing comes from the old idea, IT auditing. Traditional IT auditing means: general control audits, application control audits, network/infrastructure audits and system development audits. In this situation, traditional IT audits belong to the general management and maintenance of an IT infrastructure. In the cloud, such auditing tasks are already performed by CSPs. Among these general auditing tasks, the focus of this thesis is security auditing. IT security auditing could be defined as the process of IT risk analysis and vulnerability assessment [PY13]. Traditional IT security auditing can be categorized into four types[PY13]:

- Vulnerability assessment: The vulnerabilities of the whole IT infrastructure are verified according to security best practices and standards.

- Vulnerability audit: The normal way of this audit type is to employ a third party to perform penetration tests. Hence potential vulnerabilities could be exposed.

- Application security audit: Application level security is focused. For example, the configuration weaknesses of web applications.

- Vulnerability management: A regular and automated vulnerability scan, documentation and so on.

Those security audit are performed by CSPs in their IT environment to achieve the goal of security protection of inner networks and inner applications. There are multiple current IT security audit standards to guide CSPs to secure their IT infrastructure, such as Attestation Engagements No.16 (SSAE 16). "SSAE16 is an AICPA auditing standard for reporting on controls at service organizations (including data centers) in the United States[PY13]". CSPs like Amazon have to get the certificates that those security standards have been achieved. "AWS has achieved ISO 27001 certification and has been validated as a Level 1 service provider under the Payment Card Industry (PCI) Data Security Standard (DSS). We undergo annual SOC 1 audits and have been successfully evaluated at the Moderate level for Federal government systems as well as DIACAP Level 2 for DoD systems."[Ama14d].

But for users, these audits only ensure that the CSP could provide proper protection against outside attacks and other normal security concerns. There are other problems not solved by this traditional IT audit. The challenges for privacy in Clouds have been discussed in Chapter3. In this section, challenges are discussed in detail. [PY13] summarizes privacy challenges in Cloud into six types:

- Misuse of Administrator Rights/Malicious Insiders
  This problem exists in traditional IT, "among 300 IT professionals, 26% admitted that at least one staff member has abused a privileged login to access information [PY13] "In the Cloud, this is also a big threat to user privacy.

- Missing Transparency of Applied Security Measures
  The user is not able to have a global view of the security mechanisms deployed in the cloud.

- Missing transparency of security incidents
  When security incidents happen to user data, users could be informed by their CSP. Considering the possibility that these information is not revealed to the user, the user will not be able to notice any security incidents related to his sensitive data.

- Shared Technology Issues: VM isolation: Memory/cache isolationI/O isolation
  Isolation failures may lead to privacy information leakage to other users or attackers.

- Data Life Cycle in Case of Provider Switch or Termination
  Cloud services may be suspended due to financial crisis of the CSP.

- Monitoring of Service-level agreements (such as the location agreement)
  SLA state is not able to be monitored by user, because related interfaces are missing.

According to the above challenges, new audit service should be built to provide following abilities [PY13]:

1. Privileged user access audit

2. Regulatory compliance
3. Data confidentiality, integrity, privacy, availability and segregation
4. Investigative support
5. Monitor and control of cloud services
6. Data retention

Several new auditing standards try to address this problem, such as CloudAudit A6, EuroCloud Star Audit, Cloud Controls Matrix by Cloud Security Alliance[PY13]. Those standards give plenty of security control suggestions, but did not yet give the possibility to allow user to directly monitor the accidents in the cloud.

For most of the problems listed above, the current solution is to address them in SLA. SLAs includes many aspects of the services, such as security protection levels, maximum data access threads, etc. The focus of this thesis lies in the security aspects. SSLAs are being used to address these concerns. Setting an SSLA down, does not essentialy mean that those agreements will be properly performed in the cloud, users need to be able to acquire the state of the execution of these agreements.

In this sense, security audit means: "Formal inspection and verification to check whether a standard or set of guidelines is being followed, records are accurate, or efficiency and effectiveness targets are being met[PY13]".

A new cloud auditing framework is promoted in[PY13], which utilizes autonomous agents to allow users to monitor the security accidents in the cloud and to solve the security challenges listed above. Security and privacy rules are specified in SSLAs and are adopt by audit agents as decision reference. The agent-based *Security Audit as a Service* architecture is described in Figure 6.2: In SAaaS architectures, autonomous agents are distributed and implemented in the key points of the cloud service architecture, such as . Accidents detected by audit agents are first locally assessed according to predefined rules and policies. Corresponding event reports are sent to processing module for further security verification. The whole SAaaS service state is illustrated in presentation modules. Once security accidents that violate related rules happen, users will be alerted.

To collect all the security accidents information, audit agents could cooperate with other existing security tools. Since Cloud architecture is dynamic, audit agents are also dynamically managed. The basic audit agent design and management is specified in Figure 6.3.

The SAaaS service is not yet fully functional, much work needs to be done to complement this framework. Using a fully mature SAaaS service, advantages could be specified in two aspects:

- For user:
  This allows users to be involved in the Cloud management. Even though auser has no direct control over the Cloud infrastructure, the user is enabled to monitor the Cloud operations.

- For CSPs:
  The audit service provides better overview over the Cloud infrastructure, hence

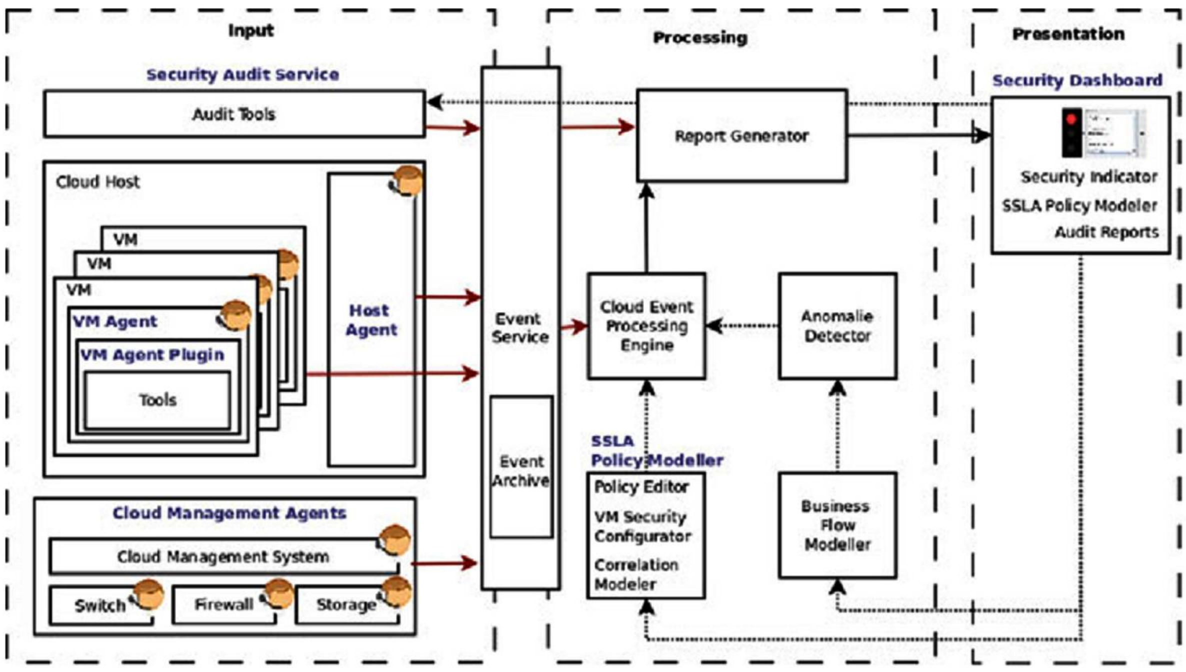


Figure 6.2: Processing sequence of Security Audit as a Service (SAaaS) [PY13]

security vulnerabilities could be observed and analyzed. Better security protection for its own IT infrastructure could be achieved. By allowing users to take part in the auditing process, trust to the cloud could be improved. Hence the potential cloud market will be expanded.

6.2 Trusted Computing

In this section, trusted computing foundation is specified at first. Then two major privacy preservation models are being discussed, Terra [GPC$^+$03] and Trusted Cloud Computing Platform (TCCP) [SGR09]. Terra focus on the privacy protection on single machines, while TCCP improves Terra by building a trusted computing pool.

6.2.1 Trusted Computing Foundation

Trusted computing is a concept promoted by Trusted Computing Group (TCG). The definition of trusted computing is because of various understandings from different aspects not standardized. The TCG definition of trusted computing focus on the expectation of the behavior of an certain entity. If the behavior always matches with the expectation, then this entity is trusted [Pea02]. From the user view: Trusted computing means, that services provided by the computer system are trustable, and the trust of computer system is able to be proofed and verified.

Despite of different understandings of trusted computing, the motivation of developing trusted computing is that software alone is not able to solve the security problem [Pea02]. Modern IT systems and applications are built upon open structure hardware. The hardware system openness together with the complexity of software make it im-

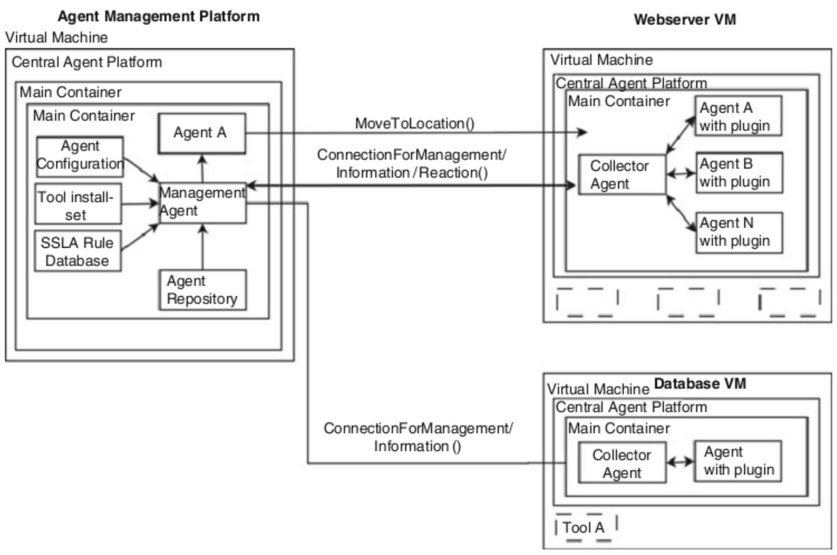


Figure 6.3: Agent management of Security Audit as a Service (SAaaS) adapt from [PY13]

possible to build a trusted system. To achieve a trusted computer system, security measure and control have to be established at the beginning of loading the system. This is achieved by implanting Trusted Platform Module (TPM) on the motherboard. TPM is a set of hardware that provides a trusted computation base. The components of TPM are described in Figure 6.4.

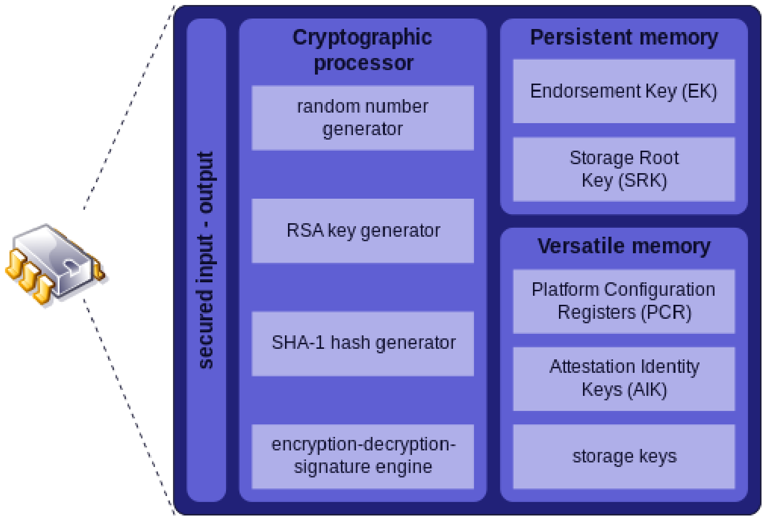


Figure 6.4: TPM components adapt from[wik14b]

From trust root, i.e. TPM, a trust chain is built up to the application software. Figure 6.5 show how to transfer trust and extend trust to application. A trust chain that extends to application software is current not possible to be built. Due to extreme complexity of the Operating System (OS), measuring its integrity and security is impractical. There is research going on on how to build a secure and trusted OS, *Next-Generation Secure Computing Base* from Microsoft [PCEM04]. But it is not mature yet. Another

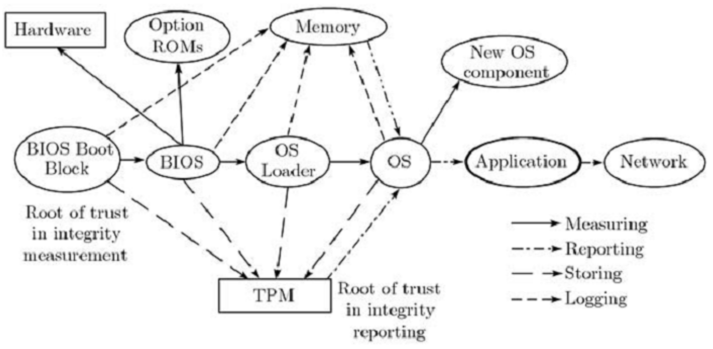


Figure 6.5: Trust chain transmit relation and process [SZW+10]

alternative is widely used in Cloud environment: Trust is extended to a Virtual Machine Monitor (VMM) instead of the OS. Detail of this approach is discussed in the next section.

TPM is the root of this trust chain, it is composed of three types of trust roots, the function of each trust root is specified as follows [Han]:

- Root of Trust for Storage (RTS): Refers to the component that is responsible for securely storing and managing sensitive data (integrity measurements, keys). The RTS is implemented by the TPM.
- Root of Trust for Reporting (RTR): Refers to the component that is responsible for creating verifiable integrity reports. The RTR is also implemented by the TPM.
- Root of Trust for Measurement (RTM): Refers to the component that is responsible for taking the initial measurements during system boot.

With the help of three trust roots and other hardware support, TPM provides the following abilities [wik14b]:

- Remote attestation: It allows remote third party to verify the hardware platform and software integrity.
- Binding: Data could be encrypted by using the endorsement key that is burned to the TPM chip, to allow data decryption only on certain platforms.
- Sealing: To make sure that data is only available when hardware platform and software are both verified.

An example that deploys trusted computing technology is Intel Trusted Execution Technology(Intel TXT). This hardware based technology is supposed to enhance the server platform security. Unlike traditional trusted computing techniques, that root of trust could be built only in static mode, Intel TXT allows systems to dynamically build a trust root and hence transform a system from an untrusted state to a trusted state [Int]. Another feature is that Intel TXT allows users to create a trusted computing pool, which contains a set of trusted VMs. Sensitive data is being processed in this trusted computing pool, VM migration in this pool is also protected. The detail working process of Intel TXT is described in Figure 6.6.

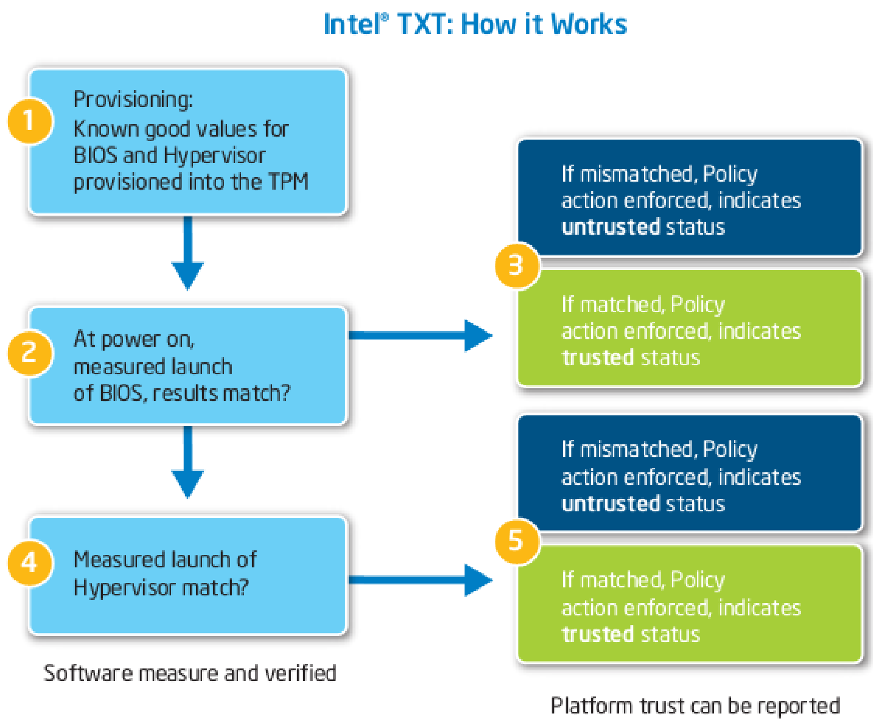

Intel® TXT: How it Works
1
Provisioning: Known good values for BIOS and Hypervisor provisioned into the TPM
2
At power on, measured launch of BIOS, results match?
3
If mismatched, Policy action enforced, indicates **untrusted** status
If matched, Policy action enforced, indicates **trusted** status
4
Measured launch of Hypervisor match?
5
If mismatched, Policy action enforced, indicates **untrusted** status
If matched, Policy action enforced, indicates **trusted** status
Software measure and verified
Platform trust can be reported

Figure 6.6: Intel trusted execution process [Int]

6.2.2 Terra and TCCP

Before the development of trusted computing, cryptographic coprocessors were being used to solve the secure computing problem on untrusted platforms. Based on cryptographic coprocessor, a Privacy as a Service approach has been promoted to protect user data privacy in the Cloud [IKC09]. This approach could be implemented by using acsTPM, because a crypto coprocessor is a component of TPM, see Figure 6.4.

The basic information of trusted computing is described in section6.2.1, there are two approaches described in this section, the first approach could solve single host security and privacy problems, and the second could solve the computing pool security problems, including the secure migrating of VMs to other hosts. A practical example of the second approach, intel TXT technology has already been successfully implemented to grant the computing pool security.

Single host protection [GPC+03]:
Considering the fact that improper VM isolation and memory isolation may result in privacy violation, a trusted computing based approach could achieve a secure computation environment in the cloud. The idea of Terra is to add one layer of Trusted Virtual Machine Monitor (TVMM), which partitions a tamper-resistant hardware platform (i.e., TPM together with other devices) into multiple isolated VMs. Traditional trusted computing could achieve a state where a trusted boot up environment is built, but to extend the trust chain to OS level is not achieved. The most significant reason is the complexity of commodity OS. It usually contains millions of lines of code [GPC+03], this huge number of code lines make it extremely difficult to assure that commodity OS is secure. The practical experience demonstrates the fact that a commodity OS is not able to provide a trusted environment. There are projects that try to build a much securer OS, for

example *Next-Generation Secure Computing Base* from Microsoft [PCEM04].

An alternative to provide a trusted OS is to provide a trusted acsVMM. Comparing to commodity OS, VMM has much less lines of code, this make it possible to achieve a trusted platform. "A VMM is a relatively simple program, with a narrow, stable, well-defined interface to the software running above it"[GPC+03].

In [GPC+03], TVMM is such a platform. VMs are running above this platform. For an applications that demand different security protections, they could run in different VMs. Privacy and security sensitive applications could run on a special designed, secure OS, which further more enhances the protection of these applications. Comparing to traditional application running environment, in which applications are isolated in process level, by using TVMM, applications are isolated in OS level, which could reduce security risks caused by other application compromises.

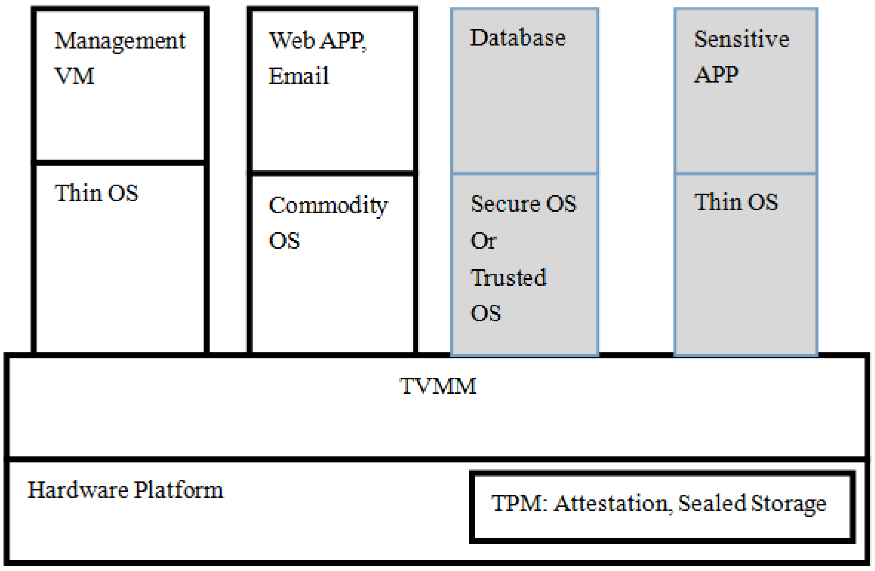


Figure 6.7: Basic structure of Terra according to [GPC+03]

In this way, applications which process sensitive privacy data are running in closed-box VMs. "Closed-box VMs are isolated from the rest of the platform.Through hardware memory protection and cryptographic protection of storage, their contents are protected from observation and tampering by the platform owner and malicious parties" [GPC+03]. This feature greatly solve the concerns regarding malicious system administrators.

Figure 6.7 describes several applications that run in different VMs on the TVMM platform. Besides these applications, a special module is *management VM*. This module could be designed as an special application that runs above TVMM or a normal application that runs in a *Thin OS* which runs above TVMM. This module could be used by CSP to perform the normal tasks. "The management VM formulates all platform access control and resource management policies. It grants access to peripherals, divides storage among VMs, and issues CPU and memory limits. It might formulate policies that limit how many VMs can run, which VMs can run (i.e. what software can run in a given VM), which VMs can access network interfaces or removable media, and so on. The management VM also starts, stops, and suspends VMs"[GPC+03]. By transferring these tasks to management VM, VMM could be a much simpler design and thus a minimal amount of bugs or security weaknesses will appear in this layer.

But in this architecture, TVMM has the root privilege, privacy and security policies are guaranteed by TVMM. Hence privacy and security policies design should only be based on TVMM. Other details could refer to [GPC+03].

Through Terra, user could achieve higher isolated and protected computation environment. But Terra is not a standard VMM that CSPs utilized (implemented) in their infrastructure. The root privilege is not controlled by the CSP, which introduces potential difficulties to the management of the Cloud. Another note is, that Terra only provides a trusted platform which enables users to securely process sensitive data above it, but additional security controls and mechanisms should be selected and implemented by the user.

Terra provides a good solution to provide a secure computation environment, but the problem is that it only works well on a single physical machine. The nature of the cloud is that, VMs are dynamically assigned and migrated among a large amount of physical machines. To provide practical protection or utilization of this technology, closed-box VMs should be available to migrate to other physical machines which provide Terra environment. An approach is promoted to allow secure VMs dynamically migrate to a trusted computing pool, which matches with the nature of Cloud.

Computing pool protection[SGR09]:

Trusted Cloud Computing Platform (TCCP) is promoted to provide a trusted computation pool environment to process sensitive data.

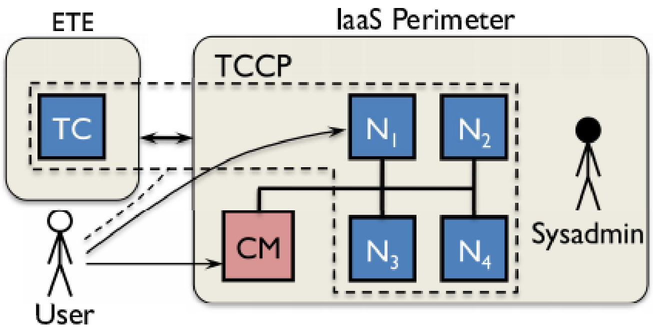


Figure 6.8: TCCP components adapt from [SGR09]

The basic idea of Trusted Cloud Computing Platform (TCCP) is, that all new computation nodes need to be verified by a TC (Trusted Coordinator) before the actual VM launch and migration begins. TC is management by external trusted party (ETE). The verification of new nodes is performed by utilizing remote attestation abilities enabled by TPM. This means that a set of secure physical machines which run Terra (TVMM) could be verified as potentialy trusted machines, they are recorded in the TC as a list, this set of trusted machines could be named as secure perimeter or trusted perimeter. When nodes are verified as trusted nodes, then nodes are granted to processing sensitive data. Otherwise, these nodes are not acceptable. This mechanism could efficiently prevent malicious system administrators from migrating previously protected closed-box VMs to unprotected platforms, and hence get access to sensitive data.

Note that, VM management is still the task performed by the CSP. In this TCCP architecture, CM refers to a VM management module, which is controlled by the CSP.

There are several actions in building and maintaining a trusted computation pool (secure perimeter), namely trusted node registration, VM launch, and VM migrate. For these three types of node actions, the processes are specified as follows:
Trusted node registration:

1. Node n sent challenge to TC to request the trust certificate of TC.
2. TC sends its bootstrap measurement encrypted with its endorsement key, which identifies the unique TPM. (The host of TC should also implement the TPM.)
3. If TC matches with security configurations, TC is trusted by node. Then node sends its bootstrap measurement encrypted with its own endorsement key, which represents the trust of node.
4. If node passes through the trust test, then it is added to a list that represents current trusted nodes (trusted computation pool). (reboot of TCCP will cause loss of the previous public/private key pairs that sent to TC at step 3, hence reboot of TCCP need to register to TC)

VM launch(based on the assumption that VMs are dynamically assigned by CSP):

1. VM image a and the hash of a is encrypted with a session key K_{vm}, session key is encrypted with the public key of TC. This message together is send to the VM management module of the cloud.
2. CM (VM management module) assigns a node N to host this VM image a. Node N send the message to TC to get the access to launch a VM image a.
3. Based on if node N belongs to the trusted computation pool, access of VM image a is allowed or denied.
4. Node N could launch the VM based on the access (decryption key of VM image) returned by TC.

VM migration, VM is currently running on Node A, the migration destination is Node B:

1. Node A sends a request to TC to verify if Node B is in the trusted computation pool.
2. If Node B is trusted, then send Node B a request to migrate VM. A session key for secure transfer of VM is also send to Node B.
3. Node N need to verify if Node A is trusted by send verification request to TC.
4. If node A is trusted by TC, then Node B could use the session key sent by Node A to receive VM image. (VM image is encrypted and a hash value is argumented to secure the confidentiality and integrity of VM).

5. VM could launch on Node B and keep its security and privacy.

Detail of the trust verification is done by open attestation technology enabled by TPM, it could refer to [SGR09].

The above model is a theoretical model, a practical example is Intel TXT, it is already a mature technology to secure VMs launch and migration on trusted nodes [Int].

[IKC09] provides a privacy as a service model, which allows privacy data storage and processing in the cloud. The basic design employs the development of cryptographic coprocessors. There are other related papers: [Pea02], [Mol], [Han]. They discussed other possible utilization of trusted computing technologies to secure network connecting and so on.

6.3 summary

Cloud auditing and trusted computing focus on different aspects of privacy preservation in the Cloud. Cloud auditing allows users to be aware of security and privacy accidents in the cloud. It is a passive approach to preserve privacy in the cloud. Trusted computing enables users to have more control over the data storage and processing in Cloud, which is an active approach as opposed to cloud auditing. These two approaches have in common that both need strong support from the CSP. Cloud auditing only demands CSPs to provide the interface for the auditing service, while trusted computing approach need CSPs to rebuild the service foundation based on the Trusted Cloud Computing Platform (TCCP) model discussed above. Considering the difficulty of rebuilding the service structure and the fact of losing root control of the VMM, CSPs may be not motivated to adapt the TCCP model. A trusted cloud service is essential to attract more customers, hence trusted computing idea could be deployed by CSPs to develop their own trusted architecture. To provide better overview of this two mechanisms that aim at improving the users trust in the cloud, Figure 6.9 describes the classification of approaches discussed in this chapter.

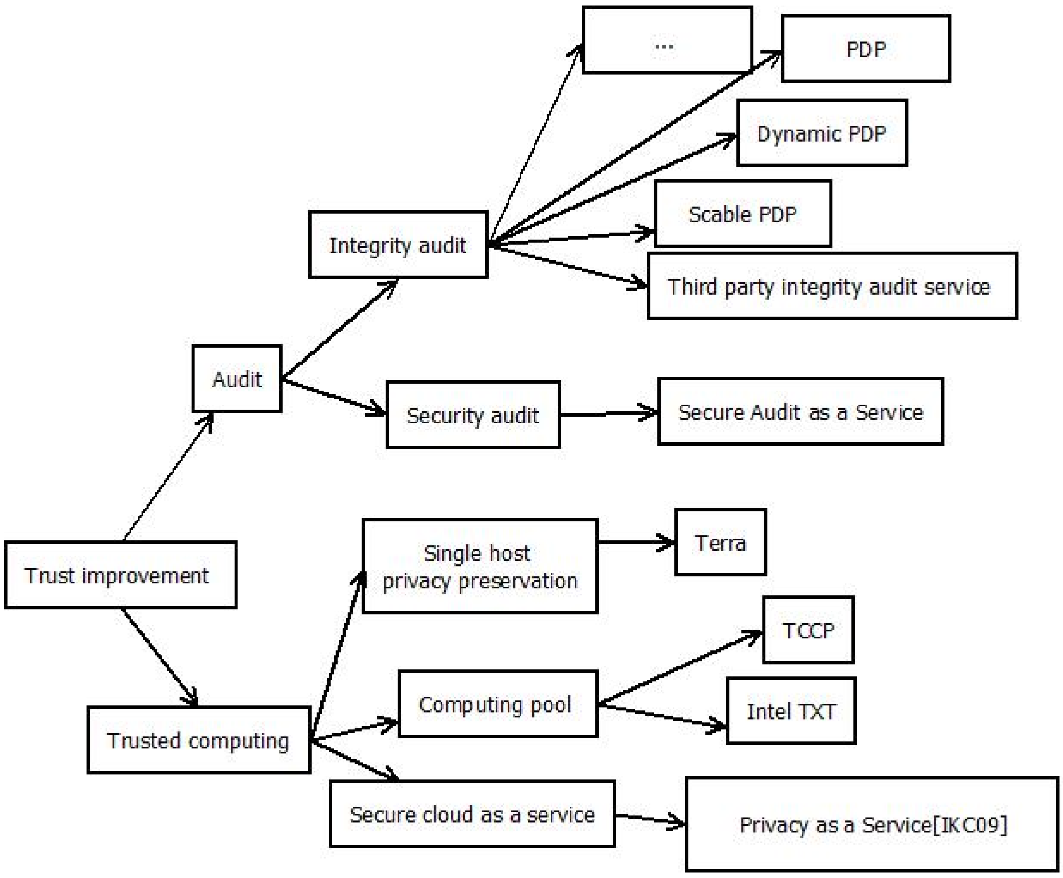

...
PDP
Dynamic PDP
Scable PDP
Integrity audit
Third party integrity audit service
Audit
Security audit
Secure Audit as a Service
Trust improvement
Single host privacy preservation
Terra
TCCP
Computing pool
Intel TXT
Trusted computing
Secure cloud as a service
Privacy as a Service[IKC09]

Figure 6.9: Overview of trust based approaches

Chapter 7

Conclusion and Future work

In this thesis, according to different privacy challenges, different approaches that aim at preserving privacy in the cloud were being discussed. There are four categories of approaches. A general classification privacy preserving mechanisms is described in Figure 7.1. Anonymity based approaches are listed in detail, because they are not separately

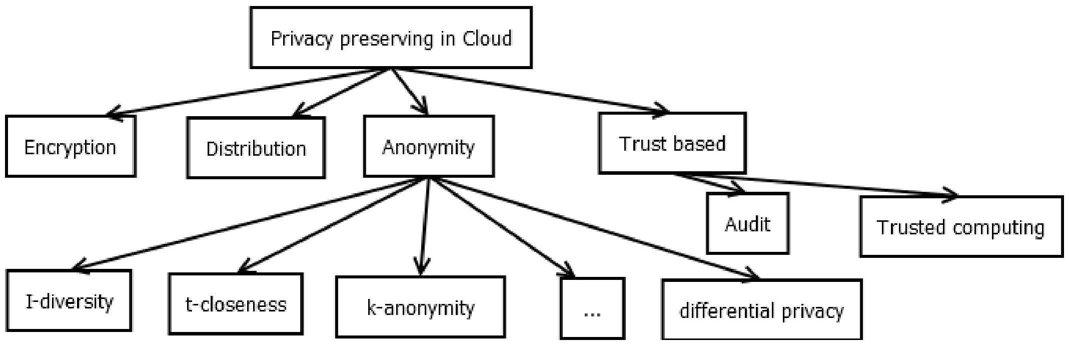


Figure 7.1: Overview of privacy preserving approaches

discussed in this thesis. Anonymity based approaches need to preprocess privacy data, the result has great information loss. Thus these approaches are suitable for data publishing scenarios. Other applications in the Cloud are limited to deploy anonymity based approaches. Overview and the detail classification of Encryption based approaches, Distribution based approaches and trust based approaches are described in the summary section of the corresponding chapter.

Figure 7.1 generally classifies privacy preserving approaches according to the characteristics of the employed techniques. According to different phases of deploying these approaches in cloud data life cycle management, these approaches were being classified as it is described in Figure 7.2

Most of these approaches focus on the *Create, Storage, Use* phase. This is natural, because these three phases are the most critical phases for preventing privacy violation in the cloud. Sharing data with others is a traditional difficulty for protecting data security and privacy. Privacy preserving mechanisms deployed by other users vary depending on their own IT system management. Sharing data in the cloud can be considered as a special case of sharing data with other users. Proper preprocessing of data is essential to protect sensitive information. From the perspective of the data recipient,

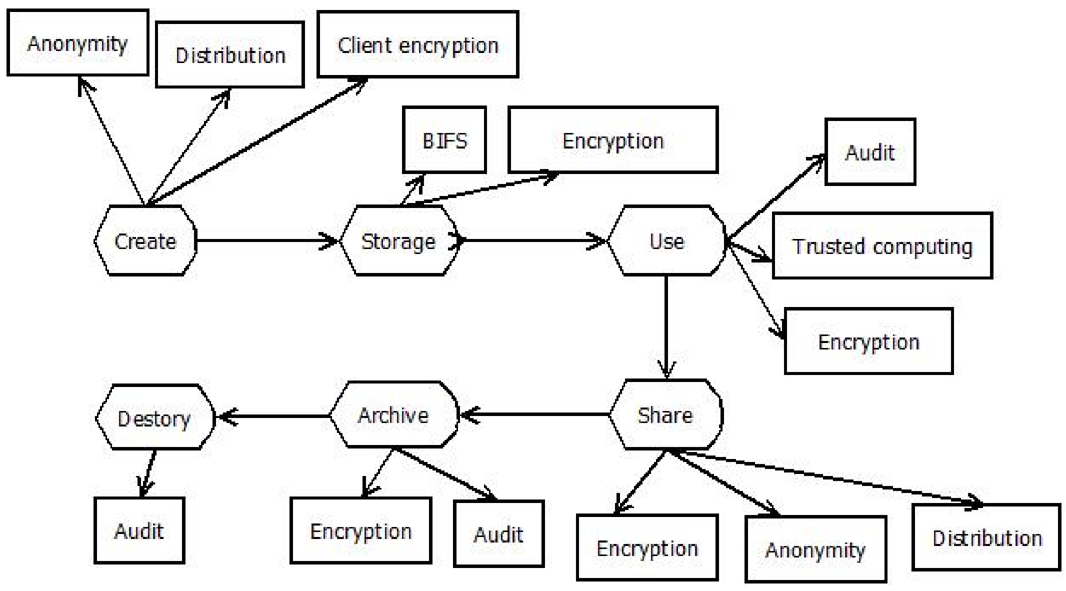


Figure 7.2: Classification of privacy preserving approaches according to data life cycle

data received from others could be considered as data creation phase. Hence approaches deployed in the data *Creation* phase could also be employed in the data *Share* phase. Data archiving means that data which is stored in the cloud is turned into an inactive state. Former privacy preservation approaches deployed on that data can be kept. To reduce management complexity, different approaches could be deployed, such as data compression and then encryption. The final phase of data destruction in the cloud is performed by the CSP. To prevent privacy leakage during this phase, Cloud audits can be employed.

Besides the fact that these approaches are deployed in different phases of the Cloud data life cycle, there are other differences which need to be discussed. Most of approaches focus on privacy data confidentiality protection, while cloud audit focuses on privacy data integrity protection. According to their application scenarios and other aspects, such as performance and CSP support, The comparison of advantages and disadvantages of these three major categories of approaches are listed below:

Encryption based approaches:

- Advantage: The main advantage of encryption based approaches is, that encryption of data provides efficient data confidentiality assurance and hence efficient data privacy assurance. Client side encryption has the most efficient protection against the privacy violation in the cloud, while server side encryption could achieve better performance and management compared to client side encryption.
- Disadvantage: Performance loss when deploying data encryption in the cloud is not avoidable. When selecting between employing client side encryption or server side encryption, the exact performance loss needs to be estimated by the user himself. When employing client side encryption, encryption key management should be properly performed by the user. When employing server

side encryption, key management and encryption/decryption are performed by the CSP. There are still potential privacy problems not solved, i.e. malicious administrators. Trusted computing needs to be combined with server side encryption to achieve better privacy protection.

Distribution based approaches:

- Advantage: These novel approach try to enable the user to have more control of privacy data in the cloud. Comparing to data encryption, data distribution demands no complex key management. Since data is being distributed among multiple CSPs and trusted third parties, better privacy preservation can be achieved, since attackers need to get access to all the essential information to rebuild privacy data. From the academic perspective, distribution based approaches provide a novel idea of protecting data privacy in the cloud.
- Disadvantage: Approaches in this category are not as mature yet as other approaches, such as encryption and trusted computing. When employing a separating duties approach, a non-standard or uniform schema could be employed. Users need to analyze their own data structures, privacy requirements and business flow to develop corresponding solutions. This consumes large amounts of human resource, time and monetary investment, which may be inacceptable for users.

Audit and trusted computing:

- Advantage: Audit is a passive way to enforce privacy protection in the cloud. The SSLA is the most important regulation between user and CSP, auditing is the measure to make sure that SSLAs are being properly installed and performed. Trusted computing builds a secure environment on top of the hardware base, a secure VM and a computing pool with the help of the key management and encryption/decryption in hardware, a proper privacy protection is hence achieved. By using trusted computing in the cloud, performance will be improved from the user perspective.
- Disadvantage: The main disadvantage of cloud auditing and trusted computing is that they need strong support from CSPs. Considering of multiple concerns, such as exposure of inner IT structure to third parties and much cloud architecture modification, CSPs may not be willing to deploy these approaches.

Different categories of approaches are often employed together to achieve better privacy preservation. Due to the complexity of modern systems, exact and proper classification of each approach is not possible. The above classification of privacy preserving approaches is based on their common characteristics. There are other privacy preserving technologies that have not been discussed in this thesis, such as VPNs (virtual private networks), which allows user to have more control over the cloud environment. These technologies can efficiently improve the confidentiality and isolation level of Cloud user data, and hence user privacy can be protected better.

Generally speaking, to provide better privacy protection in the cloud, the user needs to have more control over data. This conflicts with the nature that the cloud controls user data. In a cloud service model, the deeper model users deployed, the more control users have, so users need to choose proper cloud service provider and service model depending on their applications.

There are many privacy preservation approaches discussed in this thesis, they suit for different scenarios. The critical problem is that, there is no mature model or standard to cover all the privacy problems. Current mechanisms are developed based on security and privacy protection best practices. For each CSP, some of the standard approaches are implemented, such as encryption. But the other approaches, such as trusted computing and cloud auditing are still models and not widely deployed by them. As a conclusion, a general privacy preservation framework should be developed when designing and developing cloud services. This framework should not only address technical aspects, but also standards and regulations of privacy preservation in the cloud.

Bibliography

[AAW13] Agrawal, D.; Abbadi, A. E.; Wang, S.: Secure and privacy-preserving database services in the cloud. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, S. 1268–1271, 2013.

[ABC$^+$11] Ateniese, G.; Burns, R.; Curtmola, R.; Herring, J.; Khan, O.; Kissner, L.; Peterson, Z.; Song, D.: Remote data checking using provable data possession. *ACM Transactions on Information and System Security (TISSEC)*, Band 14, Nr. 1, S. 12, 2011.

[AEAW12] Agrawal, D.; El Abbadi, A.; Wang, S.: Secure and privacy-preserving data services in the cloud: a data centric view. *Proceedings of the VLDB Endowment*, Band 5, Nr. 12, S. 2028–2029, 2012.

[AEKR] Arasu, A.; Eguro, K.; Kaushik, R.; Ramamurthy, R.: Querying encrypted data. In *2013 IEEE International Conference on Data Engineering (ICDE 2013)*, S. 1262–1263.

[Ama14a] Amazon: Amazon web services documentation: Amazon relational database service. http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html, 21.02.2014.

[Ama14b] Amazon: Amazon web services documentation: Amazon simple storage service. http://docs.aws.amazon.com/AmazonS3/latest/dev/Welcome.html, 21.02.2014.

[Ama14c] Amazon: Amazon web services documentation: Getting started with aws cloudhsm. http://docs.aws.amazon.com/cloudhsm/latest/gsg/cloud-hsm-get-started.html, 21.02.2014.

[Ama14d] Amazon: Aws security center. http://aws.amazon.com/security/, 21.02.2014.

[Apa14] Apache: The apache hbase reference guide. http://hbase.apache.org/book.html, 21.02.2014.

[Bar] Barth, D.: Google cloud storage now automatically encrypts all data — zdnet. http://googlecloudplatform.blogspot.de/2013/08/google-cloud-storage-now-provides.html.

[BB08] Battle, R.; Benson, E.: Bridging the semantic web and web 2.0 with representational state transfer (rest). *Web Semantics: Science, Services and Agents on the World Wide Web*, Band 6, Nr. 1, S. 61–69, 2008.

[BDCOP04] Boneh, D.; Di Crescenzo, G.; Ostrovsky, R.; Persiano, G.: Public key encryption with keyword search. In *Advances in Cryptology-Eurocrypt 2004*, S. 506–522. Springer, 2004.

[BF01] Boneh, D.; Franklin, M.: Identity-based encryption from the weil pairing. In *Advances in CryptologyCRYPTO 2001*, S. 213–229. Springer, 2001.

[BGG$^+$01] Brundrett, P.; Garg, P.; Gu, J.; Kelly, J.; Kaplan, K.; Reichel, R.; Andrew, B.; Kimura, G.; Miller, T.: Encrypting file system and method. http://www.google.com/patents/US6249866, Juni 19 2001. US Patent 6,249,866.

[BM$^+$11] Brunette, G.; Mogull, R. et al.: Security guidance for critical areas of focus in cloud computing v3.0. *Cloud Security Alliance*, S. 1–176, 2011.

[BSNS08] Baek, J.; Safavi-Naini, R.; Susilo, W.: Public key encryption with keyword search revisited. In *Computational Science and Its Applications–ICCSA 2008*, S. 1249–1259. Springer, 2008.

[CSA13] CSA: Cloud Vulnerabilities Working Group: *Cloud Computing Vulnerability Incidents: A Statistical Overview ulnerability Incidents: A Statistical Overview*. 13.03.2013.

[DAT95] DATA, M. O. S.: Directive 95/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. *Official Journal L*, Band 281, Nr. 23/11, S. 0031–0050, 1995.

[DLKY04] Diament, T.; Lee, H. K.; Keromytis, A. D.; Yung, M.: The dual receiver cryptosystem and its applications. In *Proceedings of the 11th ACM conference on Computer and communications security*, S. 330–343. ACM, 2004.

[EED08] El Emam, K.; Dankar, F. K.: Protecting privacy using k-anonymity. *Journal of the American Medical Informatics Association*, Band 15, Nr. 5, S. 627–637, 2008.

[enc14] Top 5 free encryption tools to protect your data stored in the cloud. http://www.nextofwindows.com/top-3-free-encryption-tools-to-protect-your-data-stored-in-the-cloud/, 21.02.2014.

[ES12] Eike Schallehn, G. S.: Cloud data management. Lecture, 2012. Powerpoint slides.

[G$^+$03] Goh, E.-J. et al.: Secure indexes. *IACR Cryptology ePrint Archive*, Band 2003, S. 216, 2003.

[GEL] Gholami, A.; Edlund, A.; Laure, E.: Cloud privacy threat modelling.

[gooa] google: Case study: Dnanexus.

[goob] google: Google's approach to it security: A google white paper.

[Goo14] Google: Cloud storage api. https://developers.google.com/storage/docs/xml-api-overview, 21.02.2014.

[GPC+03] Garfinkel, T.; Pfaff, B.; Chow, J.; Rosenblum, M.; Boneh, D.: Terra: A virtual machine-based platform for trusted computing. In *ACM SIGOPS Operating Systems Review*, S. 193–206. ACM, 2003.

[Gro] Group, N. W.: Cloudaudit 1.0. http://tools.ietf.org/html/draft-hoff-cloudaudit-00.

[GWD14] Guo, L.; Wang, J.; Du, H.: Xml privacy protection model based on cloud storage. *Computer Standards & Interfaces*, Band 36, Nr. 3, S. 454–464, 2014.

[had14] What is hadoop. http://www.edureka.in/blog/what-is-hadoop/, 21.02.2014.

[Han] Hannover, T. H.: Trusted computing and trusted network connect in a nutshell. http://trust.f4.hs-hannover.de/2008/11/21/trusted-computing-and-trusted-network-connect-in-a-nutshell.html.

[HHK+] Henrich, C.; Huber, M.; Kempka, C.; Müller-Quade, J.; Ralf, R.: Technical report: Secure cloud computing through a separation of duties.

[HHKMQ] Henrich, C.; Huber, M.; Kempka, C.; Müller-Quade, J.: Towards secure cloud computing through a separation of duties.

[HILM02] Hacigümüş, H.; Iyer, B.; Li, C.; Mehrotra, S.: Executing sql over encrypted data in the database-service-provider model. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, S. 216–227, 2002.

[Hub10] Huber, M.: Towards secure services in an untrusted environment. In *Fifteenth International Workshop on Component-Oriented Programming*, S. 47, 2010.

[IBM14] IBM: Ibm db2 encryption. http://pic.dhe.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.fresh.doccom.ibm.db2.luw.admin.sec.doc/doc/c0005815.html, 21.02.2014.

[IKC09] Itani, W.; Kayssi, A.; Chehab, A.: Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures. In *Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on*, S. 711–716, 2009.

[Int] Intel: Intel trusted execution technology white paper.

[Jan] Jana Dittmann, Claus Vielhauser, Christian Kraetzer: Mulimedia and security.

[Liv] Livshits, A.: The cloud storage engine (clouse) - oblaksoft. www.oblaksoft.com/docs/clouse.pdf.

[LLV07] Li, N.; Li, T.; Venkatasubramanian, S.: t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, S. 106–115, 2007.

[LWW09] Liu, Q.; Wang, G.; Wu, J.: An efficient privacy preserving keyword search scheme in cloud computing. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, S. 715–720. IEEE, 2009.

[LYCL11] Li, M.; Yu, S.; Cao, N.; Lou, W.: Authorized private keyword search over encrypted data in cloud computing. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, S. 383–392. IEEE, 2011.

[Mey] Meyer, D.: Nsa's backdoor catalog exposed: Targets include juniper, cisco, samsung, huawei. http://gigaom.com/2013/12/29/nsas-backdoor-catalog-exposed-targets-include-juniper-cisco-samsung-and-huawei/.

[MG11] Mell, P.; Grance, T.: The nist definition of cloud computing (draft). *NIST special publication*, Band 800, Nr. 145, S. 7, 2011.

[Mic14a] Microsoft: How efs works. http://technet.microsoft.com/en-us/library/cc962103.aspx, 21.02.2014.

[Mic14b] Microsoft: Transparent data encryption in sqlserver. http://msdn.microsoft.com/en-us/library/bb934049.aspx, 21.02.2014.

[MKL09] Mather, T.; Kumaraswamy, S.; Latif, S.: *Cloud security and privacy: an enterprise perspective on risks and compliance.* O'Reilly, 2009.

[Moh11] Mohammad, S.: A survey and classification of data management research approaches in the cloud. Master's thesis, Unversitaet Magdeburg, 2011.

[Mol] Molina, J.: Practical applications of trusted computing in the cloud.

[NGSN10] Narayan, S.; Gagné, M.; Safavi-Naini, R.: Privacy preserving ehr system using attribute-based infrastructure. In *Proceedings of the 2010 ACM workshop on Cloud computing security workshop*, S. 47–52, 2010.

[Nin12] Ning Cao: *Secure and Reliable Data Outsourcing in Cloud Computing.* Dissertation, Worcester Polytechnic Institute, 2012.

[NS13] Neela, T. J.; Saravanan, N.: Privacy preserving approaches in cloud: a survey. *Indian Journal of Science and Technology*, Band 6, Nr. 5, S. 4531–4535, 2013.

[Ora14] Oracle: Oracle database advanced security administrator's guide, 21.02.2014.

[PCEM04] Peinado, M.; Chen, Y.; England, P.; Manferdelli, J.: Ngscb: A trusted open system. In *Information Security and Privacy*, S. 86–97. Springer, 2004.

[Pea02] Pearson, S.: Trusted computing platforms, the next security solution. *HP Labs*, 2002.

[PH10] Pfitzmann, A.; Hansen, M.: A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2010.

[PRZB11] Popa, R. A.; Redfield, C.; Zeldovich, N.; Balakrishnan, H.: Cryptdb: protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, S. 85–100. ACM, 2011.

[PY13] Pearson, S.; Yee, G.: *Privacy and security for cloud computing*. Computer communications and networks. Springer-Verlag, New York, 2013.

[San96] Sandhu, R.: Access control: The neglected frontier. In *Information Security and Privacy*, S. 219–227. Springer, 1996.

[SBM$^+$07] Shah, M. A.; Baker, M.; Mogul, J. C.; Swaminathan, R. et al.: Auditing to keep online storage services honest. In *HotOS*, 2007.

[Sen] Sensenbrenner, J.: This abuse of the patriot act must end. http://www.theguardian.com/commentisfree/2013/jun/09/abuse-patriot-act-must-end.

[SGR09] Santos, N.; Gummadi, K. P.; Rodrigues, R.: Towards trusted cloud computing. In *Proceedings of the 2009 conference on Hot topics in cloud computing*, S. 3, 2009.

[SMGL11] Sheng, Z.; Ma, Z.; Gu, L.; Li, A.: A privacy-protecting file system on public cloud storage. In *Cloud and Service Computing (CSC), 2011 International Conference on*, S. 141–149. IEEE, 2011.

[SS98a] Samarati, P.; Sweeney, L.: Generalizing data to provide anonymity when disclosing information. In *PODS*, S. 188, 1998.

[SS98b] Samarati, P.; Sweeney, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression, 1998.

[SS13] Sharon, S. E.; Saravanan, N.: A survey on keyword based search over outsourced encrypted data. *International Journal of Engineering & Technology (0975-4024)*, Band 5, Nr. 2, 2013.

[Sto10] Stonebraker, M.: Sql databases v. nosql databases. *Commun. ACM*, Band 53, Nr. 4, S. 10–11, April 2010.

[Swe02a] Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Band 10, Nr. 05, S. 571–588, 2002.

[Swe02b] Sweeney, L.: k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Band 10, Nr. 05, S. 557–570, 2002.

[SWP00] Song, D. X.; Wagner, D.; Perrig, A.: Practical techniques for searches on encrypted data. In *Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on*, S. 44–55. IEEE, 2000.

[SZ09] Shuigeng Zhou, Y. T. X. X., F. l.: Review of privacy protection oriented database applications. *Journal of computers*, Band 32, Nr. 5, S. 847–861, 2009.

[SZW$^+$10] Shen, C.; Zhang, H.; Wang, H.; Wang, J.; Zhao, B.; Yan, F.; Yu, F.; Zhang, L. et al.: Research on trusted computing and its development. *Science China Information Sciences*, Band 53, Nr. 3, S. 405–433, 2010.

[TWZ09] Tian, X.; Wang, X.; Zhou, A.: Dsp re-encryption: A flexible mechanism for access control enforcement management in daas. In *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on*, S. 25–32. IEEE, 2009.

[vii14] Viivo to easily encrypt and secure your cloud data in dropbox. http://www.nextofwindows.com/viivo-to-easily-encrypt-and-secure-your-cloud-data-in-dropbox/, 21.02.2014.

[WAEA11] Wang, S.; Agrawal, D.; El Abbadi, A.: A comprehensive framework for secure query processing on relational data in the cloud. In *Secure Data Management*, S. 52–69. Springer, 2011.

[WCKM09] Wong, W. K.; Cheung, D. W.-l.; Kao, B.; Mamoulis, N.: Secure knn computation on encrypted databases. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, S. 139–152. ACM, 2009.

[WCL$^+$10] Wang, C.; Cao, N.; Li, J.; Ren, K.; Lou, W.: Secure ranked keyword search over encrypted cloud data. In *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, S. 253–262. IEEE, 2010.

[WCW$^+$13] Wang, C.; Chow, S. S.; Wang, Q.; Ren, K.; Lou, W.: Privacy-preserving public auditing for secure cloud storage. *Computers, IEEE Transactions on*, Band 62, Nr. 2, S. 362–375, 2013.

[wik14a] Kerberos-(protocol). http://en.wikipedia.org/wiki/Kerberos-(protocol), 21.02.2014.

[wik14b] wiki: Trusted platform module. `http://en.wikipedia.org/wiki/Trusted_Platform_Module`, 21.02.2014.

[Wik14c] Wikipedia: Amazon web services - wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?oldid=591702982, 21.01.2014.

[WRLL10] Wang, C.; Ren, K.; Lou, W.; Li, J.: Toward publicly auditable secure cloud data storage services. *Network, IEEE*, Band 24, Nr. 4, S. 19–24, 2010.

[XX13] Xiao, Z.; Xiao, Y.: Security and privacy in cloud computing. *Communications Surveys & Tutorials, IEEE*, Band 15, Nr. 2, S. 843–859, 2013.

[YL12] Yang Li, G. X. e., W. W.: Summary of differential privacy protection. *Computer application*, Band 29, Nr. 9, S. 3201–3205, 2012.

Selbststaendigkeitserklaerung

Hiermit erklaere ich, dass ich die vorliegende Arbeit selbststaendig und nur mit erlaubten Hilfsmitteln angefertigt habe.

Magdeburg, den April 2, 2014

Vorname Name