Achieving Trust-oriented Data Protection in the Cloud Environment

A thesis submitted in fulfilment of the requirements for

the degree of Doctor of Philosophy

ARN laboratory, iNext research centre

University of Technology, Sydney

by

Lingfeng Chen

Supervised by

Professor Doan B. Hoang

2014

DEDICATION

To my Parents and my Grandparents

To my Wife and my Parents-in-Law

Thank you for your love and support

ACKNOWLEDGEMENTS

I sincerely express my deepest gratitude to my principal supervisor, Professor Doan B. Hoang, for his supervision and continuous encouragement throughout my whole PhD study. His guidance, wisdom, and enthusiasm have made me both more mature as a person and more confident to be a good researcher. He has been outstanding in providing insightful feedback and creating the perfect balance of my research engagement and my casual teaching work. From the beginning of determining the research direction to publishing fruitful research outcome, he always commits to foster my research skills. Without his guidance, I would still be in the marsh of research career. Further, I thank him for offering me a number of research grants that were really helpful to support my research and study. I feel so fortunate to have him as my supervisor for the past three and half years.

I thank the University of Technology Sydney for offering me an IRS scholarship throughout my doctoral program. I also thank the iNext research centre for providing valuable resources, including funding for attending conferences. Special thanks go to the School of Computing and Communication at the University of Technology Sydney for offering me the teaching internship that has significantly increased my teaching experience in academia. Thanks to these funding and support, that makes me could concentrate on my research work without the burden of living.

My thanks also go to staff members and research students in the ARN lab for their help, suggestions, friendship and encouragement: special thanks to Fatima Furqan, Dr. Minh Hoang Phung, Dr. Venki Balasubramanian, Najmeh Kamyabpour, Nor Faizah Ahmad.

Furthermore, I thank my parents for their upbringing and encouragement to succeed in my study and my grandparents for their meticulous care and attention. Then, I express my gratitude and appreciation to my wife for her love and support. Her selfless sacrifices and commitment to the family, made it possible for me to finish my PhD studies. Finally, I would like to thank my brother, my parents and brother-in-law for their encouragement, and thank all the people who helped me and contributed to this study.

CERTIFICATE OF ORIGINAL AUTHORSHIP

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Student:

Date:

THE AUTHOR'S PUBLICATIONS

International Conference Publications and Proceedings:

CHEN, L. & HOANG, D. B. 2013. Adaptive data replicas management based on active data-centric framework in cloud environment. 2013 IEEE International conference on High Performance Computing and Communications (HPCC). (ERA ranking: B)

CHEN, L. & HOANG, D. B. 2013. Addressing data and user mobility challenge in the cloud. 2013 IEEE International conference on Cloud Computing (CLOUD), 549-556. (ERA ranking: B)

CHEN, L. & HOANG, D. B. 2012. Active data-centric framework for data protection in cloud environment. 2012 23rd Australasian Conference on Information Systems (ACIS), 1-11. (ERA ranking: A)

CHEN, L. & HOANG, D. B. 2011. Towards scalable, fine-grained, intrusion-tolerant data protection models for healthcare cloud. 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 126-133. (ERA ranking: A)

CHEN, L. & HOANG, D. B. 2011. Novel data protection model in healthcare cloud. 2011 IEEE International Conference on High Performance Computing and Communications (HPCC), 550-555. (ERA ranking: B)

HOANG, D. B. & CHEN, L 2010. Mobile Cloud for Assistive Healthcare (MoCAsH). 2010 IEEE International Conference on Asia-Pacific Services Computing Conference (APSCC), 6-10 Dec. 2010. 325-332. (ERA ranking: C)

ABSTRACT

Cloud computing has gained increasing acceptance in recent years. In privacy-conscious domains such as healthcare and banking, however, data security and privacy are the greatest obstacles to the widespread adoption of cloud computing technology. Despite enjoying the benefits brought by this innovative technology, users are concerned about losing the control of their own data in the outsourced environment. Encrypting data can resolve confidentiality and integrity challenges, but the key to mitigating users' concerns and encouraging broader adoption of cloud computing is the establishment of a trustworthy relationship between cloud providers and users.

In this dissertation, we investigate a novel trust-oriented data protection framework adapted to the cloud environment. By investigating cloud data security, privacy, and control related issues, we propose a novel data protection approach that combines active and passive protection mechanisms. The active protection is used to secure data in an independent and smart data cube that can survive even when the host is in danger. The passive protection covers the actions and mechanisms taken to monitor and audit data based on third party security services such as access control services and audit services. Furthermore, by incorporating full mobility and replica management with the active and passive mechanisms, the proposed framework can satisfy *confidentiality*, *integrity*, *availability*, *scalability*, *intrusion-tolerance*, *authentication*, *authorization*, *auditability*, and *accountability*, increasing users' confidence in consuming cloud-based data services.

In this work we begin by introducing cloud data storage characteristics and then analyse the reasons for issues of data security, privacy and control in cloud. On the basis of results of analysis, we identify desirable properties and objectives for protecting cloud data. In principle, cryptography-based and third party based approaches are insufficient to address users' concerns and increase confidence in consuming cloud-based data services, because of possible intrusion attacks and direct tampering of data. Hence, we propose a novel way of securing data in an active data cube (ADCu) with smart and independent functionality. Each ADCu is a deployable data protection unit encapsulating sensitive data, networking, data manipulation, and security verification functions within a coherent data structure. A sealed and signed ADCu encloses dynamic information-flow tracking throughout the data cube that can precisely monitor the inner data and the derivatives. Any violations of policy or tampering with data would be compulsorily recorded and reported to bundled users via the mechanisms within the ADCu. This active and bundled architecture is designed to establish a trustworthy relationship between cloud and users.

Subsequently, to establish a more comprehensive security environment cooperating with an active data-centric (ADC) framework, we propose a cloud-based privacy-aware role-based access control (CPRBAC) service and an active auditing service (AAS). These components in the entire data protection framework contribute to the passive security mechanisms. They provide access control management and audit work based on a consistent security environment. We also discuss and implement full mobility management and data replica management related to the ADCu, which are regarded as significant factors to satisfy data accountability, availability, and scalability.

We conduct a set of practical experiments and security evaluation on a mini-private cloud platform. The outcome of this research demonstrates the efficiency, feasibility, dependability, and scalability of protecting outsourced data in cloud by using the trust-oriented protection framework. To that end, we introduce an application applying the components and mechanisms of the trust-oriented security framework to protecting eHealth data in cloud.

The novelty of this work lies in protecting cloud data in an ADCu that is not highly reliant on strong encryption schemes and third-party protection schemes. By proposing innovative structures, concepts, algorithms, and services, the major contribution of this thesis is that it helps cloud providers to deliver trust actively to cloud users, and encourages broader adoption of cloud-based solutions for data storage services in sensitive areas.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AA	Active Auditor
AAS	Active Auditing Service
ABAC	Attribute Based Access Control
ACID	Atomicity, Consistency, Isolation, and Durability
ACL	Access Control List
ADC	Active Data-centric
ADCu	Active Data Cube
ADNI	Active Data Network Information
AES	Advanced Encryption Standard
AF	Access Frequency
AP	Access Point
AS	Attribute Set
ATC	Average Time Consumption
CIA	Cloud Information Accountability
CPRBAC	Cloud-based Privacy-aware Role-Based Access Control
CR	Consistency Requirement
CR	Concurrent Requests
CSPs	Cloud Service Provider
CT	Current Time
DAC	Discretionary Access Control
Ddes	Data Descriptor
DMM	Data Mobility Management
DP	Data Permission

DPA	Data Permission Assignment
EHR	Electronic Health Record
EPAL	Enterprise Privacy Authorization Language
ES	Executable Segment
GCM	Google Cloud Messaging
GUID	Globally Unique Identifier
HTTP	Hypertext Transfer Protocol
HWNE	Heterogeneous Wireless Network Environment
IaaS	Infrastructure as a Service
JAR	Java Archive
JDBC	Java Database Connectivity
JRE	Java Runtime Environment
JVM	Java Virtual Machine
LSP	Location Service Provider
MAC	Mandatory Access Control
MQTT	Message Queue Telemetry Transport
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OC	Operation Cost
ODBC	Open Database Connectivity
OS	Operation System
PaaS	Platform as a Service
PDP	Provable Data Possession
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PI	Pervasive Informing

PK	Pervasive Knowing
PL	Participants List
POR	Proof Of Retrievability
QoS	Quality of Service
RA	Role Assignment
RAID	Redundant Array of Independent Disks
RBAC	Role Based Access Control
RESTFUL	Representational State Transfer
RMI-SSL	Remote Method Invocation over Secure Socket Layer
SaaS	Software as a Service
SAN	Storage Area Networks
SAML	Security Assertion Markup Language
SID	Security Identifier
SLA	Service Level Agreement
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SSH	Secure Shell
SSL	Secure Socket Layer
TDFS	Triggerable Data File Structure
TPM	Trusted Platform Module
TCG	Trusted Computing Group
TLS	Transport Layer Security
TTL	Time-To-Live
UF	Update Frequency
UMM	User Mobility Management
UUID	Universally Unique Identifier

VLAN	Virtual Local Area Networks
VM	Virtual Machine
VT	Verification Token
WS	Web Service
WSDL	Web Services Description Language
XACML	eXtensible Access Control Markup Language
XML	Extensible Markup Language
ZK	Zero-Knowledge

Chapter 1 Introduction

As a novel computing and data storage paradigm, cloud technology largely satisfies the emerging requirements of the new IT era. It has been claimed that one of the greatest benefits of cloud computing for corporations and organizations is that it can dramatically reduce the expense and hassle of managing IT systems (Corporation, 2009, L. Krutz and Dean Vines, 2010). A report by Fitzsimmons (2013) has testified to this benefit. Recently, IBM laid off up to 1500 staffs in Australia, an operation that mirrors the decline of the server market among traditional vendors. An increasing number of business customers are shifting their services and applications to cloud computing. Another report (market, 2012) estimated that the global cloud market will reach $270 billion in 2020 with a 30% compound annual growth rate. This acceleration of the cloud trend heralds the time for small-scale IT companies to extending their business. They do not need to invest in their own IT infrastructure, but can delegate and deploy their services to cloud vendors such as Google, Amazon, and Microsoft. On the one hand, this computing paradigm reduces the risk of capital expenditure expansion for IT companies; on the other hand, cloud vendors can leverage more efficient management and coordination on cloud resources to achieve profit maximization.

However, the novel technology alters the traditional service rules for data management, where data management and service are not operated in the customers' local environment but are remotely served by cloud service providers (CSPs). Although this segregation mechanism demonstrates a number of potential advantages compared with conventional paradigms, such as the provision of theoretically infinite data storage capability, high-performance services with low cost, interactive data sharing and pervasive mobile services, users are concerned about losing control of their own data. In terms of the characteristics of cloud computing, resources and services are offered to users in an abstract form. Users may not know where, when, how, why, and by whom or what their data is accessed or modified in CSPs environments (Ko et al., 2011b). Moreover, CSPs are more prone to be attractive targets of adversaries or hackers who may exploit the vulnerability of cloud systems to gain benefit. The cloud paradigm is vulnerable in respect of data security, privacy, and controllability when customers' sensitive data is stored in a third-party CSP. Encrypting data can merely resolve confidentiality and integrity challenges, but the key to mitigate users' concerns and encourage broader adoption of cloud computing is the establishment of a trustworthy relationship between cloud providers and users.

This thesis investigates data security, privacy, and related control issues in the cloud environment. A suite consisting of an innovative ADC framework, assorted security management components such as a CPRBAC model, an AAS scheme, and peripheral full mobility management and adaptive data replica management based on the ADC framework is designed and developed with the support of new ideas, solid frameworks and innovative data structure. Experimental outcome and evaluation demonstrates that this research should help to promote widespread adoption of cloud computing technology in sensitive industries or organizations where users require high transparency, high trust, and high confidence in their outsourced data.

This chapter is organized as follows: Section 1.1 provides a brief introduction to the definition of cloud computing and data storage characteristics in cloud. Section 1.2 outlines the key issues of this research. Section 1.3 introduces the motivation of the research work. Section 1.4 presents the research aims and objectives of this thesis. Section 1.5 summarizes the main contribution of this research. Section 1.6 illustrates the research model and methodology. Finally, we provide an overview of the remainder of this thesis.

1.1 Defining Cloud Computing and Cloud Data Storage

Cloud computing was defined by (Mirzaei, 2009) as "a style of computing where massively scalable IT-enabled capabilities are delivered 'as a service' to external customers using Internet technologies." From an organizational viewpoint, cloud computing is a deployment model for applications that are used by organizations to reduce infrastructure costs by effectively outsourcing to efficient CSPs their infrastructures, development platforms, or even the environment for running their applications.

Generally, cloud computing can be considered as a style of computing and as a model for provisioning and accessing shared resources over networks. The following definition reflects well these two aspects and encompasses well *NIST*'s reasonable definition:

Cloud computing is a style of computing in which dynamically scalable and often virtualized resources are provided as a service over the Internet. Users need not have knowledge of, expertise in, or control over the technology infrastructure in the "Cloud" that supports them (Mell and Grance, 2011). Cloud computing employs a model for enabling available, ubiquitous, convenient and on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, services, processing, memory, network bandwidth, and virtual machines) that can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell and Grance, 2011).

Although many definitions have been offered for cloud computing, they generally identify several common characteristics such as scalability of provisioned resources, pooling of computing resources, on-demand costing models, and network access. The NIST definition (Mell and Grance, 2011) actually identifies five characteristics:

On-demand self-service: Computing capabilities can be provisioned by end users as needed without requiring human interaction with the service's provider;

Network access: Cloud computing provide interfaces for end users via web browser or APIs to access its resources over wired and wireless networks;

Resource pooling: The provider's computing resources are pooled to serve multiple users using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to user demand;

Scalability and elasticity of provisioned resources: Capabilities can be rapidly and elastically provisioned. Cloud computing can adjust the required resources to on-demand;

Measured service: Resource usage can be monitored, controlled, and reported. Transparency can be provided for both the service provider and the consumer of the utilized services. Services can be purchased by users in metered quantity at any time.

Cloud computing has increasingly gained acceptance over the past few years. It promises ultra-large-scalable computing, outsourced data storage, real-time resource sharing, ubiquitous access, and convenient and productive services on a novel pay-as-you-use basis over the Internet with low cost (Foster et al., 2008). Some reports and research (CopperEgg, 2012) indicate that cloud computing has green capabilities: it can help organizations reduce carbon emissions by nearly 86 million metric tons annually through 2020 due to more efficient resource provisioning, and this technology can help businesses reduce energy consumption from $23.3 billion in 2010 to only $16 billion by 2020, as reported by Pike research (CopperEgg, 2012).

Cloud technology is not an isolated concept, in that it has a high overlap with other domains, such as Data Centre Clusters, Grids, Service-Oriented Architecture, and Utility computing (Schubert and Jeffery, 2012, Miller, 2008). However, there are many characteristics associated with cloud computing, such as multi-tenancy, heterogeneity, mobility, economic concerns (Schubert and Jeffery, 2012). Nowadays cloud services are ubiquitous. As the representative cloud service of *Infrastructure as a Service*, Amazon S3 and EC2 (Amazon, 2013c, Amazon, 2013a) deliver their enormous storage and computing resources for users to deploy their applications and businesses in Amazon's infrastructure with a pay-as-you-use model. In the research field, (Cisco, 2013) has reported that McMaster University and Cisco have collaborated to further bioinformatics research on the institutional research cloud infrastructure. With the help of cloud technology, research sharing and collaboration with other universities, institutes, and colleges as well as industry can be conducted and promoted. Closer to our daily life, cloud data storage services such as Dropbox (Dropbox, 2013) and Google Cloud Storage (Google, 2013b) store and manage users' data on the cloud storage infrastructure. There are several benefits when we store data in cloud:

1) **Cloud data can be reused and shared** – cloud servers store users' data and distribute them across different data centers. Open data (e.g., medical research data or air pollution data) may be utilized to analyze results by multiple organizations or institutes (Piscopo, 2012). For collaborations, the analysis results can be shared. Normal users can share their data with authorized people in cloud without costs of data regeneration and dissemination. For instance, Google Drive (Google, 2013c) provides a collaborative platform for users to share files and to collaboratively edit data of documents, spreadsheets, presentations, and so on.

2) **Cloud data can be ubiquitously accessed** – nowadays, mobile phones, tablets, and computers are capable of accessing cloud services anywhere that internet connection is available.

For instance, Dropbox (Dropbox, 2013) provides multiple access approaches (through desktop software, mobile platform, and web portal access).

3) **Unlimited storage** – due to the scalability feature of the cloud, massive storage capability can be achieved by extending cloud data storage servers. Theoretically, users can consume unlimited storage services with the pay-as-you-use model.

4) **Reliable availability** – cloud storage providers normally manage multiple data replicas for users. Hence, when unpredictable incidents occur in users' data, the replicas can guarantee the availability of accessing data services.

Despite these benefits, new challenges and issues may also arise when users store their data in outsourced cloud. In the next section, the key issues and challenges of this thesis are discussed.

1.2 Key Issues of This Research

The economy of cloud computing has received widespread acceptance. CSPs make use of the expertise in organizing and provisioning computational or storage resources to establish large datacenters at low cost (Chen et al., 2010). They sell data storage services to customers with the pay-as-you-use model. This on-demand storage model can achieve better resource utilization and lower costs for cloud users and service providers. To consume this storage pattern, however, users' data must be disseminated to cloud sites. Unlike local storage, this outsourced storage pattern inevitably increases users' concern about losing control of their data.

As is well known, cloud computing security and privacy issues are a broad topic. As a matter of fact, cloud technologies include a number of established business models such as software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS). These models may encounter various security threats related to their service features and infrastructure. For instance, IaaS can be threatened by the risk in which an adversary penetrates a customer's virtual machine (VM) allocated to the same physical server as the adversary via the vulnerability that allows escaping the hypervisor or side-channels between VMs to violate customers' confidentiality (Ristenpart et al., 2009). Since the objectives and context of this thesis focus on cloud data storage, the related networking, computing, and virtualization in cloud computing are beyond the scope of this thesis.

To mitigate users' concern and promote broader adoption of cloud data storage services, this thesis investigates the following issues:

- How to convince users to adopt cloud-based data storage services

To convince users to use their services, cloud data storage service providers must provide sufficient confidence and transparency to their subscribed users. Hence, what features and mechanisms provided by CSPs could enhance confidence and transparency?

- How to establish a trustworthiness relationship between CSPs and users

Trustworthiness is the key to encourage broader adoption of cloud computing. How can this relationship between CSPs and users be established confidentially and accountably?

- How to protect data in the heterogeneous cloud environment with possibly diverse protection standards

Unlike the traditional organization-specific clusters, cloud environments are likely to be constructed from heterogeneous machines with different processing and storage capability (Reiss et al., 2012). In a realistic cloud scenario with a complex and dynamic hierarchical service chain, data handling may be delegated from one CSP to another for business reasons (Sundareswaran et al., 2011). Often, these CSPs do not employ the same protection schemes and standards (Foster et al., 2008). Hence, how can we protect data in such dynamic contexts?

- How to protect data once adversaries penetrate the data storage layer

In cloud, when adversaries penetrate system defenses and effectively become part of the trusted system, they may have sufficient access privilege to compromise or jeopardize users' data. Encrypting data only prevents adversaries from disclosing sensitive information. It cannot effectively hinder the occurrence of data violation and inform the data owner when the danger presents.

- How to achieve a high-efficiency, low-cost data protection scheme in the cloud environment

Achieving data security and privacy protection in the cloud inevitably increases the burden and workload of cloud services. Balancing and controlling the efficiency and cost of data protection schemes is revealed as the challenge.

Based on the analysis of these issues, this research proposes a novel active data-centric framework to protect data security and privacy, beginning with the data itself. The strategy is to let data be active and smart. Any actions or behaviors affecting data should be compulsorily triggered and recorded as attestation of data violation. The detailed description of this framework is provided in Chapter 4.

1.3 Research Motivation

Cloud computing has demonstrated to be a cost effective and preferred business model for managing IT infrastructure and services. However, in the sensitive fields, the adoption and hence the benefits are not being realized due to the concern about the protection of users data in the outsourced cloud environment. Addressing data security, privacy, and control related issues normally require the construction of a variety of complex algorithms and protocols. These algorithms are important for the efficient operation of cloud services; however, the development of such algorithms is a very difficult task. It should consider quality of the employed algorithms, but also performance of execution. Traditional security and protection mechanisms used by CSPs such as encryption, authentication protocols, and digital signature are not sufficient to protect data when data is passed onto a cloud. Users no longer have control over the data and rely on the cloud provider to assure them that their data is in a safe hand. As CSPs, these traditional security concepts and techniques cannot answer when the data is accessed, what has been altered, where it is moved, by whom, furthermore, they inevitably bring in complexity and performance bottleneck due to key distribution, sophisticated encryption and decryption.

Existing work also lacks schemes to effectively prevent intrusion attack, data leakage, and deliver controllability and transparency to data owners that are regarded as the essential elements to establish a trust relationship between CSPs and users. Only under this connection, cloud computing data services can be widely adopted in the sensitive areas. In addition, in a cloud environment with a complex and dynamic hierarchical service chain, data handling may be delegated from one CSP to another for business reasons. There CSPs do not always employ the same protection schemes and standards and data may lose its protection on the new cloud hosts. Therefore, we believe that ensuring cloud data security, mitigating users' concerns, and encouraging broader adoption of cloud computing requires alternative methodologies and technologies. Our research provides very realistic, feasible, and complementary solution that can be built at a tiny cost on existing cloud infrastructure to provide additional accountability, auditability, and deliverable controllability to users in outsourced cloud environment. Users' concern on their outsourced data in cloud can be significantly mitigated.

1.4 Research Aims and Objectives

The primary focus of this research is protecting data in the cloud storage system. We aim to establish a comprehensive data protection framework that can be leveraged to mitigate users' concerns and promote broader adoption of cloud-based data storage services. In existing data security and privacy protection solutions, data is commonly considered to be static, a passive entity that is unable to protect itself. As a consequence, data urgently requires the use of other trusted third-party entities to protect security and privacy. We propose an innovative approach that encapsulates passive data into an active data cube. Here, users' data is not strongly reliant on the peripheral security environment and protection mechanisms. The independent and compact structure encapsulating the data allows it to survive even if the hosts are compromised. Any violations of policy and tampering with data would be compulsorily recorded and reported to its users via the mechanisms within the active data. A trustworthy relationship between cloud providers and users is essentially important. If confidence in this trustworthiness is not established, it is difficult to encourage broader adoption of cloud-based services.

Additionally, the identified factors of data security and privacy such as confidentiality, integrity, availability, scalability, intrusion-tolerance, authentication, authorization, auditability, and accountability must be satisfied to warrant users' confidence. Details related to these factors are introduced in Chapter 2.

The objectives of this thesis are identified to address the above research issues, and can be described as follows:

1) We identify the determinants that affect data security, privacy and controllability in cloud.

2) We examine and explore possible schemes to address trustworthiness-related issues. Various technologies are utilized to solve the defined issues.

3) We design an innovative data structure that enables data to self-protect and self-defend independently in a secure manner. The new data can survive even when the host is in danger. As well as the normal data structure, novel data can also deliver service efficiently at low cost.

4) We investigate related access control, auditing, and mobility and replication management based on the proposed novel data structure, to establish a comprehensive data protection theory that addresses the issues discussed.

5) We evaluate the proposed data protection mechanism to verify whether it satisfies the configured features of protecting data in cloud.

1.5 Research Contribution

The study focuses on data protection in cloud only. Most previous studies on related topics have been based on other diverse platforms, such as Grids, P2P systems, Web service systems and so on. Despite the long history of research in data security and privacy, cloud data protection is still relatively new. This novel computing and storage pattern brings in new challenges and issues that have not previously been experienced. In the following, the research contribution of this thesis is illustrated.

1) Issues of outsourced data security, privacy, and control in cloud are comprehensively and systematically investigated. Through analyzing and researching related work on these issues, a novel data protection model adapted to cloud is proposed. This model is trust-oriented and can establish more trustworthy relationship between cloud providers and users.

2) An innovative ADC framework that achieves data security and privacy protection with platform-independent and multi-tenant support is proposed. This new framework changes the conventional data protection pattern that relies highly on third-party protection mechanisms and strong encryption. The security framework tightly binds data. Any access to data will compulsorily trigger the verification procedure and logging collection. This transparency scheme satisfies the demands of accountability and auditability.
3) Ubiquitous monitoring and tracking of data usage based on the ADC framework with a light workload but high efficiency is fulfilled. Data operations are processed within the active data cube rather than executed by a third party interface, thus guaranteeing minimum information disclosure of data.
4) Two vital services operate on the proposed ADC framework: a CPRBAC service and an AAS. The former governs the access control operations on given active data that can meet the requirement of cloud data access. The latter executes active auditing of users' data operations, collaborating with the CPRBAC service to maintain consistency and security of data manipulation.
5) Full mobility management related to the ADCu is comprehensively investigated and discussed. As a result, active data can maintain a bundle relationship with users via a mobile device that can ubiquitously receive alert messages generated from the ADCu when a violation occurs. Moreover, adaptive data replica management is proposed to address three challenges of availability of cloud data: the degree of data replicas, the distribution of data replicas, and the consistency of data replicas.
6) The experimental result generated on a real cloud test-bed demonstrates the efficiency, dependability, and scalability of the proposed data protection framework. Although a reasonable overhead may be incurred with data structure and operation, the portable, active and independent feature of the ADCu can provide more trustworthy data storage status than conventional mechanisms.
7) A case study is conducted on a real application of protecting electronic health records in eHealth cloud, demonstrating the practicability of the research outcomes presented in this thesis.

1.6 Research Model and Methodology

Figure 1.1 frames the research model and methodology based on a comprehensive software engineering approach. This research begins with research questions which determine the scope and objectives of this work. In terms of the research questions, we first clarify the question concepts. Through the study of related work, we identify the requirements and create the new proposal and design to satisfy them. We then implement the prototype according to the design of the proposal. With the use of a few validation schemes and evaluation of results, we refine the proposal and design to achieve better research outcomes.

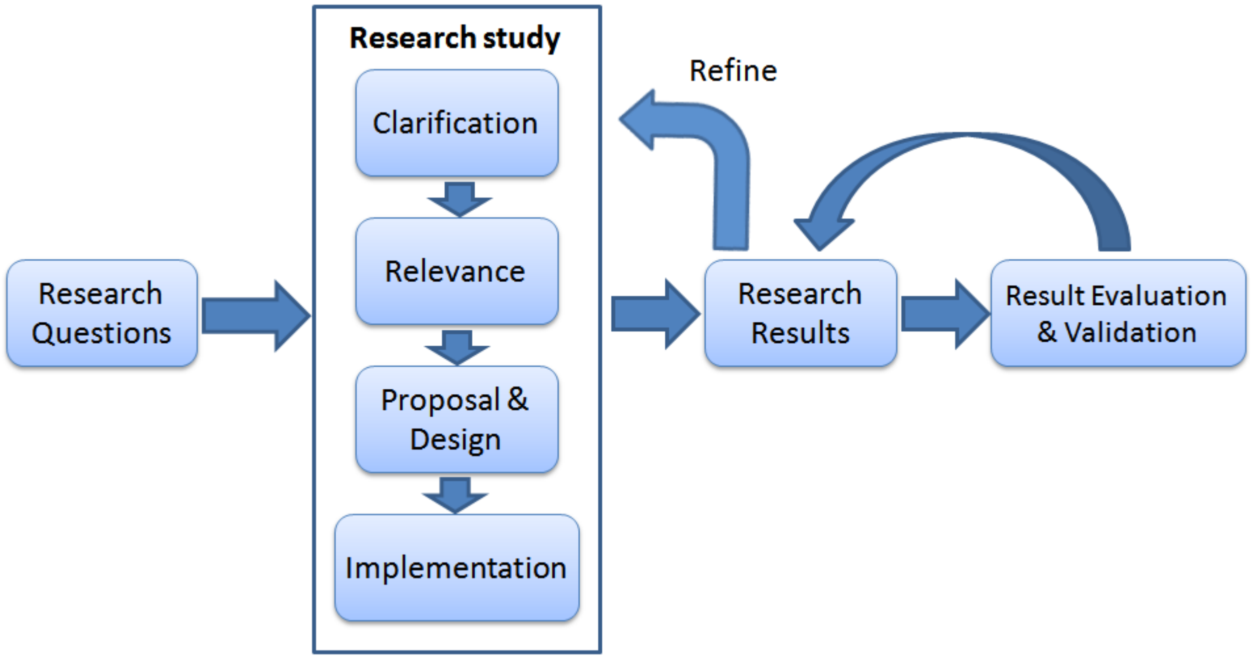


Figure 1.1 Comprehensive software engineering-based research model

1.7 Structure of the Thesis

This research has produced six published international conference papers and two journal papers under review. Figure 1.2 illustrates that the thesis is organized into nine chapters as follows:

This thesis is organized into three main parts. Part I is the introduction of research background and context of cloud data storage and its relevant data protection issues. Part II gives proposed trust-oriented data protection scheme from three layers: data core protection layer, data security and control layer, and data operation and management layer. Part III demonstrates experimental result and analyzes security attributes of proposed frameworks, algorithms, and procedures.

Chapter 1 is an introductory chapter that presents an overview of the whole study, including cloud computing definition and characteristics of cloud data storage, research questions inspired by users' concern on the controllability, security, and privacy of outsourced data in cloud environment, research motivation, research aim and objectives, research contributions, and research model and methodology.

Chapter 2 presents a review of the literature on protecting sensitive data in outsourced cloud environment. We first introduce the characteristics and architecture of cloud computing, and then discuss data security, privacy and control issues in detail in relation to various impact factors. Subsequently, we identify and explore existing solutions for data protection in the cloud context. Further research orientations and comparison of our study to other work are discussed in the end of this chapter.

Chapter 3 introduces a systematic overview of the proposed trust-oriented data protection framework in the cloud context. We first discuss data threat models in the cloud context and then outline the overall trust-oriented data protection infrastructure with three layers. We then discuss data operation patterns based on the ADC framework and compare traditional data structures with the active data structure. Afterwards, we explain the research content via a comprehensive case study.

Chapter 4 mainly investigates the data core protection layer from the perspectives of protecting data confidentiality, integrity and satisfying the intrusion-tolerance feature. This chapter focuses on the ADC framework. There are two vital components proposed in this ADC framework: ADCu and supervisor. They jointly achieve data protection task on the unstructured cloud data with active protection capability.

Chapter 5 focuses on the outer protection layer of the ADC framework – the data security control layer. We propose two significant security services: a CPRBAC service and an AAS. They jointly achieve data security control task through fine-grained accessing control that ensures avoiding abuse of data access privilege, and active auditing on data behavior and manipulation that ensures accountability.

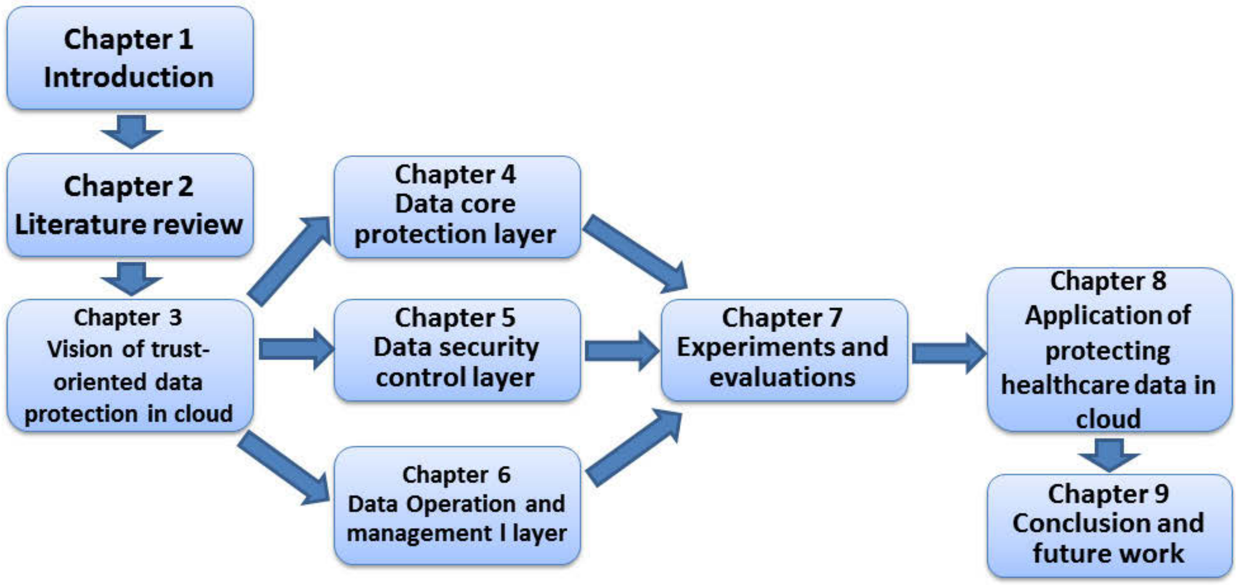

Chapter 1 Introduction
Chapter 2 Literature review
Chapter 3 Vision of trust-oriented data protection in cloud
Chapter 4 Data core protection layer
Chapter 5 Data security control layer
Chapter 6 Data Operation and management l layer
Chapter 7 Experiments and evaluations
Chapter 8 Application of protecting healthcare data in cloud
Chapter 9 Conclusion and future work

Figure 1.2 Thesis structure

Chapter 6 discusses the upper layer of the ADC framework – the data operation and management layer. This chapter addresses two important issues: data full mobility management and data replication management. Although these two schemes are not quite associated with data security and privacy, they are indispensable mechanisms to address data mobility and availability challenges and to form a comprehensively trust-oriented data protection infrastructure in outsourced cloud environment.

Chapter 7 demonstrates experimental results on a private mini cloud platform to evaluate the work described in the entire thesis. We first introduce the cloud platform test environment and simulation environment and then conduct security evaluation, functional testing, conformance testing, and performance testing based on our proposed work. To the end, we compare our schemes with other related works discussed in Chapter 2 from various aspects and features

Chapter 8 introduces a data protection scenario in the eHealth cloud field based on the proposed data protection framework and points out an eHealth development trend in the future.

Chapter 9 summarizes the ideas presented in this thesis, the major contributions of this research, and future research plans and work.

Chapter 2 Literature Review and Related Work

Researchers have proposed various mechanisms or technologies to protect data security and privacy in outsourced cloud, including designated cryptographic-based schemes, dispersal storage schemes, third party trust managers, policy-driven frameworks, and data-binding frameworks. In this chapter, we present a review of the literature on these mechanisms and introduce an ADC framework for protecting sensitive data in outsourced cloud. The novelty of this framework compared to other mechanisms and technologies is that ADC framework enables data to be protected in an active and safe box. It can efficiently prevent intrusion attack and deliver substantial controllability and transparency to data owners.

The structure of this chapter is as follows: Sections 2.1, 2.2, 2.3, and 2.4 introduce data storage characteristics, development history, and architecture of cloud computing. Section 2.5 discusses cloud data security, privacy and control issues in details from various aspects. Section 2.6 focuses on the existing solutions for data protection in cloud and points out the differences to our work. Section 2.7 concludes the chapter.

2.1 Cloud Computing Models and Structures

The three most commonly used service models in cloud are:

Software as a Service (SaaS): This delivers specific-purpose software or applications that consumers access over the Internet (e.g. Google Docs, Google Drive, Dropbox).

Platform as a Service (PaaS): This offers development and deployment platforms for software developers to build or test their web applications (e.g. Google AppEngine).

Infrastructure as a Service (IaaS): This provides underlying IT resources such as processing, storage, and network capability (e.g. Amazon EC2 and S3).

More categories can be suggested in terms of the patterns and resources in cloud. We can cite "X" as a service, and the X may be storage, database, security, information, testing, processing, etc. (Wikipedia, 2013a)

Cloud systems can be classified as public, private, or hybrid. "**Public Clouds**" are deployed via the public Internet and share computing resources. "**Private Clouds**" are services that employ operations with data and enterprise applications running on private, secure off-site data centers. "**Hybrid Clouds**" leverage both private and public cloud services to address various issues with the current cloud technologies.

Cloud computing is associated with high-level abstraction and convenient interfaces, and is thus suitable for both business-to-consumer and consumer-to-business businesses. Cloud computing now provides integrated support for services, platform and infrastructure costing models. Simple cloud services include web hosting services, e-mail, word processing, spreadsheets, and hard-drive back-up. More sophisticated, specialized cloud services may be tailored and offered to corporate and government sectors in the future. Although cloud computing offers services through the Internet, it is not Internet computing, because cloud computing goes beyond just using the Internet and the Web. Cloud computing deploys many other technologies in its environment in providing a whole range of services. Cloud computing is not something that an end-user buys. In fact, end-users are buying results, not assets. They are not concerned with how resources are amassed, configured, or utilized as long as the desired results and applications are delivered cost-effectively.

2.2 Evolution and Development of Cloud Computing

Cloud computing is facilitated with and builds upon many existing technologies, beginning with the availability of broadband networks, inexpensive storage, advanced virtualization techniques through the development of grid computing, utility computing, service-oriented architecture and multi-tenant architecture (Donohue and Ypsilanti, 2009).

Focusing on the perspectives of resource pooling and service provided on demand, a possible evolution path from grid to cloud computing is presented in Figure 2.1 (Stanoevska-Slabeva et al., 2009).

In the early 1990s, **grid computing** utilized interconnected computers and geographically distributed resources collectively to achieve higher performance computing and resource sharing. Grid infrastructure was designed to support collaborative computing and resource sharing across organizational and institutional boundaries. Grid computing middleware software (e.g., Globus) provided the tools for participants to bind together into virtual organizations and share their resources collectively.

In the late 1990s, **utility computing** aimed to offer grid resources in a way similar to the metering of utilities such as gas, electricity, and water. It initiated a business model whereby computing power on demand was offered on a pay-as-you-use basis. Utility computing became mainstreamed with its billing model for computing resources. With the appearance of service-oriented architecture (SOA), grid computing has been offered in the form of grid services that can be used flexibly by application developers to deploy their application on a grid infrastructure.

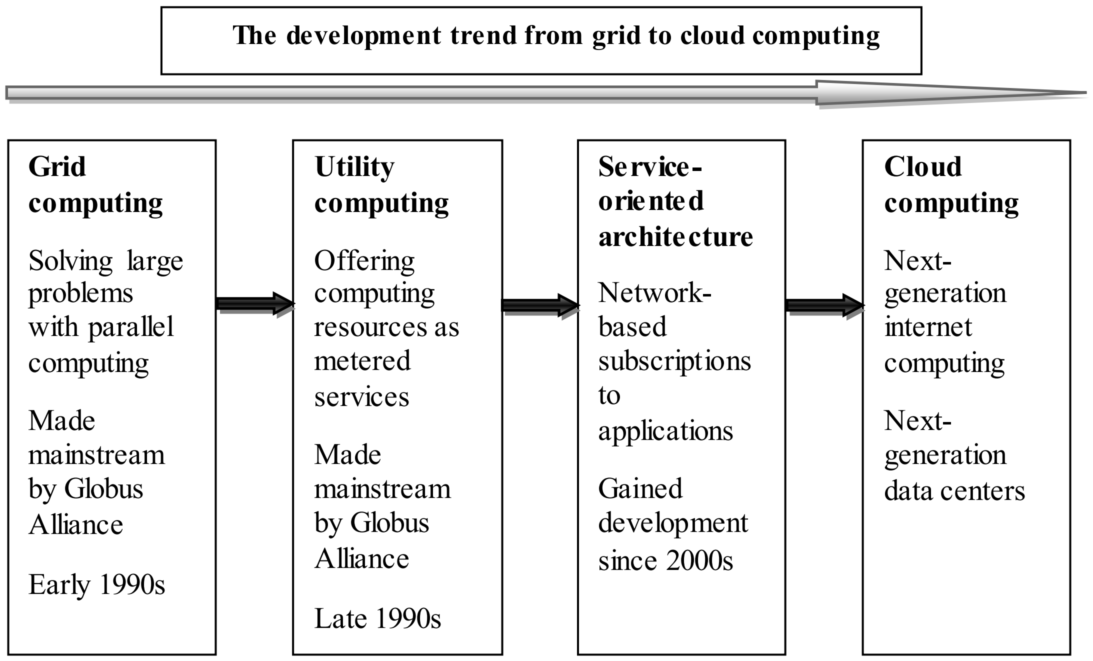

The development trend from grid to cloud computing
Grid computing
Solving large problems with parallel computing
Made mainstream by Globus Alliance
Early 1990s
Utility computing
Offering computing resources as metered services
Made mainstream by Globus Alliance
Late 1990s
Service-oriented architecture
Network-based subscriptions to applications
Gained development since 2000s
Cloud computing
Next-generation internet computing
Next-generation data centers

Figure 2.1 Evolution of cloud computing

Web services/SOA were introduced since the 2000s as software components designed to provide specific services that are accessible using standard Internet technology such as

XML/SOAP (Extensible Markup language/Simple Object Access protocol), HTTP protocol, and WSDL (web service description language).

2.3 Cloud Architecture

Cloud computing can be thought of as a computing model built upon all the above technologies. Although different cloud providers possess unique cloud architectures, a generic cloud computing architecture can be depicted as in Figure 2.2.

The lowest layer of the architecture is the **physical resource layer**. This layer encompasses all components of the physical infrastructure for realizing cloud services. These include servers, storage, database, software, and network.

The **virtual layer** is the core layer that provides cloud resources. This layer utilizes virtualization techniques to implement cloud resources over physical infrastructure. Virtualization is a means of partitioning and configuring physical resources into variable logical resources of the same type in a flexible manner and on demand. For example, a physical machine (computer) can be partitioned into a number of virtual machines which can be run on different operating systems and allocated to different users on demand. All these virtual machines share various partitions of the underlying physical resources including memory, storage, CPU cycles, and network I/O. This layer includes software components for resource pooling and methods for resource deployment. Collectively, the physical layer and the virtual layer provide the resource capabilities of a cloud system.

The next layer is the **utility layer**. This layer provides facilities for monitoring and recording resource usage by each customer and also provides the means for billing according to the pricing policy and model.

Above these layers are a number of management layers that provide and manage the cloud environment for proper operation of services. These include **user management**, **security management**, **task management** and **resource management**. Together with the utility layer, collectively, these form a management middleware layer that deals with a secure environment housing users, scheduling tasks, allocating resources, and ensuring quality of service.

At the top of the architecture, the **SOA layer** provides facilities for housing and offering cloud services, including service registration, discovery, and deployment. The web-service API can be used to interact with the SOA layer and allows users to invoke cloud services and resources.

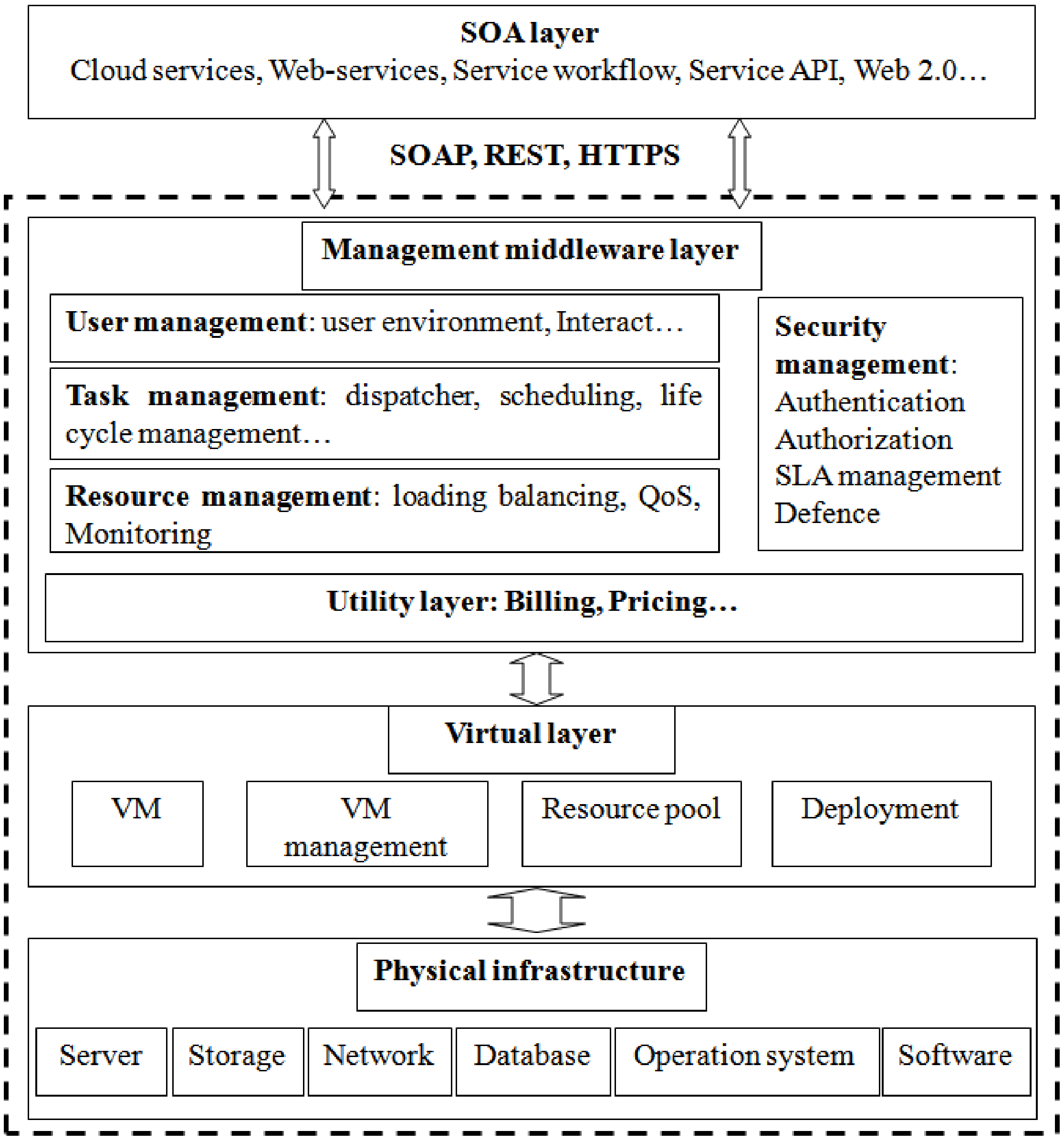

SOA layer
Cloud services, Web-services, Service workflow, Service API, Web 2.0…
SOAP, REST, HTTPS
Management middleware layer
User management: user environment, Interact…
Task management: dispatcher, scheduling, life cycle management…
Resource management: loading balancing, QoS, Monitoring
Security management: Authentication Authorization SLA management Defence
Utility layer: Billing, Pricing…
Virtual layer
VM
VM management
Resource pool
Deployment
Physical infrastructure
Server
Storage
Network
Database
Operation system
Software

Figure 2.2 Generic cloud system architecture

2.4 Characteristics of Cloud Data Storage

Cloud computing provides highly scalable, distributed and flexible services over the Internet on as-needed basis (Mather et al., 2009). This novel and exciting paradigm supports some notable commercial products such as the cloud data storage service providers (e.g. Amazon S3 (Amazon, 2013c), Dropbox (Dropbox, 2013), Google storage (Google, 2013b)). They offer online data storage and accessible capability of data over the Internet on their designated cloud infrastructures. This outsourced mechanism mitigates the burden of local data storage and maintenance by the owners (Cong et al., 2010), and also enables authorized users to share their data flexibly.

2.4.1 Structured and Unstructured Data

Cloud data can be classified as *structured* or *unstructured data* in terms of management type (Webopedia, 2013). The term *structured data* refers to data with an identifiable structure. The most common instance of the *structured data* is the database system, where data is stored based on predefined features and properties and is also searchable by data type with access interfaces. Conversely, *unstructured data* normally has no identifiable structure that refers to any data type. Media data, documents, and complex designated data formats like the electric health record (EHR) are considered *unstructured data*.

To protect these various data types in the outsourced cloud environment, *structured data* management typically interfaces with data by using secure connection interfaces such as ODBC and JDBC, and an enterprise database system may contain more advanced security management protocols and tools to manage its data such as auditing, access control, etc. *Unstructured data* strongly relies on third party services and encryption schemes for auditing and protection. Once third party services are compromised, *unstructured data* would be vulnerable to violation and tampering. For this thesis, we concentrate on protection mechanisms for *unstructured data*. We can deploy *unstructured data* services on the regular cloud platforms that set up MapReduce (Dean and Ghemawat, 2008) and related software such as the open source Hadoop (Shvachko et al., 2010), which are designed to process the parallelization of large-scale data management and workloads (Abadi, 2009).

2.4.2 Multi-tenancy

Multi-tenancy is regarded as one of the essential features of cloud technology. It refers to the principle whereby a single instance of a software application or service serves multiple tenants (Blaisdell, 2012). The customers share the same underlying infrastructure of applications or services, but the flexible configurations enable each customer to customize the individualized appearance or workflow of services. Meanwhile, each tenant may have different security requirements with the subscribed CSPs (Blaisdell, 2012).

The benefits of multi-tenancy architecture for cloud data management can be summarized as follows according to (Blaisdell, 2012):

- The tenants share the same underlying infrastructure. Hence, software development and maintenance costs are lower.
- Each tenant's data is highly isolated and invisible to other tenants. High security can be achieved.
- With multi-tenancy architecture, data of each tenant can be aggregated for consolidated management and auditing.
- With multi-tenancy architecture, CSPs need to carry out updates only once to fulfill all updates of their tenants.

2.4.3 Data Sharing

The cloud data sharing scheme is different from the traditional way of sharing data with other users through disseminating the content from end to end. Cloud storage providers uniformly govern the data center storage of all tenants' data and information. The sharing scheme can be achieved simply by a procedure that assigns authorized cloud users the specific permission to access the shared data via web service interfaces through a distributed access control scheme (Song et al., 2012). The network overheads are also significantly reduced due to the absence of dissemination. This sharing scheme is more flexible because users can selectively share one or more data objects with other users via a fine-grained access control service. Furthermore, this architecture expedites the development of instant collaboration services such as Google drive (Wikipedia, 2013b). Users can share files with anyone of their choosing and can edit files together, the process resembling teamwork from any number of devices with internet connection.

2.4.4 Data Segregation

From users' perspective, data owners no longer physically possess their cloud data. This segregation mechanism eliminates users' ability to control their data in cloud. The resources and services are offered to users in an abstract form. Users may not know where, when, how, and why, or by whom and what their data is accessed or modified in CSPs environments (Ko et al., 2011b). Although outsourced storage provides cost efficiencies in utilities, real estate, and trained personnel and offers larger data storage resources than local storage schemes (L. Krutz and Dean Vines, 2010), the potential disadvantage is that CSPs are more prone to be attractive targets to adversaries or hackers who may exploit the vulnerability of cloud systems to gain benefit. From the perspective of CSPs, once such situations occur, CSPs must take responsibility for the incidents even if the cloud misbehavior is not deliberate. In the local storage paradigm, in contrast, users are expected to shoulder the responsibility of protecting their data from violation or attack.

2.4.5 Other Features and Terms

Cloud data is stored in an outsourced environment.

1) Data is hidden – the physical location and access address are not publicly known. To reach specific data, the index information of the data must be fetched.

2) Data is monitored – data status and data operation are under surveillance corresponding to the subscribed SLAs.

3) Data is auditable – data operation logs and reports are compulsorily recorded in cloud for consistent audit attention.

4) Data is accessible – data can be accessed ubiquitously if the Internet connection is available via mobile platform, desktop platform, web platform, etc.

Privacy of data refers to the right of an entity to determine the degree to which it will reveal or share with data other entities (Shirey, 2003).

Sensitive data refers to data the owners of which do not want to make public (Othmane, 2010).

Data dissemination refers to a process of transferring data from one party to a destination party.

2.5 Cloud Data Security, Privacy, and Control Issues

Essentially, problems of data security, privacy, and control are caused by the outsourced storage environment in cloud. In this section, we introduce the KPMG data security life cycle, provide a detailed analysis of the reasons for the above issues, and illustrate desirable properties of cloud data protection.

2.5.1 Data Security Lifecycle

Data lifecycle in cloud may comply with the KPMG data security life cycle (Mather et al., 2009) as shown in Figure 2.3:

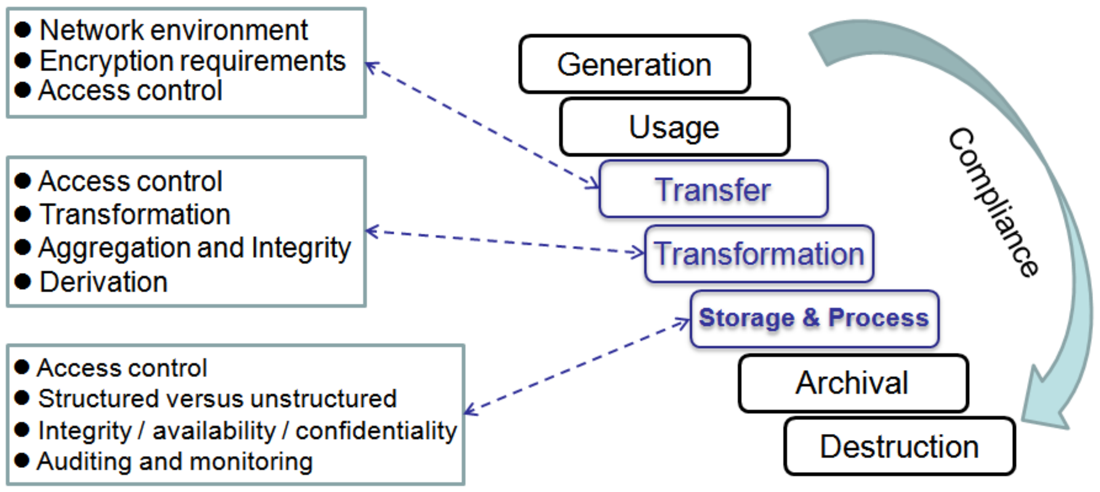


Figure 2.3 KPMG data security life cycle

This life cycle can include: 1) generation of data from users or system configuration; 2) usage of data (access, update, and delete operations, etc.); 3) transfer of data from its owner to one of the cloud hosts; 4) transformation of users' data from static data structure to active structure; 5) storage and processing of data under auditing and monitoring; 6) data archival management to prevent uncontrolled loss or error; 7) destruction of data if it is no longer used. This thesis mainly focuses on the phases of KPMG data security life cycle from *transfer*, and *transformation* to *storage* and *processing*.

2.5.2 Reasons for Data Security, Privacy and Control Issues

The cloud's segregation mechanism has a number of potential advantages compared with conventional paradigms, such as the provision of a theoretically infinite data storage capability, high-performance services with low cost, interactive data sharing, and pervasive mobile services. However, this paradigm entails crucial vulnerabilities of data security, privacy, and control aspects when users store critical data in a CSP. These issues have been cited as the most significant barriers to the widespread adoption of cloud computing, especially in sensitive domains such as finance and healthcare (Squicciarini et al., 2010, Grobauer et al., 2011). From the perspective of users, they may be concerned that their confidential data or privacy could leak in the outsourced cloud. They may not know where, when, how, and why, or by whom or what their data is accessed or processed in CSPs (Ko et al., 2011a). From the perspective of CSPs, they must take responsibility once users' data is violated against the SLAs established when users subscribe to CSPs. After all, with the use of cloud-based services, the risk profile and security responsibility are transferred to CSPs.

Moreover, CSPs are more likely to be attractive targets for adversaries or hackers, who may exploit the vulnerability of cloud systems to gain benefit. The results can be disastrous, including malicious data tampering, data breach, and data confidentiality disclosure. As well, in the real world, due to the inevitability of bugs, crashes, operator errors, hardware failure, or server misconfiguration, data may be leaked, tampered, unavailable, or returned to users inconsistently in outsourced clouds (Popa et al., 2011). Much harder to detected and prevent are internal attacks and violations committed by employees of CSPs.

These concerns are not just assumptions. Recall the Amazon S3's downtime due to server-to-server communication problems (Amazon, 2013b); the Google Docs data security violation which allowed inadvertent sharing with unauthorized users (Popa et al., 2010); the Gmail deletion incident which deleted some users' emails that could not be restored (Arrington, 2006). The fact that these incidents occurred among the best-known commercial cloud service providers illustrates that it is extremely difficult to develop a zero-fail or zero-compromise system to guarantee the safety of data: violations and attacks can still occur at any time. Once such situations occur, CSPs are obliged to take responsibility for the incidents even though the cloud misbehavior might not be deliberate. It is not acceptable to users if the CSP merely admits a small amount of damage or even buries the violation incident to pacify its customers.

2.5.3 Desirable Properties and Objectives for Protecting Data in Cloud

We identify the following desirable properties and objectives for protecting cloud data.

- **Confidentiality**

Data confidentiality refers to the prevention of intentional or unintentional unauthorized disclosure of data or information (L. Krutz and Dean Vines, 2010). In cloud, data confidentiality is usually related to encryption and access control. It is extremely difficult for unauthorized users to reveal and access sensitive information from the encrypted data.

- **Integrity**

Data integrity refers to three principles (L. Krutz and Dean Vines, 2010): 1) data cannot be modified by unauthorized entities or processes; 2) authorized entities or processes cannot execute unauthorized modification; 3) data is consistent among both all internal properties and all external situations.

- **Availability**

Data availability implies that cloud data can be accessed reliably and in a timely fashion by appropriate entities. Availability can be assessed by measuring how the cloud system is functioning in accessing or retrieving data. The denial-of-service event was an example of an attack against availability (L. Krutz and Dean Vines, 2010).

- **Scalability**

Data scalability implies that cloud data can be flexibly extended or disseminated in a flexible manner across different cloud servers. Data can be scaled to satisfy resource growth in a capable manner and distributed to other domains to achieve load balance.

- **Intrusion-tolerance**

Data in cloud should be intrusion-tolerant when the third party security tools are compromised. Once the data storage layer is penetrated by adversaries, data should still be able to self-protect and self-defend against the external intrusion attack when it is disclosed to the adversaries.

- **Authentication and Authorization**

Authentication involves verification of service requesters' identification. It recognizes the requester's identity and ensures access to claimed users. Authorization refers to the specific rights and permissions which can be assigned to authenticated entities to access anticipated resources.

- **Auditability**

Cloud data auditability refers to the ability to monitor the ongoing data operational procedure to ensure that the data workflow is consistent with preconfigured procedure without any deviation or errors. A compulsory audit log should be generated along with the ongoing activities. The documentary evidence of data operations can be testimony as to why data is violated or broken. An active auditing scheme can aid in improving the transparency of the outsourced data in cloud. Through inspecting log information, data owners can know what occurred to their data. A strong auditing framework is considered the key to increase users' confidence in a CSP (Juels and Oprea, 2013).

- **Accountability**

Accountability, in our context, can be defined as the enforceable reporting and auditing mechanisms on the cloud data governance. The goal of this process is to prevent the data from disproportionate harm when a violation occurs (Pearson and Charlesworth, 2009). It can be achieved via SLAs procedure, self-regulation, the use of access controls, and policy compliance. Once a violation occurs, CSPs take corresponding agreed-upon responsibility to compensate users.

Overall, *confidentiality* (C), *integrity* (I), *availability* (A), *scalability* (S), *intrusion-tolerance* (It), *authentication*, *authorization*, *auditability*, and *accountability* (4As) are the expected features to address data security, privacy, and control challenges in the outsourced cloud environment. Satisfying confidentiality, integrity, auditability, and accountability can be considered the primary features necessary to establish mutual trust relationships.

2.6 Possible Solutions for Data Protection in Cloud

Researchers have proposed various mechanisms to protect sensitive data in the cloud environment, which can be categorized as cryptography-based schemes, trust computing technologies, and data-policy binding mechanisms. These solutions focus on different angles of protecting data. In this section, we present a review of the literature on these solutions for protecting data.

2.6.1 Protecting Cloud Data Using Cryptography-based Schemes

Confidentiality and integrity of data in the outsourced cloud can be addressed by cryptography-based schemes. The data is encrypted so that it cannot be readable by unauthorized entities. The quality and robustness of the encryption algorithms is normally demonstrated by the amount of effort required to decrypt the data. Cryptography-based solutions may include encryption, verification of integrity, secure file or storage system, obfuscation, and steganography. In the following sections we investigate these schemes in details.

2.6.1.1 Encryption

Kamara and Lauter (2010) proposed a secure cloud storage service on top of the public cloud infrastructure. The data is first indexed and then encrypted in a symmetric cryptography manner using a unique key. The index file is then encrypted by searchable encryption. To enable users to verify the integrity of the data, the data and index are finally encoded. This work mainly addresses the confidentiality and integrity of the outsourced data through a series of encryption schemes. However, searchable encryption involves a performance bottleneck. In addition, other factors related to data security and privacy in cloud, such as scalability, auditability, and accountability have not been addressed in the work, but we believe they are significant for protecting data.

Virvilis et al. (2011a) proposed a cloud provider agnostic protocol for outsourcing both static and dynamic data to third parties. The protocol combined hybrid encryption and message authentication code technologies to preserve data confidentiality and integrity. Since all operations of the model were performed on the client side, there was no requirement for the cloud providers; meanwhile, it did not introduce additional storage or processing overheads on the server side. However, this work revealed some drawbacks in supporting the operations on resource-limited devices and supporting multi-users access. The work also only discussed the confidentiality and integrity of the data.

Yu et al. (2010) proposed a scheme which achieves fine-grained access control, data confidentiality, and scalability by key policy attribute-based encryption and proxy re-encryption and lazy re-encryption techniques. Furthermore, the proposed work also achieved confidentiality of user access privilege and user security key accountability because most of the computation overheads were delegated to the cloud side. However, this work inevitably entailed high computation overheads, and data integrity and auditability were not achieved or discussed by the scheme.

Another two related studies (Wang et al., 2009c, Yun et al., 2009) proposed encryption schemes for the outsourced data, rather than the cloud context, which also lacked the support of scalability and auditability. Fundamentally, they split the data into smaller data units, and then encrypted the units using symmetric keys. Hence, data confidentiality and integrity could be fulfilled.

All the above solutions focused on resolving data confidentiality and integrity through appropriate encryption mechanisms. Data scalability and the 4A factors were rarely discussed except by (Yu et al., 2010), and computation overheads could be introduced due to the key management and encryption procedure.

2.6.1.2 Verification of Integrity

In the outsourced cloud scenario, there is a scenario in which users may be interested in verifying data integrity with high probability and low communication and computation overheads (Virvilis et al., 2011b). This process should verify without actually downloading the outsourced data. Studies that address that challenge can be generally divided into two schemes (Bowers et al., 2009a):

- Proof of retrievability scheme (POR): a challenge-response protocol whereby CSPs can demonstrate to clients that their files in cloud are retrievable without any loss or corruption.
- Provable data possession schemes (PDP): a challenge-response protocol that only verifies the integrity of the files, but gives no guarantee as to retrieving the files.

Juels and Kaliski (2007) introduced a POR scheme that encrypted data and randomly inserted a set of randomly-valued check blocks (sentinels). By asking servers to retrieve the associated sentinel values, files could be proved as retrievable with high probability. This method supported only static data and allowed partial integrity verification. Moreover, data expansion had to be introduced because of the inserted sentinels, and the overall performance was also affected by the encryption of all the blocks.

Similar work by (Bowers et al., 2009b) achieved a higher level of assurance, lower storage requirements and low computational overheads for a similar technology. The extended work of (Bowers et al., 2009a) supported distributed systems.

Shacham and Waters (2008) proposed a method using homomorphic authenticators in the POR scheme. The homomorphic authenticators addressed performance issues with only insertion of integrity values.

Ateniese et al. (2007) proposed a model for a PDP scheme. By sampling random sets of data blocks from the server the user can verify data possession, with the amount of metadata kept constant.

Erway et al. (2009) introduced a dynamic PDP model enabling users to dynamically update or delete stored files. Wang et al. (2009b) proposed an improved version, which supported dynamic data operation and public verifiability.

The main advantages of the above methods are that data integrity can be verified without physically possessing the data. In this way, communication complexity and overheads can be efficiently reduced. However, integrity violation of data can only be detected after data is tampered with. It is extremely difficult to inspect ongoing violation by the above approaches. In some situations, the damage may be reduced if ongoing violation can be detected in advance. Furthermore, extra data is inserted between the data. Performance would be significantly affected when the size of the data is huge. Verifying such enormous amounts of data may incur computational overheads.

2.6.1.3 Data Dispersal Storage

In cloud, the data dispersal storage scheme is a method of using data fragmentation technology that divides the original data into multiple data units for separate storage, and encrypts them by specific cryptographic schemes. Data confidentiality can be significantly protected due to the increased difficulty of decryption. Theoretically, the more data units are fragmented, the more secure the data can be.

Wang et al. (2011) proposed an information dispersal algorithm to encode and disperse data into n uninterpretable pieces in cloud. To reconstruct the data, only m ($m<=n$) pieces were necessary. Through a secure cloud data access scheme based on *column-access-via-proxy* and

B+ tree index, data integrity and availability can be guaranteed when no more *n-m* servers are faulty.

In earlier similar work (Storer et al., 2008) addressed security requirements for long-term archival storage through three key techniques: secret splitting (similar to the work of (Wang et al., 2011)), approximate pointers (used to reconstitute the data in a reasonable time), and the use of secure, distributed RAID techniques. This work was not based on cloud context, but might be feasible to adapt to the cloud environment.

Wang et al. (2009a) proposed an effective and flexible distributed mechanism using the homomorphic tokens with distributed verification of erasure-coded data in cloud. Importantly, the scheme supported efficient dynamic operations on data blocks, and the authors claimed that the method was resilient to Byzantine failure, malicious data modification attack, and even server collusion attack.

The above mechanisms can clearly meet security and confidentiality requirements of data in the outsourced context. However, this kind of scheme inevitably introduces significant storage overheads and may result in poor performance for update-type data operations and large amounts of data due to the dispersal feature.

2.6.1.4 Secure File Storage System

This type of protection scheme is a system-level cryptographic storage technique. It is used to address secure data storage on untrusted servers.

Kallahalla et al. (2003) introduced a file system, Plutus, that provided strong security and integrity guarantees on file contents by a system-level cryptographic scheme. Key distribution and cryptography were introduced while manipulating dynamic access to files. Significantly, the authors created a file-group for each set of files sharing similar features, with each file-group owning a symmetric lockbox-key. Each file in the file-group was encrypted by a file-block key. Those keys were shared by all files in the file-group. When the owner wished to share a file-group, the corresponding lockbox-key had to be disseminated to users. To avoid redundant cryptography and key distribution, the authors employed lazy revocation and key rotation schemes.

Similarly, Goh et al. (2003) proposed SiRiUS, which was designed over the existing systems such as NFS, CIFS, and so on. Confidentiality was obtained by encrypting the stored files. Additionally, each file in SiRiUS contained the file's access control list to provide end-to-end security.

However, these methods lacked scalability due to the key distribution, management, and complexity of encryption. They were not easily extendable to meet the desired properties discussed previously. Besides, these researches were not based on the cloud context.

2.6.1.5 Obfuscation and Steganography

Obfuscation is normally introduced to make data confusing and harder for others to interpret. To a certain extent, this approach is similar to the encryption scheme. In relevant work, (Pearson et al., 2009) proposed a privacy manager for cloud computing using obfuscation technology. It reduces the risk of data being stolen or misused. A similar mechanism used to address software protection issue was introduced by (Collberg and Thomborson, 2002).

Another innovative security technology, steganography (Wikipedia, 2013g) has been widely used to hide the sensitive content in regular files such as images, videos. The advantage of steganography is that the hidden messages in the files do not attract attention and do not significantly affect the original files. Through such concealment of information within computer files, it is extremely difficult for adversaries to discover the hidden content.

In overview, these cryptography-based techniques impose some constraints on addressing data security and privacy requirement in cloud. Firstly, complexity can be introduced in both key distribution and data encryption and decryption, even though efficient algorithms are utilized to reduce the complexity of key management issues when the usage of data is audited in cloud, or when mining some specific portion of data (Ko et al., 2011b). Secondly, this type of scheme is not suitable for mobile cloud platforms (Christensen, 2009) due to computational overheads. It is not efficient in terms of computation and energy consumption to execute such a high-intensity encryption task on a smartphone. Thirdly, adversaries are still able to violate encrypted data after possessing it, through deleting or moving it to unknown places, even though they cannot extract sensitive information from the encrypted data. In such cases, users would never know what happened to their data. Normally, we must rely on third party auditors (TPAs) to uninterruptedly monitor the state of data in order to take precautions against violation. In the framework presented in this thesis, we do not rely on complex encryption work and complicated key distribution management. In order to increase the accountability and auditability of data, our framework offers maximum transparency to users' data through the proposed ADC framework and pervasive informing framework.

2.6.2 Protecting Cloud Data Using Trust Computing Technologies

To mitigate data security issues, trusted computing technologies are introduced in some cloud solutions to build up a trusted cloud. Trusted computing is defined as: "An entity can be trusted if it always behaves in the expected manner for the intended purpose", which is promoted by the Trusted Computing Group (TCG) . It proposed the TPM (Morris, 2011) based on a low-cost secure chip. It uses measurement scheme to extend trusted region from the trust root to other necessary nodes. Initially, the trust root can be configured by Dynamic Root of Trust for Measurement (DRTM) via invoking the particular instruction in a hypervisor as parameter. DRTM then reconfigures processors and chipset to a trusted status. Subsequently, DRTM calculates the hash value of the hypervisor and save it into a trusted Platform Configuration Register (PCR) (the protected region of TPM). The PCR will be locked and the hypervisor will obtain the control. Hence, The PCR can be the evidence of trust repository (Hao and Cai, 2011). This standard-based hardware components also offer stronger security than traditional software-based approaches that can be hacked, impersonated, or even causing security breaches due to the adoption of dual-level identification and authentication on the hardware layer .

Besides that, on software stacks, TPM can easily integrate with well-known security identification standards for federated ID management such as OpenID, Security Assertion Markup Language (SAML), WS (Web service)-Federation and others . Such as, a cloud service can employ OpenID certificate within TPM to provide strong machine authentication to their service. It also supports measuring the health of the client, verifying a relationship of multiple users, and providing the appropriate access. The other software identification standards such as SAML, and WS-Federation can be also deployed in a similar manner for establishing a trustworthy endpoint across cloud nodes.

A number of researchers have proposed their trust models for cloud services or in distributed environment. The works (Abawajy, 2011, Pawar et al., 2012, Hwang et al., 2009) proposed the reputation-based trust management systems in the cloud environment. This type of reputation models have been widely used in e-commerce and P2P networks. The reputation of services or models is typically represented by a comprehensive score reflecting the overall option, or a small amount of scores on several major aspects of performance, QoS, and security (Huang and Nicol, 2013). Through developing and importing some trust-calculation mechanisms such as dishonest rating, strategic feedback rating, unfairness rating, etc., the trust value of a cloud system can be evaluated and exposed to users. Theoretically, the higher trust score achieved, the better trust and security guaranteed. However, the trust value is highly rely on a large number of cloud users to rate their cloud services or service providers against a large set of complex and fine-grained criteria, which is not realistic (Huang and Nicol, 2013). It may be helpful for users when initially selecting the service. But it exposes shortage after users subscribe to the cloud services. It also discloses vulnerability once adversaries penetrate cloud data storage layer and obtain direct access privilege to data. These proposed solutions lack certain protective ability for data per se. To avoid such constraints, data itself must be smart enough to self-defend and self-protect without the assistance of trusted third party.

The work (Haq et al., 2010) proposed a SLA verification based trust model. This model enables users to verify and re-evaluate the trust with their subscribed cloud services through the SLA. The QOS of the cloud services can be informed to their users that are regarded as one of most important advices for dealing with the trust relationship between cloud users and CSPs. However, this trust model mainly focuses on the SLA defined elements of cloud services, such as performance, inform service, and QoS-related monitoring, and it does not address the security, privacy, and control issues on data storage. Another issue is that individual users and small organizations without technical capability may have to highly rely on a professional third party to provide fine-grained QoS monitoring and SLA verification behalf of them. This inevitably increases another layer of trust management between CSPs and users.

Some researchers also proposed the policy-based trust models. In policy-driven frameworks, cloud policies are generally translated from the natural language polices which define the usage constraints of the corresponding data. The data is only accessible when access is authorized (Ko et al., 2011b). When cloud data is processed, the action must adhere to the policy constraints. From the perspective of data protection, policy-driven frameworks play a role of adjudication in deciding the legality of access in the external layer of data storage.

Lin and Squicciarini (2010) introduced a data protection model based on a policy-driven framework that integrates a policy ranking model (a potential cloud user can find a suitable CSP from a user-based ranking of the CSPs), a policy integration model (the integrated policy is automatically generated according to users' and CSPs' requirements), and a policy enforcement model (that enforces the evaluation of policies across multiple parties).

Takabi and Joshi (2012) proposed policy management as a service that enables users to control and manage their distributed resources across multiple CSPs. Users can specify authorization policies to confine the usage of their data resource via a unified control point. Similarly, (Goyal and Mikkilineni, 2009) presented a conceptual architecture for a policy-driven management service in cloud scenarios.

Policy-driven frameworks focus mainly on the provision of access privileges for requests. If adversaries have sufficient hacking skill or can leverage elevation of privilege to bypass the access control service, cloud data is still in danger. Hence, we consider that the policy-driven framework can be regarded as a peripheral access barrier only. Invalid access requests can be simply filtered or hindered by the barrier. Only legal requests that conform to the contract's policy can be disseminated to the cloud data storage layer to retrieve data.

2.6.3 Protecting Cloud Data Using Data-policy Binding Mechanisms

The data-policy binding mechanism is a novel data protection scheme which encapsulates data, policy, and functional entities within an encapsulated object. The integrity of data structure is enhanced. Any access to data will compulsorily activate the bundled security procedures.

In early relevant work, (Holford et al., 2004) developed security-aware applications to protect software objects through the self-defending object approach. This self-defending object is an extension of the object-oriented programming paradigm. It encapsulates resources and functionalities rather than being distributed throughout the applications. The security aspects of applications could thus be made more reliable. This work was early-stage theoretical research, and no practical outcome was shown. Additionally, the target was to protect security-sensitive software.

Another representative proposal by (Mont et al., 2003) introduced sticky policies which are inseparable from the associated data, to enforce confidentiality protection. This technology prevents attackers from accessing data without satisfying the bundled privacy policies. Two approaches were introduced: (1) identifier-based encryption, and (2) a trusted platform module.

A similar data protection scheme (Maniatis et al., 2011) introduced a deployable data protection scheme based on secure data capsules. Sensitive data is cryptographically bundled with the data-use controls policy (DUCs) and provenance. The authors declared that DUCs might cover confidentiality, integrity, non-repudiation, logging, and access control. However, there was no practical outcome demonstrated in that work.

Othmane et al. (2010), Ranchal et al. (2010), and Angin et al. (2010) proposed an active bundle scheme to protect disseminated sensitive data. Each bundle packages data and metadata involving data access and dissemination policies. Through relevance to a VM with each bundle, the data can be protected by enforcement of the privacy policies in the VM.

More recent research focusing on the cloud area was introduced by (Squicciarini et al., 2010), (Sundareswaran et al., 2011), and (Sundareswaran et al., 2012). These researchers designed cloud information accountability (CIA) frameworks to keep data usage transparent and trackable by embedding the enforceable access control policies and logging policies in a JAR (Java archives) file. One of the main features of their work is that data owners are able not only to audit the content but also to enforce strong back-end protection if demanded. A further extension of their work was published by (Squicciarini et al., 2013), emphasizing independent policy enforcement and data replication management based on autonomous self-controlling objects in the distributed cloud environment.

Despite the similar concept, in fact, our work is different from the above in several aspects. Firstly, in our ADCu data files are encapsulated in an active data container with diverse granules. The size of ADCus can vary to satisfy the requirements of different data types. Secondly, we move our authorization and authentication work from inside the ADCu to external third party access control servers due to the following considerations: 1) the access control policy is designed for the cloud context to achieve highly distributed access control scenarios. Executing fine-grained authentication within the ADCu could involve more overheads; comparatively, our data structure has lighter-weight features; 2) security and privacy policies are highly dynamic in the cloud environment. An obsolete policy in an ADCu might cause inconsistency of authentication. We can store fresh policy in the external access control services. Moreover, we develop two-level protection barriers to achieve a tamper-proof scheme, namely the shell and core of the ADCu. We also employ a ZK proof scheme to verify requests for identification without revealing any vital information which might be leveraged by hackers or attackers to disclose data. Last but not least, our ADC framework can cooperate with an additional data security control layer and the data operation and management layer to establish a more secure and trustworthy relationship between CSPs and users with high efficiency.

2.6.4 Protecting Cloud Data Using Active Data-centric Framework

Protecting data in the outsourced cloud requires more than encryption (Juels and Oprea, 2013), which merely addresses data confidentiality, anti-tampering, and anti-disclosure. We hold the opinion that the key to mitigating users' concerns and to promoting broader adoption of cloud computing is the establishment of a trustworthy relationship between CSPs and users. For users to trust CSPs, data first should be confidentially protected so that no one can disclose sensitive information from it except authorized users. Secondly, any actions or behaviors executed on the data should be compulsorily recorded as evidence of data violation. Once a breach occurs against the SLAs agreed between CSPs and users, the record can be utilized as proof that the CSP has violated the agreed service level. Compensation may then be offered to users. Last but not least, data can survive independently in the heterogeneous cloud environment with possibly a diverse protection framework.

We believe that protecting cloud data is not simply adopting a number of security technologies. CSPs providing data storage services must satisfy *confidentiality*, *integrity*, *availability*, *scalability*, *intrusion-tolerance*, *authentication*, *authorization*, *auditability*, and *accountability* to warrant the confidence of cloud users. In this thesis, we propose a novel ADC framework that embraces activeness of data in protecting itself. Compared with conventional data protection models, our encapsulated active data does not rely strongly on the peripheral security environment and protection mechanisms. The independent and compact structure encapsulating the data allows it to survive even if the host is compromised. A trustworthy relationship between CSPs and users is of prime importance. If the promise of this trustworthiness cannot be fulfilled, it is difficult to argue for broader adoption of cloud-based services. This thesis presents a comprehensive data protection model combining active and passive mechanisms in three layers: the data core protection layer, the data security control layer, and the data operation and management layer, as introduced in Chapter 3.

2.7 Summary

In this chapter, recent studies in the literature on data protection in the cloud environment or related scenarios were reviewed. From the cloud service model, evolution, and architecture, we then moved to the characteristics of cloud data storage. By investigating data security, privacy, and control issues in the cloud context, we analyzed the causes of these issues and illustrated desirable properties and objectives for protecting data in cloud. Finally, we introduced some related work and pointed out the differences and features of our framework.

Chapter 3 The Vision of Trust-oriented Data Protection in Cloud

Active data protection refers to the actions and mechanisms designed to prevent an intrusion attack. Such a scheme can compulsorily inform users when any violation of agreed SLAs occurs on users' data. Conversely, passive data security protection covers the actions and mechanisms taken to monitor and audit data manipulation and status based on third party security services. A robust trust-oriented data protection infrastructure should consider and employ both active and passive approaches because they emphasize different angles. In this chapter, we introduce the overview of our proposed trust-oriented data protection framework in cloud from three key layers in which each layer deals with various properties of protecting cloud data security, privacy, and control.

The research in this thesis is motivated by the concern of cloud users regarding data security, privacy, and controllability in the outsourced cloud environment. In accordance with the key issues discussed in Section 1.2, and the desirable properties and objectives for protecting cloud data in cloud described in Section 2.5.3, we propose a comprehensive data protection infrastructure adapted to the cloud environment. In this chapter, Section 3.1 defines threat models in terms of cloud data context and features. Section 3.2 discusses the features of our proposed data protection framework to ensure cloud data security, privacy, and control issues. Section 3.3 provides an overview of trust-oriented data protection infrastructure in this thesis from three layers. Section 3.4 introduces the data operation workflow in the ADC framework. Section 3.5 compares protection schemes between traditional data structures and active data structures. Section 3.6 is a case study based on the proposed frameworks and mechanisms.

3.1 Defining Threat Models for Cloud Data

"You cannot build secure systems until you understand your threats," claimed Michael Howard (Howard, 2013). We believe that we cannot build a secure system until we evaluate threats to the system, with the goal of reducing the overall risk. Verifying the threat modeling may require significant time, but it can yield tremendous benefits when we design security frameworks and robust systems.

A threat is a possible event that, if it occurs, can cause damage to the confidentiality, integrity, and availability of data. Threats may be accidental, such as accidental deletion of a file or a transaction error, or they may be malicious, such as intentional attacks specific to data. In terms of cloud data characteristics, we propose a threat tree in the cloud data context as shown in Figure 3.1.

Data isolation threat: Users' sensitive data may fall into the wrong hands after its security mechanisms are compromised. This can happen deliberately through an internal violation or through observing the dissemination of data by adversaries, or it can occur accidentally due to false permission setting and enforcement. This is one of the primary concerns of organizations when adopting cloud computing. Normally, data isolation implies that data loses the security and protection capability from its jointed host. Adversaries can further leverage various hacking skills or technical to violate sensitive data without any limitation and regulation.

Data access threat: Adversaries may violate the key components in cloud services such as access control systems or other security entities to fulfill the control of data. Once these access control security mechanisms are compromised, accessing those data will be out of control. This violation would breach the consistency of data access and the SLAs of cloud.

Data exposure threat: This refers to whether data is understandable after adversaries obtain it. This threat can be enlivened if the data is not encrypted, if the data index is leaked, and if the encryption and decryption keys are known by adversaries.

Coarse-grained protection threat: In cloud, a coarse-grained access control scheme may allow adversaries to penetrate the security systems more easily. This might be caused by a coarse security policy structure or lack of multi-domain support.

Elevation of privilege threat: An unprivileged adversary may gain privileged access or an attacker may penetrate the system's defenses and effectively become part of the trusted system. The attacker may have sufficient access to compromise or destroy the entire system. When cloud security systems contain an unexpected privilege account or expose the system entry point, this threat may be present.

Based on the above threats, we propose a combination of active and passive data protection framework. Users' data needs to be encapsulated into an ADCu by an ADC framework that protects the data within from external penetration and tampering. The integral and cohesive data structure packages encrypted data and enables it to be distributed across heterogeneous cloud hosts with protection.

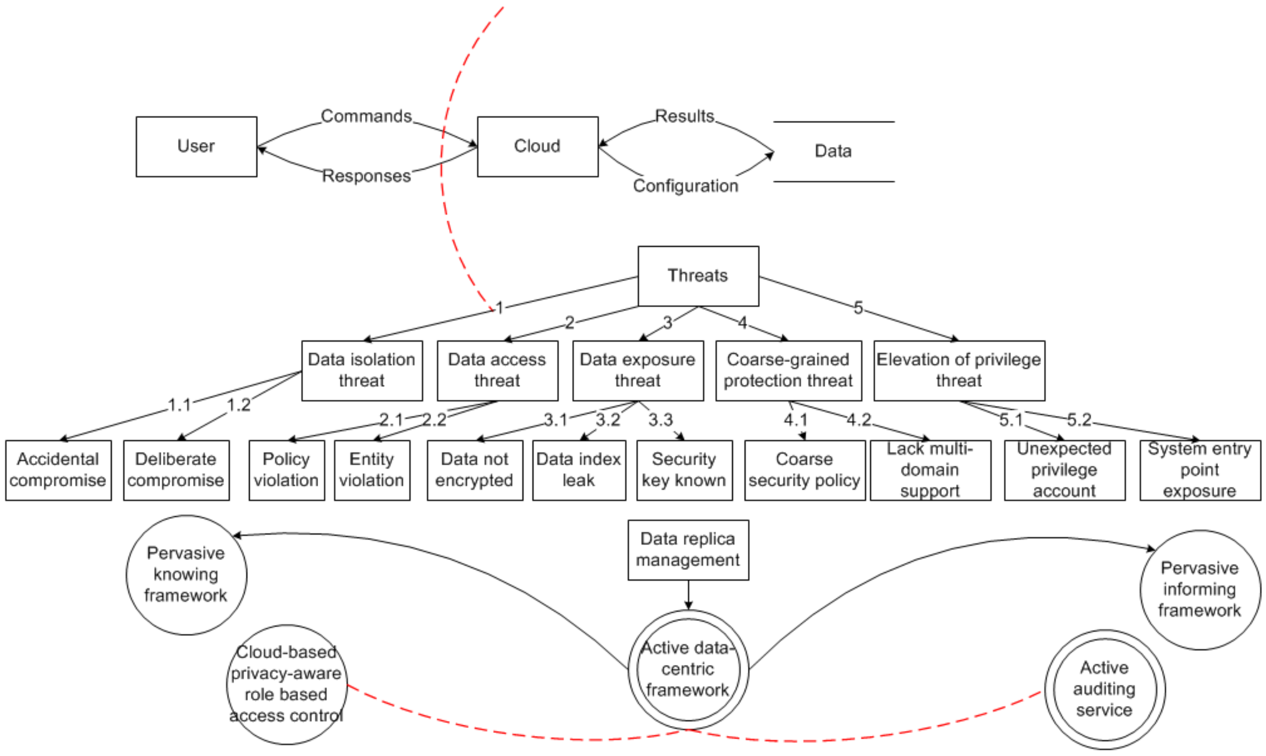


Figure 3.1 Threat models for cloud data

As the core of passive data protection, a designated cloud-based access control service is employed to address authentication and authorization issues in the multi-domain environment. The fine-grained policy structure in the CPRBAC service satisfies the sophisticated security and privacy protection required in cloud. To ensure that both internal and external processes comply with the configured policies, procedures, and regulations in the SLAs subscribed by users, we propose an AAS when users' requests are executed in cloud. Related audit records would be safely and actively stored as the evidence of data manipulation in outsourced cloud. On the one hand, CSPs can use the records to avoid false accusations by users; on the other hand, data owners can browse the audit records generated from their data to discover the activities of their data. As well, we explore full mobility management of users' ADCus through a pervasive knowing framework and a pervasive informing framework. Data owners can ubiquitously fetch relevant physical address of data in cloud and receive alert messages related to data violation or compromise from their bundled ADCus. We also discuss management of data replicas in cloud, which is regarded as an important strategy to enable data recovery and guarantee data availability.

3.2 Ensuring Cloud Data Security, Privacy and Control

This research work consists of three major aspects on cloud data protection: Security, Privacy, and Control. Several concepts are proposed and the research outcome makes a number of innovative models and advances to the knowledge base of the discipline.

Active data concept and techniques for data protection: The active concept has been encountered in places such as "Active Networks", "Active Radio Frequency Identification (RFID)", or even "Active Virus Hiding"; our active concept here allows us to view data from a new angle that enables us to take a different approach and techniques in monitoring, controlling that have not been thought of or attempted before. Clearly this presents an advance in the knowledge base of the information management and security discipline. It delivers active control capability for static data to self-defend jeopardy and inform users when danger presents.

Data and user mobility model: As data may move within and among various Cloud environments, users and CSPs (who may also move outside the environments) need to know the location of their data. Our novel model provides techniques for binding users and their data. Users can then locate their data anywhere anytime and the data informs the users its location whenever it is moved. From this perspective, users obtain substantial control of their outsourced data in the cloud.

Framework for active data protection: Protection of data in cloud environments is more than just dealing with security issues such as authentication, encryption, authorization and integrity; it is also about what happens to the data and its accountability. Based on our proposed active data-centric model, the project develops a novel framework for data protection that consists of dynamic cloud privacy-aware role-based access control, active data encapsulation, and active data auditing. The framework establishes an atomic unit for ensure full protection of data and remove user security and privacy concern.

3.3 Trust-oriented Data Protection Infrastructure

In our framework, data can move across distributed environments with protection by the ADC framework. This is a secure data container that manipulates data access and verification within the data without the involvement of a third party service.

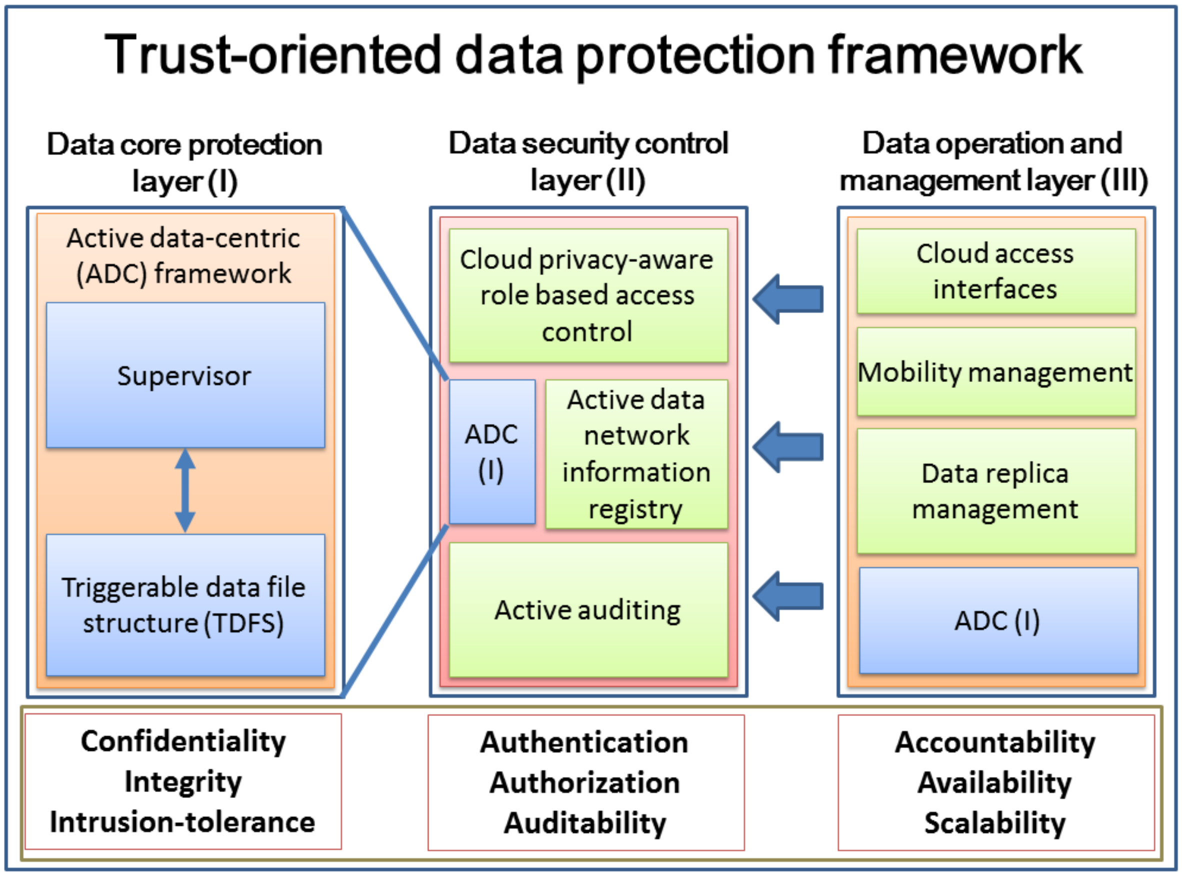

Trust-oriented data protection framework
Data core protection layer (I)
Data security control layer (II)
Data operation and management layer (III)
Active data-centric (ADC) framework
Supervisor
Triggerable data file structure (TDFS)
Cloud privacy-aware role based access control
ADC (I)
Active data network information registry
Active auditing
Cloud access interfaces
Mobility management
Data replica management
ADC (I)
Confidentiality
Integrity
Intrusion-tolerance
Authentication
Authorization
Auditability
Accountability
Availability
Scalability

Figure 3.2 Trust-oriented data protection solution in cloud

This structure presents a higher level security and trust than the conventional data protection models which rely on the peripheral security environment and third party protection mechanisms. Our core goal is to enable data to self-defend and self-protect when intrusion or danger presents. Data misbehavior and violation can be actively detected by the data itself, reducing the risk of use by adversaries. To enable the ADC framework to work in cloud scenarios, we define three layers as shown in Figure 3.2: the data core protection layer, data security control layer, and data operation and management layer. The data core protection layer has an active security approach, whereas the data security control layer and data operation and management layer operate passive security scopes.

3.3.1 Data Core Protection Layer

The data core protection layer is designed to exercise active surveillance, smart analysis, self-protection, and self-defense capability directly on outsourced data. The data encapsulated in a novel data structure called a *triggerable data file structure (TDFS)* manifests confidentiality (through data block encryption), integrity (via a tamper-proof scheme), and intrusion tolerance (via the shell and core structure of the ADCu). This type of data security mechanism distinguishes from traditional sandbox technology commonly employed in computer security (Wikipedia). Sandboxing technology is normally used to test unverified programs that may have a virus or other malignant code, without allowing others to violate the host device (Geier). It is implemented by a tightly controlled set of resources that run in an independent environment without network access or heavily restricted system circumstance. Technically, the proposed ADCu is defined as a deployable data protection unit that can secure sensitive data in the distributed cloud environment. ADCu can achieve isolation of tenants' data, active protection and defense, encapsulation, self-verification, compulsory logging, communication, and tamper-proof. Comparing these two technologies, sandbox is a security mechanism that can apply to network security, virtual machines, native hosts, and program security in a small area without network access or heavily restricted system circumstance. However, our proposed ADCu is designed to satisfy the cloud context that can secure data in the cloud-wide environment. It is able to provide accountability and auditability of data usage via disseminating the log information to subscribed data owners. It also compulsorily records and reports data violation to the data owner via the notification mechanisms. Additionally, it is manageable by the supervisor that can monitor OS level data manipulation which cannot be detected by the ADCu itself. The *supervisor* is an active service instance that is activated when the corresponding tenant subscribes to cloud storage services. Cloud data services can deploy several *supervisor* service instances to deal with a large number of requests in diverse VMs. Cloud service also can terminate idle *supervisor* services when requests are scarce. The *supervisor* executes multi-directional communication between active ADCus, *CPRBAC* service, *active data network information (ADNI)* registry database, and *AAS* deployed in the *data security control layer* and *data operation and management layer*. All the active data of one tenant maintains a connection with the corresponding *supervisors*. Additionally, the *supervisor* is designed to monitor operational system (OS) level data manipulations that cannot be detected by the ADCu itself. Through external environment monitoring, OS level operations such as copy, move, or remove the entire data can be actively detected by the verification monitor in the *supervisor*. Subsequently, the *supervisor* would activate the ADCu to execute a runtime environment analysis and inspect the validity of data operations. A detailed description associated with this layer is presented in Chapter 4.

3.3.2 Data Security Control Layer

In the peripheral environment of the ADC framework, this layer is proposed to execute data security control. The CPRBAC service is proposed to define and describe the security boundary on data operations in cloud. Requests to access the resources in ADCu that are not permitted in the policy repository will be rejected. The fine-grained policy structure allows users to configure and define more specific and secure protection requirements on their data. Authentication and authorization will be offered by the service. As well, the CPRBAC is designed for a distributed cloud scenario. Traditional access control models such as mandatory access control (MAC) and discretionary access control (DAC) (Saidani and Nurcan, 2006) are not designed to satisfy cloud data privacy protection due to their lack of basic components required by privacy regulation. They may, moreover, create enormous access lists when applied to healthcare cloud context. A role-based access control (RBAC) model (Jin et al., 2008) can provide permission administration for a large number of users through authorizing permission to users in terms of their roles. However, the RBAC model is not an effective method in a highly dynamic cloud scenario due to its design for closed domains such as centralized servers. The AAS is introduced to execute and audit users' requests in a secure and consistent manner. Users' requests must be actively audited under a distributed transaction management session. When the communication is not stable, or the verification and logging service cannot be carried on, users' request on the ADCu will be suspended or denied due to the atomic scheme of data operation in our active data-centric framework. Through recording audit data as evidence, the CSPs can report instances of data violation to users and false accusations can be avoided. Users would be more inclined to adopt cloud solutions for their businesses if they can build acceptable SLAs with their subscribed CSPs in a firm and trustworthy relationship. Auditability can be achieved by the AAS. Another important component in this layer is the ADNI registry. It manages relevant active data information, tenants' and supervisors' information. When a new ADCu is created, the related context data will be generated and registered to the ANDI registry. To locate specific data, the *supervisor* can search the index of the data from ADNI registry. If any geographic and network addresses alter, the relevant context information of the data will be asynchronously updated in real time. A detailed description associated with this layer is presented in Chapter 5.

3.3.3 Data Operation and Management Layer

This layer aims to optimize external data operation and mobility management to achieve accountability, availability, and scalability. *Cloud access interfaces* provide data service interfaces to access active data in cloud storage systems. Notably, active data is active when it is triggered or "called" by external functional entities with parameters and is static when it is idle or "uncalled". This design allows sustainable scalability of data. Making data active persistently would cause unnecessary system overheads and may affect scalability requirements. As the important part of achieving accountability, *full mobility management* is proposed between tenants' active data and bundled mobile terminals through an active binding framework and a pervasive informing framework. Any violation message generated from data or any variation of location information can be compulsorily and ubiquitously disseminated to bundled mobile devices via a push mechanism. Moreover, due to the inevitability of bugs, crashes, operator errors, hardware failure, or server misconfiguration, data may be unavailable. *Adaptive data replica management* is adopted to satisfy availability in this layer. A detailed description associated with this layer is presented in Chapter 6.

3.4 Data Operation Workflow in ADC Framework

Within the ADC framework, data operation patterns exhibit salient differences from traditional data operations which rely mainly on third-party interfaces. Figure 3.3 illustrates the top level sequence diagram when users request resources from the targeted ADCus. For an overall, a valid request will experience three important security control tiers: access control, active auditing, and ZK-identity verification. We allocate supervisor instances and AAS to collaborate with the ADCu. When requests committed by the AAS demonstrate that they conform to the configured policy and transaction regulations, they are permitted to execute the necessary operations on the ADCu. The only constraint is that the corresponding *supervisor* of the ADCu can successfully trigger and push operations without giving alarm rise to verification failure. First, the user lodges a data access request to the subscribed CSP through configured web access interfaces. In terms of the selection procedure, cloud access interfaces choose the most optimized active auditor to distribute the request workload. When the request is delegated to an active auditing session, it experiences five significant stages: request analysis, transaction preparation, legislation process, ballot process, and commit process. As the consequence of auditing, a verification token is created as a proof of permission to execute the current operation. After that, the request is delegated to an active *supervisor* instance which takes charge of the targeted ADCu. To further invoke the ADCu to obtain data, the *supervisor* needs to localize the ADCu and generate corresponding request parameters along with the verification token, request context, and its own certification. If the ADCu passes the verification and identification procedure by the ZK proof scheme, the ADCu performs data access or manipulation according to the request context. The result of the request is then returned to the user.

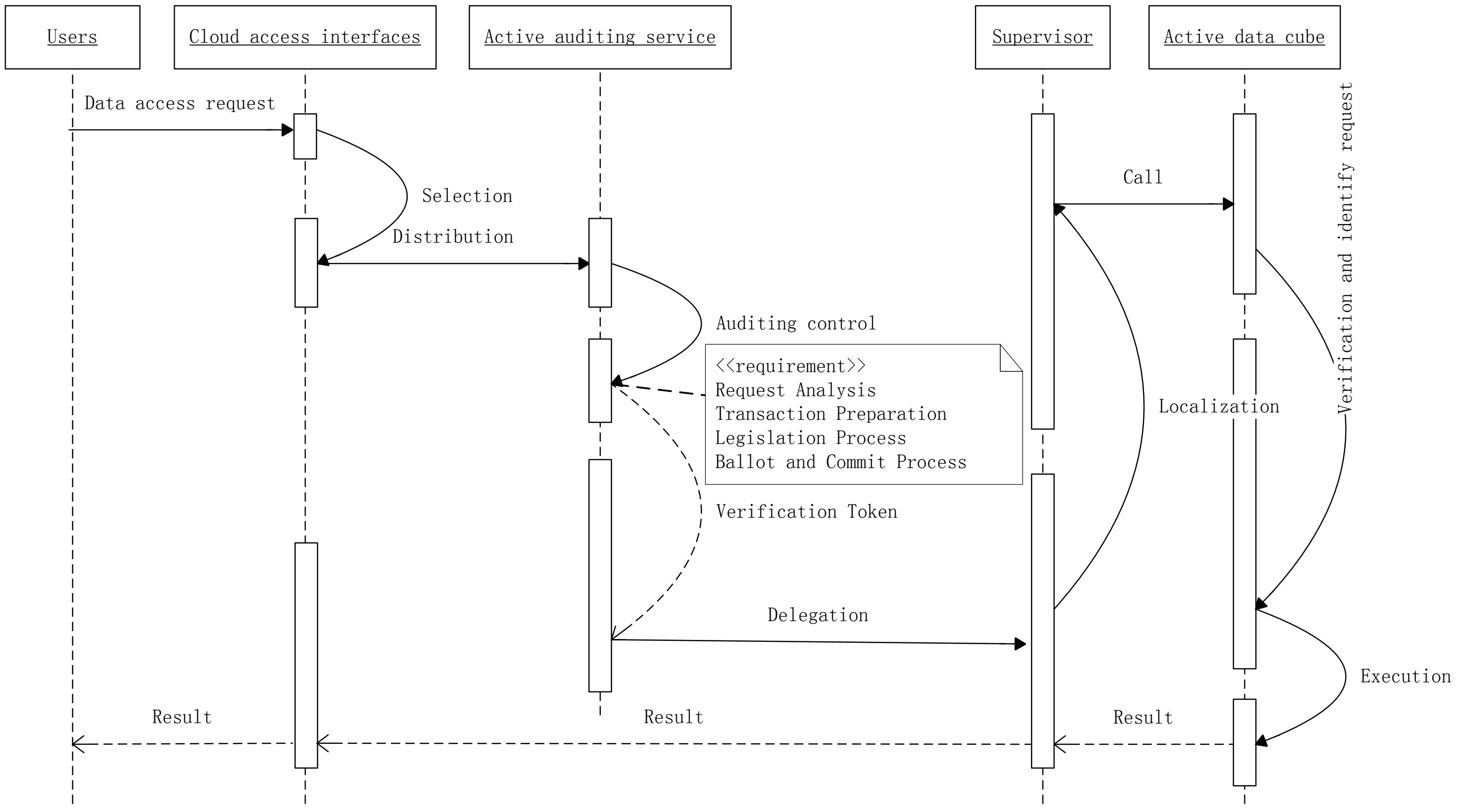


Figure 3.3 Top-level sequence diagram when users request resources from their subscribed cloud services

Figure 3.4 depicts the workflow diagram when users subscribe to cloud storage service. Initially, in order to consume the cloud storage service, users must subscribe to the relevant CSPs. Users key and identity token are generated and stored in the secure authorities upon acceptance of the service terms of the CSP. After subscription, the CSP deploys a virtual user directory and workspace from VM for the user. Subsequently, the Cloud services execute a sequence of initialization tasks for accommodating the data uploaded from the user side, such as invoking

"Ant" lib to compile the required active scripts, generating and configuring empty ADCus with various data types. When the user starts to store his/her data, the corresponding ADCus will be selected to encapsulate the data encrypted on the fly in the sandbox environment in terms of data types. Each ADCu then binds with the allocated supervisor to fulfill the external environment protection to the ADCu. In this procedure, the supervisor requires to deploy and register the ADCu, bind mobile surveillance service in the user's mobile device via GCM framework, and initialize, configure, and adjust replicas network if necessary to achieve load balance and performance optimization. As users, they are allowed to orchestrate their customized cloud access policy to secure the uploaded data in the CSP via the cloud policy management portal. Detailed description of these procedures is provided in Chapters 4, 5, and 6 respectively.

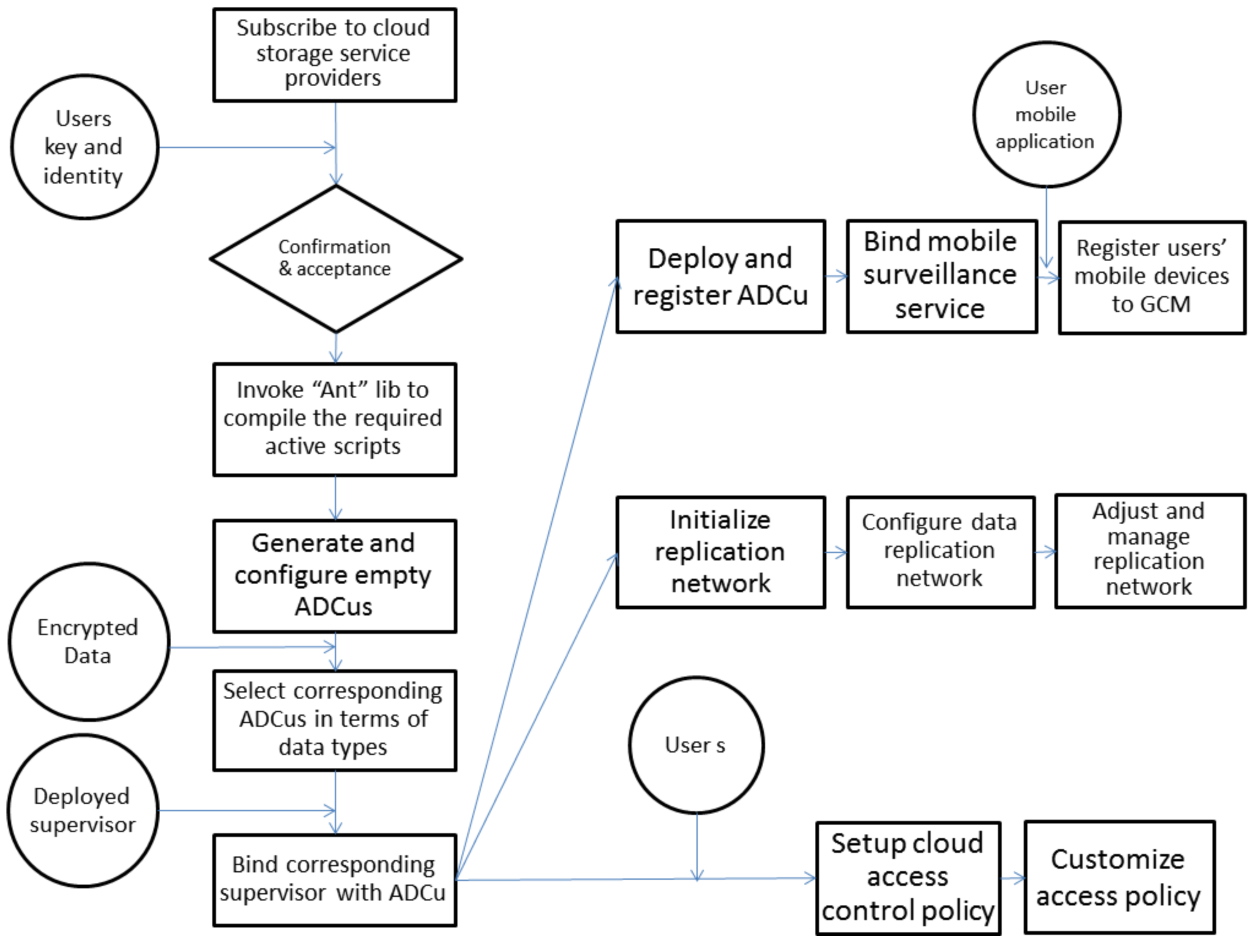

Subscribe to cloud storage service providers
Users key and identity
Confirmation & acceptance
Invoke "Ant" lib to compile the required active scripts
Generate and configure empty ADCus
Encrypted Data
Select corresponding ADCus in terms of data types
Deployed supervisor
Bind corresponding supervisor with ADCu
User mobile application
Deploy and register ADCu
Bind mobile surveillance service
Register users' mobile devices to GCM
Initialize replication network
Configure data replication network
Adjust and manage replication network
User s
Setup cloud access control policy
Customize access policy

Figure 3.4 Workflow diagram when users subscribe to cloud storage service

3.5 Comparison between Traditional Data Structure and Active Data Structure

Our innovative data structure has smarter functionality than traditional data structure which is heavily reliant on third party protection schemes. Active features enable data to collaborate with a collection of security frameworks or models such as the ADC framework, the AAS, the CPRBAC, mobility and replication management. Not only can active protection be achieved in the heterogeneous cloud environment; passive protection can also enhance data security and privacy in the peripheral environment. In contrast, in traditional data protection frameworks these security control services are normally third-party-based and may not be as trustworthy as ADCus.

Table 3.1 provides a comparison between tradition data structure and active data structure with regard to data security and privacy protection. For an overview, protecting data in outsourced cloud requires more than encryption (Juels and Oprea, 2013), which can merely provide data confidentiality, anti-tampering, and anti-disclosure. Our framework is designed to establish a trustworthy relationship between CSPs and users. We arm cloud data with active properties, transaction support, encryption, and verification of integrity. Meanwhile, with our efficient schemes we maintain a light-weight structure and low overheads of data storage and manipulation.

Table 3.1 Comparison between traditional data and active data

Type	Traditional unstructured data	ADCu
Characteristic	It is Static; it is loaded and analyzed by third party services; it should be protected by passive schemes or external services	Dynamic, triggerable, self-protection scheme. self-load and self-analysis
Cloud environment	It is not suitable to directly store in cloud	It is perfectly designed for cloud data security and privacy issues.
Multi-tenant	It does not support multi-tenant feature in cloud; it requires third party services to isolate each tenant's data	Data file per se is an isolated executable entity, and it encapsulates verification, communication, and data loader functions, etc. It could be programmable-secure
Confidentiality	It strongly requires to be encrypted; but hard computation overhead brings in	ADCu does not require hard encryption scheme; data per se embeds control and verification modules
Instantaneity	It is hard to implement protection scheme that monitors and reports the attack or damage on data in real-time.	Once ADCu is called by other entities, it will be triggered immediately and carried out a communication session with the corresponding supervisor. Meanwhile, the operations on data will be logged immediately and compulsorily.
Transaction management	It requires third party transaction manager to control transaction flow	ADCu can embed sub-transaction manager which cooperates with external transaction manager.
Privacy	It can not prevent privacy leak.	Any operation on data would be filtered by ADCu per se. Only valid operation that has already been conformed to the policy can further access the data inside ADCu . Furthermore, the log information also can display any slight modification or even read operation on any specific parts of data.
Security	It highly depends on the external security schemes, such as encryption, anonymity, authorization, and authentication.	It supports both external and internal security schemes. The external schemes can be any current popular security methods, the internal security schemes mainly orient to the encapsulated unit.

3.6 Case Study

In this section we explain the proposed data protection framework from a case study via a metaphor of protecting secret archives in the physical world. As an analogy, we can assimilate cloud data to an archive which contains sensitive information. A cloud datacenter can assimilate to an archive library. Normally, to protect these secret archives, our best strategy is to locate the library far away from adversaries and to install surveillance systems to monitor the activities of all archives. Yet it is relatively easy to penetrate these mechanisms because adversaries can eventually find the place and violate the surveillance systems via disguise or tampering techniques. Once adversaries gain access to the archives, they can reveal sensitive information from within. Traditionally, such archives can be encrypted by designated cryptographic schemes that make it extremely difficult for adversaries to disclose secrets from them, even if the contents fall into the wrong hands. However, on the one hand, strong encryption can incur inevitable computing overheads; on the other hand, even if adversaries just tamper with the archives, it is difficult to detect such misbehavior unless the library carries out an integrity check of all archives.

However, if we allocate all the archives into one independent strongbox with active, alert, communication and trigger functions, we can simply detect any minor change in the secret archives. Moreover, in cloud contexts, creating such an active data box involves almost no cost. In a normal situation, adversaries who are unfamiliar with the access structure of this strongbox will easily trigger an alarm when they try to break into its details. Only an authorized entity (like a supervisor) can obtain archives from the strongbox. The ADCu is designed for such a scenario. We aim to enhancing security preservation even when data itself is under direct attack. The self-protection capability requires a number of controls or intelligences to be installed in the ADCu to enable self-description, self-alarming, and self-defense regardless of the environment. Only an authorized entity which passes the verification in a consistent transaction session and obtains authorization can retrieve information from the data. If we move the strongbox from one library to another library, it actively analyzes the network and location information, and an alarm is raised if the new location violates the policy regulated in the library. As an active unit, the ADCu is also capable of enhancing security and privacy protection through collaboration with the peripheral access control system and audit mechanism. The access control system resembles a firewall that authenticates and authorizes users to access archives. Every access or operation on archives would be compulsorily recorded through the audit scheme. Furthermore, a violation message generated in the secure box can be disseminated to the bundle administrators via mobile devices. The administrator is then aware of every action on and behavior of the protected archives.

3.7 Summary

This chapter has given a systematic overview of the proposed trust-oriented data protection framework in the cloud context. We first defined threat models for cloud data and further introduced the overview of trust-oriented data protection infrastructure in three main layers: the data core protection layer, data security control layer, and data operation and management layer. These diverse layers take charge of satisfying different data protection requirements that contribute toward achieving the goal of trust-oriented data protection. Then we introduced data operation patterns based on an ADC framework and pointed out the difference in data protection between traditional data structures and the active data structure. Finally, we used a case study to explain the proposed mechanisms and frameworks.

Chapter 4 Data Core Protection Layer – ADC Framework

Protecting data with the ADC framework is designed to satisfy the confidentiality, integrity, and intrusion-tolerance requirements for unstructured data in the outsourced cloud environment. The core of this framework makes data active so that it can independently and autonomously analyze risks and violations, rather than relying on third party security schemes. In this chapter, we introduce and discuss the data core protection layer in the proposed trust-oriented data protection framework.

The structure of this chapter is as follows: Section 4.1 introduces the fundamental structures of the ADC framework, mainly involving the features of the ADCu, TDFS, supervisor, and data decomposition scheme. Section 4.2 presents data operation patterns related to the ADC framework, which include regular data operations and tamper-proof schemes. Section 4.3 proposes verification and request identification based on a ZK proof scheme. Section 4.4 describes how to implement the ADC framework in cloud. Section 4.5 focuses on security analysis and evaluation in diverse attack cases targeting the ADC framework. Finally, we summarize this chapter.

4.1 Fundamental Structures of ADC Framework

The ADC framework is a combination framework that involves two entities: supervisor and ADCu. The ADCu protects its inner content from tampering and violation. The supervisor is deployed in the peripheral environment of the ADCu to guard its operations and behaviors at the OS-level.

4.1.1 Features of Active Data Cube

Through encapsulating cloud users' data in ADCus, the following features can be achieved:

- *Isolation of tenants' data:* Each ADCu securely and physically isolates users' data with an active and independent file unit. A user's ADCus can be distributed to a specific domain for considerations of security and isolation.
- *Active protection and defense:* The ADCu is orchestrated by active scripts that enable raw data to execute configured security defense and protection in an active form.
- *Encapsulation:* An ADCu can encapsulate much raw data with similar properties and features to achieve flexibility of data structure and control the granularity of the ADCu.
- *Self-verification:* An ADCu is capable of verifying requesters without involvement of a third party verifier, through employment of the ZK proof scheme.
- *Compulsory logging:* Any data access and query procedures within the ADCu compulsorily trigger the collection of log information, due to embedded checkpoints throughout all active scripts.
- *Communication:* An ADCu encapsulates a necessary network module that assists the data unit to communicate with its supervisor or other security components in cloud services.
- *Tamper-proof:* Double-protection structure (shell and core) in the ADCu provides strong tamper-proof capability.

4.1.2 Overall Structure of ADC Framework

Through the encapsulation of data with active scripts that can be executed in the cross-platform runtime environment, data itself is then capable of violation detection, alarm raising, network environment analysis, communication, regular data operations, and so on. The framework is designed with the following objectives:

1) Embedding data with active functions to prevent it from being tampered with, even if CSPs are dishonest.

2) Compulsorily triggering an automated and authenticated procedure inside data when it is accessed. Through recording every access and transmission of the necessary log information to the data owner, transparency and accountability of data security issue are significantly enhanced.

3) Allowing data to move in cloud environments. In a highly complex and dynamic hierarchical service chain, data handling may be delegated from a direct CSP to another different CSPs in a flexible manner (Sundareswaran et al., 2011), but often these CSPs do not employ the same protection schemes and standards (Foster et al., 2008). In this case, raw data may lose its protection on the new cloud hosts. We propose to enable data to still survive in the new host even though that host may be compromised. Figure 4.1 demonstrates the architecture of the ADC framework.

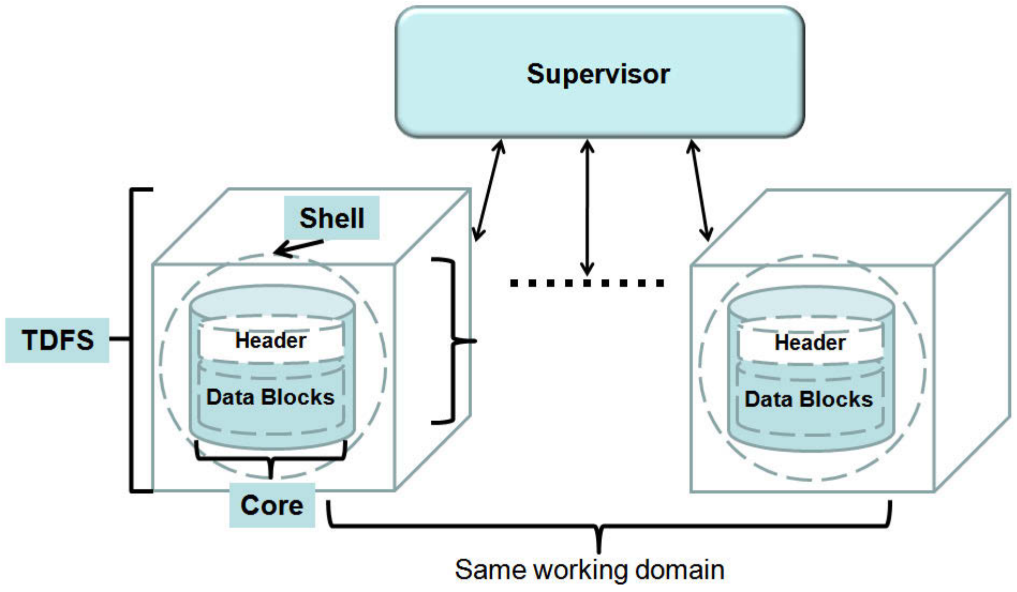


Figure 4.1 Active data-centric framework architecture

There are many situations where attackers may utilize unconventional approaches such as decompiling or reverse engineering to crack the internal defense of ADCus to maliciously destroy the active data. Theoretically, we cannot fully resist attempts by adversaries to acquire data in cloud (Sundareswaran et al., 2012). They may leverage some vulnerabilities of the system, such as analyzing compiled byte codes or memory breakpoints to penetrate the defense of ADCu. However, we can increase the difficulty of data acquisition through encryption technology, enforceable monitoring and bundled logging, or raising an alarm during ongoing violation, rather than preventing the occurrence of attacks. To achieve these goals, our approach appends an additional *shell* layer outside the data *core* layer. This concept is inspired by the concept of armored viruses (Internet-Guide, 2011). Virus programmers normally create a "shell" program to protect or hide the content of viruses from disclosure to escape the defense of antivirus software. Although this might result in destructive effects on the system or data because the virus might succeed in escaping antivirus software via the armored technique, we can leverage the technique to protect ADCu. We define this layer's program as the "shell". The *shell* performs detection of tampering, verification and identification of requests, remote communication, real-time logging, and terminating access when tampering is evident. All components associated with data operations are wrapped into the *core* in the ADCu. Additionally, a *supervisor* is designed to monitor OS-level data manipulation that cannot be detected by the ADCu. Through external environment monitoring, OS-level operations such as copy, move, and remove entire ADCu can be actively detected by the verification monitor inside the supervisor. Subsequently, the supervisor would activate the ADCu to execute a runtime environment analysis and inspect the validity of data operations.

4.1.3 Triggerable Data File Structure

As the core of the ADC framework, the TDFS is a new data structure (referring to the ADCu) that has a major responsibility of protecting the inner data content from violation or attack, as well as providing an efficient data provision service for incoming requests. In this section, we describe the TDFS in detail.

As shown in Figure 4.2, the TDFS consists of a *shell* and a *core*. The *shell* is equipped with active tamper-proofing codes and is executed prior to the *core* when the TDFS is triggered. The TDFS is associated with the Java runtime environment. When it is triggered by a process, the Java.exe in the JVM invokes the *GetMainClassName* function which obtains the *JNIEnv* instance, and then invokes *getManifest*() in *Java.util.jar.JarFileJNIEnv* class to return the object value which contains *Main-class*. The main class runs when the TDFS is called. With that, the main function invokes the *LoadClass* method in *Java.c* to load the main class. Next, the main function invokes the *GetStaticMethodID* method in *JNIEnv* instance to search for the "*public* *static void main (String[] args)*” method and then invokes the *CallStaticVoidMethod* method to process the incoming request.

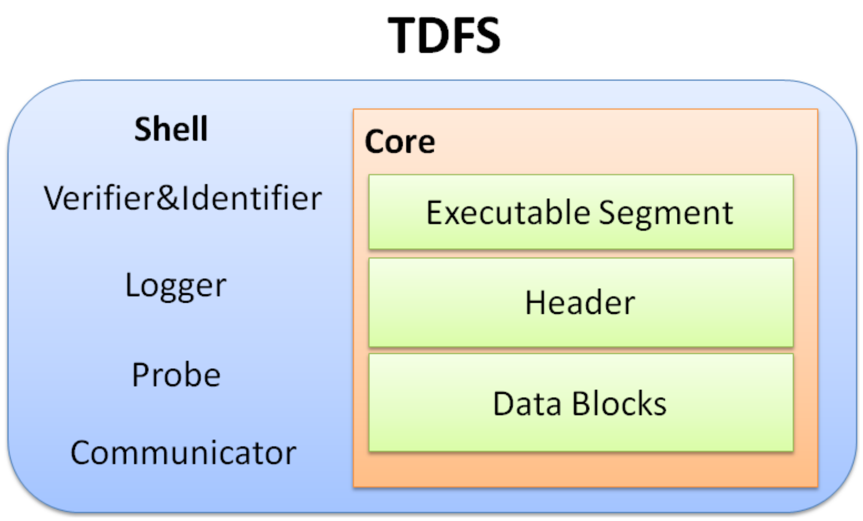


Figure 4.2 Skeleton of TDFS

At the entry point to the *shell*, the scripts invoke *verifier and identifier* to certify the validity of the request in terms of the format of request parameters and contents via request identification and verification process. In our context, a permitted access request conforming to the configured CPRBAC policy is issued a certification which signifies that the access is authorized by the CPRBAC service. The TDFS subsequently leverages the ZK proof scheme to verify and identify the requester which is the *supervisor*.

Another significant component of the *shell* is the *logger* module, utilized to record significant checkpoints during transactions, data operation outcomes, and even errors when the TDFS throws exceptions. The *logger* is required to record significant intermediate information when the TDFS is running in JVM, which efficiently improves data accountability and auditability. Each log record in the TDFS is created in the form of [Subject_ID, TDFS_ID, Entity, Behavior, TimeStamp, Location, Priority], which indicates that a *Subject* (referring to the entity that requests data) identified by *Subject_ID* performed a *Behavior* on the TDFS identified by *TDFS_ID* with *Priority* level at *TimeStamp* in *Location*. During a single transaction, all log records marked with regular *Priority* level are stored temporarily in memory for performance consideration. Once the data operation finishes, the *logger* leverages the *communicator* in the *shell* to upload the log records to TDFS's external supervisor. However, a log record marked with an emergency tag will be immediately triggered by the *probe*, which then notifies the *communicator* to raise an exception. *TimeStamp* uses the Network Time Protocol (NTP, 2013) to take into account the fact that cloud resources may be distributed across different time zones. *Location* of the TDFS is determined by IP address and MAC address of current host. Each TDFS's log information is transparent to its data owner. When the log records are stored in cloud, they are encrypted using the RSA encryption to avoid possible information leakage. Only the data owner has the corresponding key to disclose those records. In addition, sending out the log information of data usage rather than storing it inside the TDFS is activated to maintaining the light-weight feature. Increasing log information could raise the cost of storage, dissemination, or replication of the TDFS.

The *communicator* uses RMI-SSL technique (Konstantinou, 2003) to enhance communication security. Each TDFS has a corresponding *supervisor* deployed in the same domain, which takes charge of monitoring external data operations (such as move, and copy) that cannot be detected by the internal probe in the TDFS, and communicating with TDFS. If the TDFS cannot establish a proper network connection or cannot contact its supervisor, it would switch to the termination state to avoid offline attack.

A *probe* in the *shell* is triggered by three types of activity: program exception, inconsistent checksum in data blocks, and verification failure of ZK proof procedure.

Once the verification and identification procedure succeeds, the *shell* delegates control to the data *core*. The *core* of TDFS is wrapped by an executable segment (ES), a header, and data blocks. The runnable scripts in the ES consist of a number of basic data operations, data loading, and data analysis functions. These functions are tightly coupled with the data. We leverage dynamic delegation approach in the ES to call back the *shell* to trigger the *core* and execute data operations. The *header* refers to a manifest specifying the basic information of supplementary data residing at the beginning of *data blocks*. The following are necessary elements for the TDFS header: security identifier (SID), data descriptor (DDes), and timestamp.

SID – The unique value of variable length for identifying a specific data unit in the global cloud environment. It is linear and consists of *Revision* (refers to the version of SID) +*Domain identifier* (refers to the domain ID of data) + *Data unique identifier* (described by 32-character hexadecimal string).

DDes – The security information of data structure includes *data content description*, *data owner identifier*, and *data type identifier* (e.g. designated EHR type, image type, or script type). The *data checksum* is a fixed-size arbitrary block of data which is used to verify data integrity. Malicious tampering in data blocks can be detected by recomputing the checksum and comparing it with the original. Another important block in the DDes is the *policy index*. To avoid malicious modification of policy from within data, for security reasons we only store the index of policies in data.

Timestamp – The string sequence recording the time when an event occurs.

The *data blocks* are used to store raw data requiring protection. They are encrypted by AES symmetric encryption for performance considerations. We employ the RSA non-symmetric encryption method to encode the key of AES and store it safely in trusted authorities. We choose not to use the RSA encryption for entire data blocks due to its feature that only each 117 bits can be encrypted. For the case of encrypting a whole data file, it may be necessary to chop the data into a large number of pieces to encrypt, which inevitably increases overheads. Notably, ADCu does not decrypt and encrypt data blocks inside the active cube, but executes them on the fly. Decrypting and encrypting data in ADCu would impact on access and security performance since each of ADCu is an independent-active instance. A long processing in an instance would break the scalability and reliability. Hence, we execute them on the fly in the secure sandbox environment. For one side, the sandbox environment can be deployed and distributed in the cloud hosts, for another side, it is secure enough to provide encryption and decryption services in enclosed execution environment.

4.1.4 Supervisor

In our design, the ADCu can only protect and manage the internal content. It cannot be triggered to active status if adversaries attempt to execute OS-level operations such as move or delete the entire data cube via elevation of privilege attack. In addition, the ADCu is designed to only accept triggering requests from its trusted entities for enhanced security concern. To address these two design constraints, a supervisor service instance is developed as shown in Figure 4.3. A1 represents an active data cube supervised by the supervisor instance marked by red color. It is a significant component in the ADC framework to supervise the OS-level violation for ADCus. Each VM may deploy a number of supervisor instances that depend on the quantity of served users. Each user can own more than one supervisor that depends on the quantity of stored data cubes. Supervisor is a semi-trusted entity in the ADC framework normally deployed in the same working domain as the ADCus. It only presents the highly trusted bind relationship with its supervised ADCus via the ZK verification scheme. Hence, it is a 1:* relationship to ADCus.

Four components are configured in the supervisor: *communicator* (employed RMI-SSL protocol to securely communicate between external data security, management services and ADCus), *OS-level monitor* (configured as a background service running in the allocated VM. Through listening to data status changes, the monitor knows what is occurring in the ADCu), *verifier* (used to elementarily verify incoming requests from external services in cloud), and *trigger* (used to “call” the designated ADCu with corresponding parameters and tokens). In the ADC framework we adopt a data access pattern that is different from conventional method that uses third party tools to manipulate data directly. Data operation and query are manipulated by data itself through encapsulating necessary operation interfaces and functions. We consider that the ADCu can demonstrate more trustworthy operational patterns without the involvement of other external entities. Moreover, for performance overhead considerations, the ADCu does not continuously maintain active status; each ADCu possesses a thread running at the backend of the cloud server when it is triggered. We turn the ADCu from active to static status when it finishes the working task.

Since it is not a fully trusted entity, there is still a chance to be violated by adversaries if they bypass the data operation and management layer and the data security control layer. In this case, the ADCus are not functional to the adjacent supervisor. They remain secure due to the encapsulation features and their inner security mechanisms. In another case, if the ADCus are moved from one VM host to another new VM host, ADCus will first attempt to call the new supervisor instance in the new VM host to request register. The supervisor in the new host will communicate with the supervisor in the original host to retrieve corresponding security and register information related to the ADCus. Once the new register is done in the new host, the original supervisor will unregister the supervised ADCus. If the ADCus are moved without any authorization from one host to another untrusted host without trusted supervisor instance, the ADCus will raise an alarm to inform their data owner. We conducted a test of moving one ADCu from one cloud host to another public cloud host with our ADC framework. The experimental outcome proves the feasibility of our framework. The detailed description about this test is discussed in Section 7.2.3.

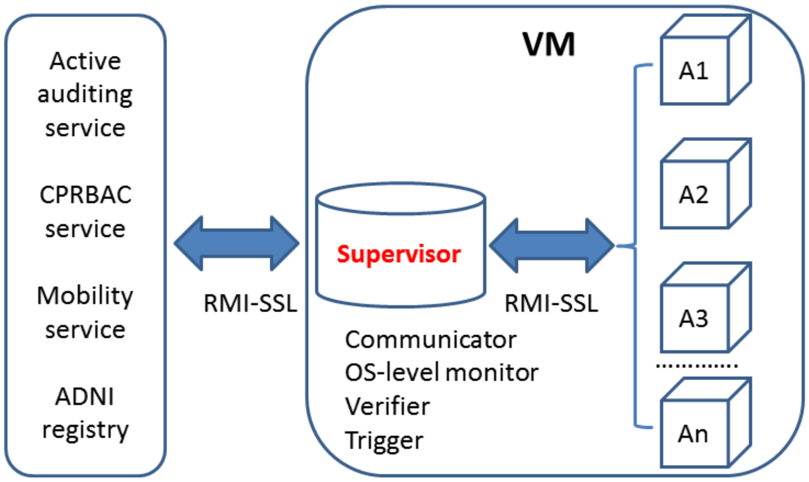


Figure 4.3 Supervisor service instance

4.1.5 ADCu Decomposition Scheme

In a distributed cloud storage scenario, although the ADCu encapsulates tenants' data for integrated management and security administration, massive data files packaged into one active data container can reduce the flexibility and performance of data operations, such as moving entire data, replicating an ADCu for better parallel access, and so on. Conversely, encapsulating each data file within an ADCu may cause high storage overheads and performance bottlenecks, especially with a great quantity of small files, such as script files. Hence, we propose a decomposition scheme to allocate tenants' data into multiple secure boxes in terms of data type and size. Figure 4.4 illustrates the data decomposition diagram for a tenant's data. When a tenant's data is stored in cloud, its position is generally unordered and invisible. We first categorize the tenant's data in terms of data types, such as media data, text data, or script data, and so on. According to the data type, data files are then encapsulated in a fixed size to guarantee flexibility and granularity of the ADCu.

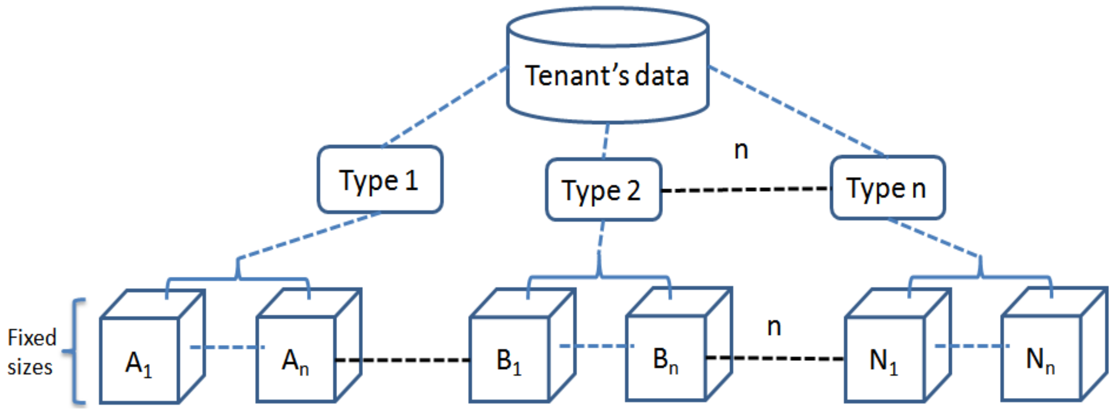


Figure 4.4 Data decomposition diagram of a tenant's data

This tree-type structure enables data with similar properties and features to be encapsulated into one ADCu. Normally, two ADCus are logically adjacent but not physically adjacent. They may be distributed across different cloud virtual hosts for workload balance. When a data cube is experiencing frequent update operations, its space may be available to encapsulate new data. To improve the utilization rate of each data cube, we allocate new data to the data cube by sorting the utilization rate of the data cubes. In terms of the type of data, the size of data cubes could be designated to satisfy different granules of data. For example, video data has greater granularity than script data. Hence, we configure a larger data cube for video type data than for script data.

4.2 Data Operation Patterns in ADC Framework

Data operations in the ADC framework have a different form from regular data operations that rely on third party operation interfaces. In this section, we explain in detail the ADCu update and read operations and the tamper-proof scheme.

4.2.1 Data Update Operation

Figure 4.5 introduces data update operations that occur when we use the ADC framework. The empty ADCu is initially signed and sealed. However, it has already packaged necessary active scripts. When tenants want to update or add new data into the ADCu, the corresponding supervisor first triggers the ADCu to notify that there is a data update or add request on it. This request must contain a set of parameters: identities of request (the supervisor and subject of the tenant in this case), verification token (generated by CPRBAC service during active auditing session, which indicates that the request conforms to policy and has been permitted to execute), new data digest, and data type. The ADCu then inspects the validity of the request. If the request is illegal, the ADCu raises an alarm to notify the bundle tenant. The verification procedure requires a three-pass ZK proof scheme between the supervisor and the ADCu in order to make sure that only the certified supervisor of the ADCu can activate it without failure of verification. After successful validation, new data must be encrypted by a symmetric cryptographic scheme on the fly by the supervisor in order to increase the difficulty of disclosure.

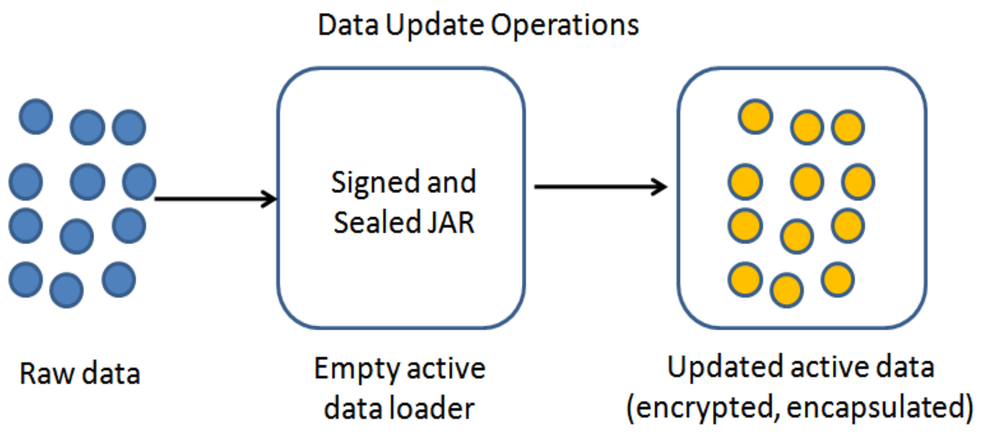


Figure 4.5 Data update operations

4.2.2 Data Query or Read Operation

As well as update operations on an ADCu, when tenants execute a query or read request on the targeted data, a three-pass ZK proof scheme is first conducted between the supervisor and the ADCu. Figure 4.6 illustrates a data query or read operation on the ADCu. The request from an authorized user is processed by the data access interfaces in the ADCu. An invalid request or one forged by adversaries would trigger an alarm. The request would be blocked by the ADCu by terminating the runtime instance of the current data cue. A valid request is delegated to the data core to execute data loading. Subsequently, the ADCu returns the query result to the supervisor, and it is then disseminated to the cloud data access interface.

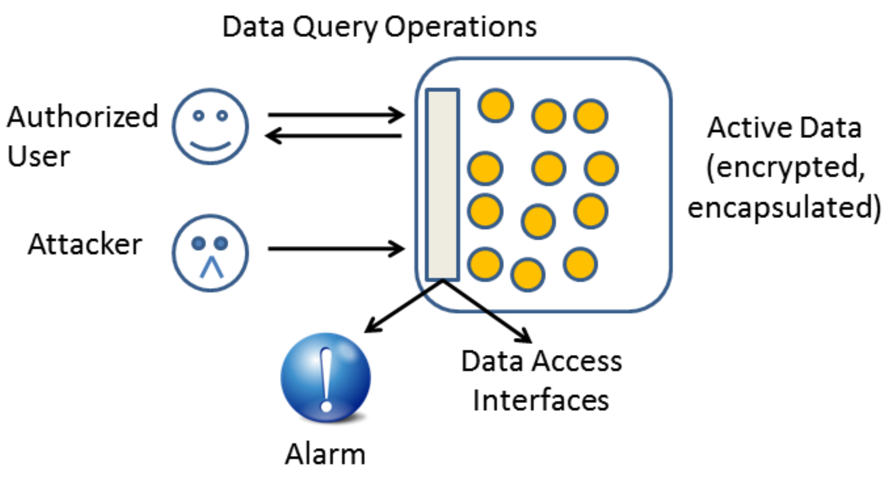


Figure 4.6 Data read or query operations

4.2.3 Data Tamper-Proof Schemes

Since we cannot guarantee that adversaries will never gain possession of stored active data in cloud, to prevent ADCus from tampering attacks after adversaries penetrate cloud security firewalls (such as CPRBAC and auditing service in our context) and obtain direct contact with data, on the one hand, we can rely on the strength of cryptographic primitives to guarantee the integrity and confidentiality of data; on the other hand, the *shell* inside the TDFS actually conceals the real entry point of data operations by preventing direct access to the *core*. The *shell* executes prior to the *core* in the TDFS and the *core* only accepts requests from the *shell*. Thus, when adversaries attempt to extract useful information from the *shell* scripts to access *core*, we deploy the following approaches to detect the malicious tampering:

- *Verification and identification*: the *shell* scripts use a ZK proof scheme to verify the requester's identity through a three-pass protocol. The details of the scheme are introduced in the next section. This procedure ensures that only the corresponding supervisor of the ADCu can successfully delegate a user's request to retrieve a result.
- *Integrity check*: Before execution of verification and identification, the *supervisor* starts a self-check program to check the checksum of targeted data. If the new checksum is

inconsistent with the checksum stored in the *header*, the ADCu terminates the program immediately to block all requests, meanwhile sending an alert message to its *supervisor*.

- *Intermediate result check*: We define a number of checkpoints throughout the whole active scripts in both shell and core of ADCu. These checkpoints must be executed while the active scripts are running in an orderly fashion. If digest value of each checkpoint is not consistent with the configured value, the ADCu also will throw an exception and trigger the *probe* to raise an alarm.

4.3 Verification and Request Identification

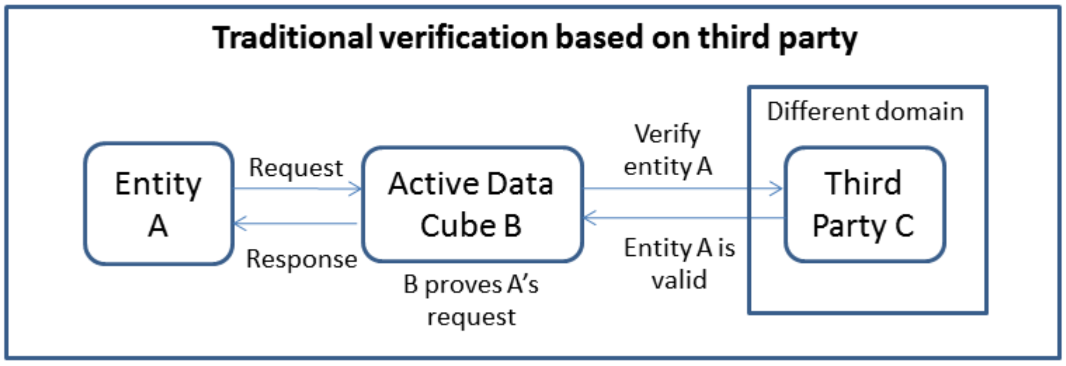
The ADC framework enables data to ascertain the validity of incoming requests. (Squicciarini et al., 2013) introduced a bundle policy enforcement scheme inside data packaged by self-controlling objects, that guarantees data security through the adaptive security policies enforcement. However, we shift the policy enforcement module from the ADCu to external access control systems for several reasons:

1) The access control policy should be designed to operate in the cloud context with highly distributed access control scenarios. Executing a fine-grained authorization inside ADCu may entail high overheads, especially when a large number of requests target one ADCu. Additional authorization would slow down the response from the ADCu.

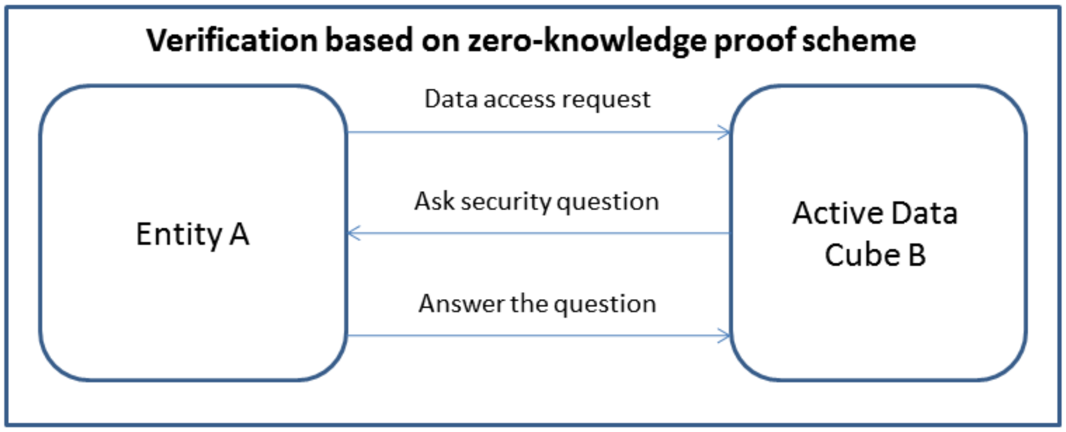
2) The policy is highly dynamic in the distributed cloud environment. An obsolete policy in the ADCu can cause inconsistency of authorization. Conversely, if we release this work to external access control services, the performance of authorization tasks can be significantly improved via optimization and allocation of cloud re-sources. Meanwhile, we can keep policy fresh.

Nevertheless, we must still implement a compact re-quest identification scheme within the ADCu to verify the requesters, after which is authorized by the designated access control service (CPRBAC). We propose a verification and identification scheme which adopts a ZK proof scheme to verify the request's identification without revealing any vital information that may be leveraged by adversaries to disclose sensitive information of data by forging the set of valid request parameters. As the ZK proof scheme is conducted between the ADCu and its supervisor, it can still provide a verification function even if the ADCu moves to an untrusted host without an external network connection because the supervisor is generally deployed in the same working domain as its supervised ADCus.

When a data request arrives at the final commit stage, the active auditor (AA) forwards the request and a permitted access token to the supervisor of the targeted ADCu. The supervisor subsequently triggers the requested ADCu with a valid proof and verification token generated by the CPRBAC service to fulfill data operations. The enforcement engine inside the ADCu requires the proof of authentication of the requester. We assume that an active eavesdropper A has already penetrated the security systems and obtained a straightforward access privilege on ADCu. A may possibly be capable of analyzing the triggering parameters from the supervisor via network eavesdropping, thereby forging a valid request to access the ADCu even though A does not participate in the entire procedure. Hence, distinguishing a valid access from an invalid access on the same ADCu becomes a challenge. Figure 4.7 illustrates the traditional verification scheme based on third party verifiers (shown in Figure A) and the verification scheme based on the ZK proof scheme (shown in Figure B). In general, the third party proof scheme relies heavily on network communication in the real world because these proof services are normally deployed in diverse cloud domains. Hence, the ADCu cannot fulfill the verification process if it cannot contact the third party proof services. Moreover, it is easy for this multi-party communication procedure to reveal vulnerability to adversaries through a possible man-in-middle attack. But the ZK proof scheme (Pieprzyk et al., 2003) occurs between just two entities. It can still provide the proof function even if the ADCu moves to an untrusted host without external network connection. From the perspective of cloud computing architecture, data may be distributed across multiple domains geographically. This scheme is suitable to assist ADCus to distinguish valid and invalid requests. Moreover, it can efficiently reduce the probability of forging proof due to the zero information disclosure during the proof process. For the ZK proof protocols, three typical schemes are often deployed: Fiat-Shamir (Fiat and Shamir, 1987), Guillou-Quisquater (GQ) (Guillou and Quisquater, 1988), and Schnorr (Schnorr, 1990) scheme. Comparing these three proof protocols, Schnorr scheme possesses the advantage of efficiency between two functional entities due to being based on the discrete logarithm modulo a prime integer problem and remains high security (Petković and Jonker, 2007). We adapt Schnorr identification scheme in the ADC framework:



(A)



(B)

Figure 4.7 Traditional verification based on third party (A) and verification based on ZK proof scheme (B)

Parameter initialization in active auditor (AA):

1) The AA chooses two primes p and q such that $p - 1$ is divisible by q, and ensures that the discrete logarithms of modules p and q are incalculable in theory.

2) The AA determines the security level through security parameter t, and meets $2^t < q$.

3) The AA chooses β with multiplication order q, which meets $1 \leq \beta \leq p - 1$ and $\beta^q \equiv 1 \bmod p$.

4) The AA selects a one-way hash function $h(m)$ that calculates the signature for the message m. AA publishes the set of security system parameters (p, q, β) and verification function $h(m)$.

Registration of cloud users:

1) When a request arrives at the AA for registration of the service, the AA verifies the requester's identity and then generates an identifiable string I_A containing the basic information related to the requester.

2) The user chooses a private key α which meets $0 \leq \alpha \leq q - 1$. The AA calculates a corresponding public key $v \equiv \beta^{-\alpha}$ (mod p).

3) The user verifies his or her identity from the AA through conventionally presenting personal information, and then delivers the v to the AA. The AA issues a certification $cert_A = (I_A, v, h(I_A, v))$ that binds I_A and v.

Identification protocol: (Entity A proves its identity to ADCu B)

1) A selects a random number r which meets $1 \leq r \leq q - 1$, and computes $x \equiv \beta^r (\text{mod } p)$. A sends $(cert_A, x)$ to B.

2) B verifies the certification from A through the public key v in $cert_A$ in the AA, and then sends a random number e (which is never used) which meets $1 \leq e \leq 2^t$ to A.

3) A sends the answer $y \equiv \alpha * e + r (\text{mod } q)$ to B.

4) B computes $z \equiv \beta^y * v^e (\text{mod } p)$ which can be decomposed as $z \equiv (\beta^y \text{mod } p * v^e \text{mod } p) \text{mod } p$, B accepts A's proof if z=x, otherwise refuses A's proof.

4.3.1 Analysis of ZK Proof Scheme

The AA module provides system security parameters. The protocol in this scheme takes three-round communications between entity A and ADCu B. Clearly, if A follows the protocol, it will be always correctly identified by B. If not, ADCu B will reject A's request. Now we analyze the security factors of the protocol from the following aspects:

1) *Forgery probability*: We assume that an attacker D would like to guess B's question. Let us assume that D's guess is g. B sends a question e and D must respond $y \equiv \alpha * (e - g) + r (\mathrm{mod}\, q)$. If D can guess a valid response $y \equiv r (\mathrm{mod}\, q)$, the probability of a correct guess of e is 2^{-t}. If $t \geq 40$, the probability of success will be less than $9 * 10^{-13}$. Theoretically, it is extremely difficult to crack this question in a certain period of time.

2) *Minimum information disclosure*: This scheme does not reveal any useful information related to α since x is a random number, y requires α to calculate, and D cannot answer y without α.

3) The *discrete logarithm* issue is extremely difficult to compute in mathematic theory (Pieprzyk et al., 2003).

4) Once verification time exceeds the normal time threshold, the ADCu automatically terminates the verification procedure.

5) Once the private key α has been chosen, it is easy to compute the corresponding public key v. For the inverse process, however, computing α from v requires computation of the discrete logarithm with base β of $v - 1$.

4.4 Implementing ADC Framework

Implementing the active data structure is similar to the work that extends the object-oriented programming paradigm to offer active functions and protect sensitive data as an object (Holford et al., 2004). The Java-based approach Java Archive (JAR) (Oracle, 2011) is one of the best options to achieve the demand we have defined. All active functions such as logging, security verification, network communication, and data I/O operation can be developed by the standard Java library or extension. To encapsulate users' data with active scripts, we employ ANT (Kallambella, 2006) to programmatically create the ADCu together with compiled functional scripts and encrypted sensitive data. Furthermore, CSPs can digitally sign and obfuscate JAR files to increase security and difficulty of disclosing their content. Each generated JAR file is executable across platforms when third party tools trigger it with appropriate parameters. Moreover, JAR supports a compression scheme that reduces its size for light-weight considerations. Another active instance service supervisor is developed by an OS-level service which keeps running when the hosted VM is on. The service is implemented based on the JPathWatch API which is a Java library for monitoring directories for changes on hosts' platform's native OS. Currently, this library supports all-windows, Linux, Mac OS X, and FreeBSD OS. The details related to implementing the ADC framework are illustrated in Chapter 7.

4.5 Security Analysis and Evaluation

In this section, we analyze possible attacks to our ADC framework. As the peripheral environment protector, the data security control layer in the entire protection framework resembles a firewall in the network security system that manages incoming and outgoing network traffic to secure the internal network or computers. The data security control layer blocks and defends against external attacks by adversaries. Through the authorization and authentication scheme based on the CPRBAC service, unauthorized requests are prevented from accessing the active data stored in the cloud data storage layer. However, adversaries may leverage elevation of privilege or illegal channels to gain higher administration privileges that can bypass the data security control layer to directly commit an internal attack or penetrate the data storage layer. In this work, we mainly focus on cases of the attack targeting the ADCu, to test and analyze the security attributions.

The following analysis and evaluation are based on a number of assumptions. We assume that data owners do not release any sensitive information to unauthorized parties, including secret keys used to generate a signature and encrypt data and personal privacy that may be utilized by adversaries to crack the user account. We assume that the supervisor and CPRBAC service are trustworthy and behave correctly. We assume that the adversary has penetrated the data security control layer and has accessed the data storage layer of cloud. He or she will attempt to carry out the following attacks:

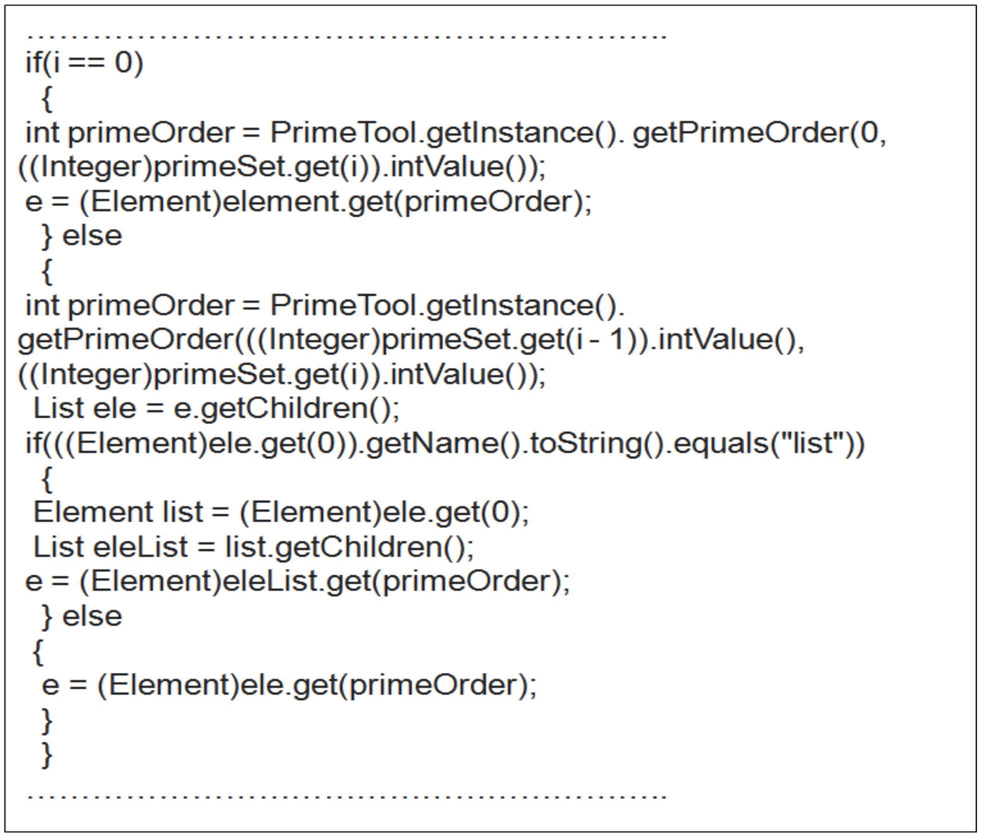
4.5.1 Direct Access and Intrusion Attacks

Once the adversary bypasses the data security control layer and gains direct access to active data, he or she can dynamically trigger the ADCu to analyze scripts. Through executing a number of tests with different entry parameters, the key elements of the scripts may be disclosed to the adversary. For instance, the verification and identification module in the shell of the ADCu would first execute validity analysis of the request. Only the request passing verification is allowed to into the data core for further data access.

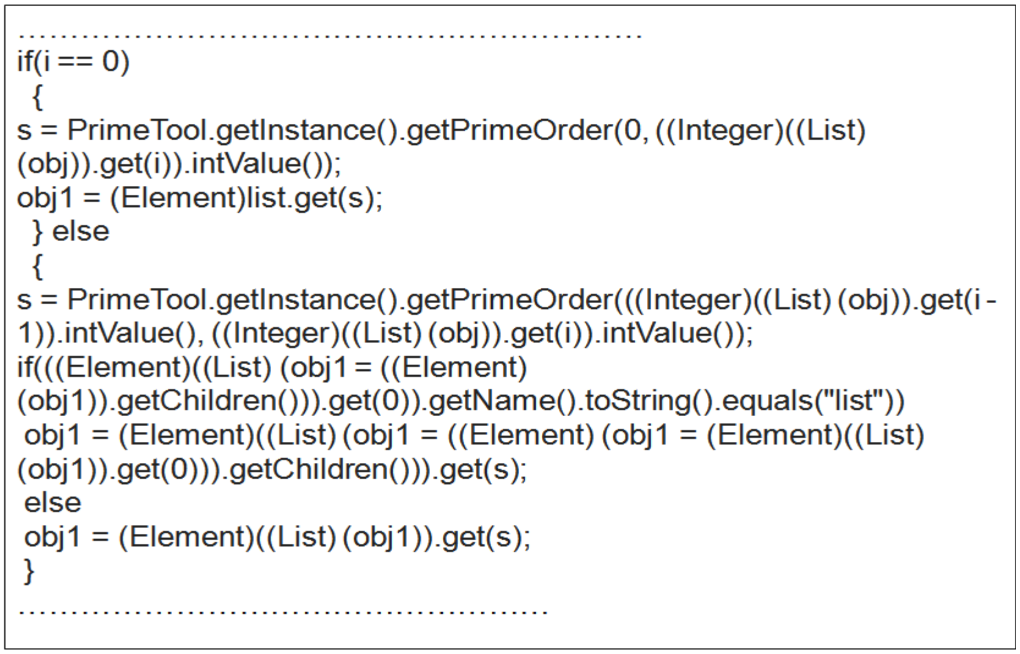
A request without any parameters is immediately regarded as an intrusion. The probe embedded in the ADCu would alert its supervisor about the intrusion event. If the adversary understands all entry parameters and verification procedures, and attempts to forge a set of valid request parameters to steal the packaged data, the ADCu will also raise the alarm to terminate the program when it detects such forged requests. Based on the fact that the ZK proof scheme in the verification and identification module verifies requesters' identification without revealing any vital information that can be leveraged by adversaries to disclosure sensitive data, only an authorized entity like the supervisor is capable of successfully executing the verification procedure. Furthermore, if the ADCu experiences a large number of requests from the same domain or if the verification time exceeds the configured time threshold, the ADCu actively rejects the requests.

4.5.2 Reverse Engineering and Decompilation Attacks

The ADCu is implemented by JAR technology (Oracle, 2011). As we proposed, an ADCu mainly consists of active scripts and encapsulated data. The active scripts are the threshold to access the encapsulated data in ADCu. While, the active scripts function as security guard that is able to verifying the incoming requests, examining checkpoint outcome, and informing data violation, etc. Due to the active scripts are implemented by Java compiled bytecode, the adversary with sufficient IT and hacking skills might be able to disassemble ADCu through reverse engineering and decompilation attacks. Reverse engineering (Wikipedia) is the way of discovering the technological principles of a device, object, or system through analysis of its structure, function, and operation. Once the ADCu is disassembled, the compiled script files and encapsulated data will be exposed to the adversary. Through decompiling the script files, the original source code can be extracted. It is highly dangerous if the adversary injects new codes to violate the active functions within the ADCu after understanding the whole security protocol from the original source codes. To defend against this type of attack, code obfuscation technology (Wikipedia, 2013e) is one of the options to conceal the security logic and protocol from the original content so that it cannot be recognized. Technically, code obfuscation increases the difficulty of decompiling and does not impact on the normal operations of the ADCu. Thus it can efficiently reduce the possibility of decompilation. To test the response to this attack, we adopted the well-known obfuscator ProGuard (Zhong et al., 2010) to obfuscate a demo ADCu. We applied the default configuration of obfuscation. We also utilized the decompiler tool JAD (Kouznetsov, 2013) to decompile the obfuscated ADCu and the original ADCu. Figure 4.8 illustrates the partial decompiled codes for the original and obfuscated source codes. We can observe that the original source code without obfuscation can be decompiled to the point where it is almost readable. However, the script logic decompiled from the obfuscated ADCu becomes messy.



```
.........................................................
if(i == 0)
  {
 int primeOrder = PrimeTool.getInstance(). getPrimeOrder(0,
((Integer)primeSet.get(i)).intValue());
 e = (Element)element.get(primeOrder);
  } else
  {
 int primeOrder = PrimeTool.getInstance().
getPrimeOrder(((Integer)primeSet.get(i - 1)).intValue(),
((Integer)primeSet.get(i)).intValue());
  List ele = e.getChildren();
 if(((Element)ele.get(0)).getName().toString().equals("list"))
  {
  Element list = (Element)ele.get(0);
  List eleList = list.getChildren();
 e = (Element)eleList.get(primeOrder);
  } else
 {
  e = (Element)ele.get(primeOrder);
  }
  }
.........................................................
```



```
..........................................................
if(i == 0)
 {
s = PrimeTool.getInstance().getPrimeOrder(0, ((Integer)((List)
(obj)).get(i)).intValue());
obj1 = (Element)list.get(s);
 } else
 {
s = PrimeTool.getInstance().getPrimeOrder(((Integer)((List) (obj)).get(i -
1)).intValue(), ((Integer)((List) (obj)).get(i)).intValue());
if(((Element)((List) (obj1 = ((Element)
(obj1)).getChildren())).get(0)).getName().toString().equals("list"))
 obj1 = (Element)((List) (obj1 = ((Element) (obj1 = (Element)((List)
(obj1)).get(0))).getChildren())).get(s);
 else
 obj1 = (Element)((List) (obj1)).get(s);
 }
..................................................
```

Figure 4.8 Decompiled code for the original source code (upper) and the obfuscated code (lower)

Notably, we only obfuscated the active scripts portion before resembling to an ADCu. They would not impact on the encapsulated data. Moreover, if an ADCu is disassembled and the attacker breaks the obfuscated scripts the data blocks encapsulating sensitive data will be also revealed but in encrypted form. The data is still relatively secure due to the prior application of AES encryption on the data blocks. We used the RSA non-symmetric encryption method to encode the key of AES encryption and store it safely in trusted authorities. We chose not to use the RSA encryption for entire data blocks due to its feature that each 117-bit segment should be encrypted individually. When encrypting a whole data file, we would need to chop the data into a large number of pieces to encrypt, which would inevitably increase overheads. Hence, the computation overhead of RSA will be far more expensive than the AES scheme.

4.5.3 Tampering and Integrity Attacks

The adversary may also attempt to tamper with the script logic to conduct the jailbreak operation in order to bypass the verification and identification procedures and obtain access to the resources of the core once the ADCu is disassembled. Since the compiled scripts are obfuscated, we can simply insert a set of checkpoints throughout the whole ADCu. Any inconsistent outcome created by the checkpoints would be actively detected while the ADCu is running, and then cause its termination. This resembles the try-catch clause in the Java program. Only when all the checkpoints are passed through will the eventual commit operation be executed. Normally, these checkpoints are distributed at all key function points, such as verification, secure connection, and request delegation. To implement these checkpoints in the ADCu, we simply calculate the digest value of the checkpoints via the hash function and compare it with preconfigured digest value.

It is extremely difficult for adversaries to corrupt all the checkpoints in ADCu, for two reasons:

1) All checkpoint codes have been embedded in obfuscated scripts. Retrieving the original checkpoint functions and tampering with them to eliminate their functions is technically difficult.

2) Any functional and integrity-related corruption of the ADCu can be detected by verification of the code signature. We have implemented the integrity guarantee through the sealing and signature technology (Oracle, 2011) which is supported by JAR technology. Once the original script content or resource files change, the ADCu can detect the exception by re-signing its content. Figure 4.9 is the partial example of signed message of an ADCu, which is implemented by JAR sign tool.

```
Manifest-Version: 1.0
Implementation-Vendor: ARN Lab in Univerity of Technology,Sydney
Ant-Version: Apache Ant 1.8.2
Implementation-Title: ADCu (Active data cube)
Implementation-Version: 1.2beta
Created-By: 1.7.0_17-b02 (Oracle Corporation)
Main-class: arn.inext.Shell_Main
Build-By: cenling40

Name: org/jdom/filter/ElementFilter.class
SHA-256-Digest: D2xIagFWp2ZGlBhYI2jGUtvqbhHKrEMFlYLswyJNo5M=
Name: org/jdom/transform/JDOMSource$JDOMInputSource.class
SHA-256-Digest: QoNhAKCWH9Bsr3Uxq6KRUBWBM3dQAbM/renc9CKedJg=
Name: core/dataloader/Coder.class
SHA-256-Digest: rKEvH1PvBNttYIc2JUuuNOZLeiAVaHK0U7ou2HGdxIg=
……………………………………………………………………….
```

Figure 4.9 Signed message of a demonstration ADCu

4.5.4 Runtime Environment Attacks

Our ADCu relies heavily on the Java Runtime Environment (JRE). Once the JRE of the ADCu is corrupted, adversaries may leverage a puppet JRE to attempt to extract content through control of execution of the ADCu. However, even when it falls into the hands of adversaries without the appropriate JRE and the active features of the ADCu are killed. , the adversary who seizes control of the ADCu can retrieve nothing from the ADCu without the valid cryptographic key due to the obfuscated scripts and encryption on data blocks. A falsely behaved ADCu on a corrupted JRE will be detected by communicating with its supervisor and inner checkpoint log results. Moreover, we can periodically inspect the validity of the JRE through integrity examination of the JRE compared to the predefined and correctly behaved JRE instance checksum.

4.5.5 Man-in-the-Middle Attacks

The adversary may intercept communication messages between the ADCu and its corresponding supervisor and auditor, and attempt to convince them that they are communicating with each other over a correct and secure connection. In fact, the adversary may control the entire communication by man-in-the-middle attacks. This type of attack can be prevented by deploying the RMI-SSL communication protocol which guarantees both entities using a mutually trusted communication tunnel, and we have employed the ZK proof scheme between the supervisor and the ADCu to avoid interception by a third party entity. Mathematically, these two security protocols are sufficient to defend against man-in-the-middle attacks.

4.5.6 Host Compromise Attacks and Unpredictable Failures

The cloud data storage system stores users' data unordered and invisible. No sensitive information (such as data index or location information) is revealed. It is extremely difficult for adversaries to eavesdrop information targeted to the specific user. However, an adversary can still carry out an attack that compromises the whole data storage host. This can occur much like a natural disaster, which is unpredictable. The only way to reduce the impact of such attacks is to create a multi-replicas mechanism and distribute the replicas to geographically separate hosts. The details related to data replication management of the ADCu framework are discussed in Chapter 6.

4.6 Summary

This chapter introduces and discusses the data core protection layer in the proposed trust-oriented data protection framework. We propose a new approach based on bundling together sensitive data, metadata, and active scripts into an independent and constructed data structure called the ADCu. Through integrating self-verification, network communication, regular data operations, logging, and probing functionalities in the ADCu, and by deploying a supervisor instance in the same working domain as the ADCu to monitor OS-level data manipulation, we can guarantee data confidentiality, integrity, and intrusion tolerance. The evaluation illustrates that the proposed ADC framework can efficiently prevent data from direct access and intrusion attacks, reverse engineering and decompilation attacks, tampering and integrity attacks, runtime environment attacks, and main-in-the-middle attacks.

Chapter 5 Data Security Control Layer – CPRBAC and AAS

Chapter 4 illustrated the data core protection layer that encapsulates active scripts with data by the ADC framework. That layer is designed to enable data to protect itself while traveling in heterogeneous and distributed cloud scenarios. Confidentiality, integrity, and intrusion tolerance can be satisfied through the ADC framework.

This chapter focuses on the data security control layer for the ADC framework. The aims are to meet authentication, authorization, and auditability of data access and manipulation. Two vital services are proposed in this layer: the cloud privacy-aware role based access control service and the active auditing service. The former provides an access control service that meets the requirement of cloud data access. The latter executes active auditing of users' data operations, collaborating with the CPRBAC service to maintain consistency and security of data manipulation in the trust-oriented framework.

5.1 Foreword

The ADC framework provides active security protection whereby all actions and mechanisms are designed to prevent any breach of users' data and are able compulsorily to inform users of any violation occurred on their data when uncontrollable incidents or attacks occur. To establish more comprehensive security mechanisms on the ADC framework, we propose to build up a number of data security control services in the peripheral environment of the ADC framework, including an access control service and an auditing service. These components in the whole data protection framework contribute to the passive security schemes. In this chapter we explain these schemes using the healthcare cloud context for the purpose of illustrating the security context in a practical scenario.

5.1.1 Problem Definition and Requirement for Cloud Access Control Services

Cloud access control policy is normally orchestrated as the SLA made with users at the time that cloud consumers sign on to data services. Access control is an indispensable component for describing and defining the security boundary for data operations in cloud. Traditional access control models, such as mandatory access control (MAC) and discretionary access control (DAC) (Saidani and Nurcan, 2006), are not adequate to satisfy cloud data privacy protection due to the lack of basic components required by privacy regulation. As well, they may cause enormous access lists when are applied to the distributed healthcare cloud context. The role-based access control (RBAC) model (Jin et al., 2008) can provide permission administration for a large number of users through authorizing permission to users in terms of their roles. However, the RBAC model is not an inadequate way of fulfilling access control requirements in a highly dynamic cloud scenario due to its design for closed domains such as centralized servers. In the dynamic and distributed cloud, we identify the following demands that can affect security measures when we propose access control services:

1) Heterogeneity of services in the distributed cloud environment necessitates various degrees of granularity in access control services (Almutairi et al., 2012). The risk of unauthorized access to cloud data may be introduced if inadequate or unreliable authorization procedures are used.
2) Multi-tenancy requires an explicit and controllable authorized information flow among different cloud domains without interference.
3) Resource sharing and service collaboration requires access control policy inter-operation for secure service delivery.
4) The multi-domain cloud environment requires a policy structure that supports the delegation of users' permission on an organization basis and across multiple organizations for federated authorization concern.
5) The sophisticated security context and cloud management require fine-grained access control architecture to assist in reducing the risk of exceeding authority.

5.1.2 Problem Definition and Requirement for Cloud Audit Service

As well as access control systems, we believe that audit control is an indispensable and important scheme to ensure compliance with established access control policies and corresponding operational procedures. Traditional approaches (Wang et al., 2003, Wei et al., 2010a) have adopted periodic diagnostic schemes to verify audit records to further detect unauthorized usage or intermediate violations. However, these approaches may lack immediacy even though a number of algorithms can be applied to intelligently reduce latency of diagnosis. Once a violation occurs, it is difficult to ensure that the audit component can respond correctly and timely. We identify the following demands for audit control in the distributed cloud:

1) Audit procedures should be compulsorily conducted along with any security-related cloud operations and data manipulations to satisfy immediacy and authenticity. Audit records should be retrieved from reliable and security-related components in cloud, and they cannot be fake.

2) The distributed security components in cloud mean that audit control may involve multiple participants to fulfill an entire security flow.

3) The audit service should provide distributed transaction management to ensure data operation consistency.

4) The CSPs should validate the behavior of each party while executing data audit management in case of false accusations from clients.

5.2 Cloud Privacy-Aware Role Based Access Control Service

This section focuses mainly on the novel access control service designed for the cloud-based context. We present a novel framework that addresses the challenges and requirements proposed in Section 5.1.1.

5.2.1 Related Work

An access control mechanism is ordinarily employed to confine the actions or operations that only authorized users can perform (Sandhu and Samarati, 1994). From the perspective of cloud infrastructure, the access control service can be deployed in the virtualization layer, platform layer and application layer (Almutairi et al., 2012). Because the working domain on which we are focused is the application layer, we do not discuss the access control challenges in the other two layers. Moreover, access control services in the virtualization and platform layers entail various risks and issues. In the next section, we introduce various access control models and distinguish them from the model used here.

Discretionary access control

The DAC model refers to access restriction to resources based on the identity of subjects and/or groups (Alhaqbani and Fidge, 2008). The access controls are discretionary in the sense that a subject with certain access permissions can transfer permissions to other subjects. The identity of users is the key to execute DAC. Access control lists are the most commonly used technologies to manage authentication and authorization of data access.

Mandatory access control

With the MAC model the policy is normally controlled by a central authority (Alhaqbani and Fidge, 2008) rather than by the individual owner of an object. MAC refers to a type of access control by which the system constrains a subject to execute some sort of operations on an object (Wikipedia, 2013d). Compared with the DAC mechanism, protection decisions are decided not by the object's owner but by the security system. With DAC and MAC mechanisms in the distributed cloud it could be extremely difficult to manage the huge number of users who might be involved. Furthermore, these two mechanisms lack basic components to support privacy regulation and varying degrees of granularity in access control.

Role-based access control

In the RBAC model, each subject's access permission is determined based on his or her roles and privileges corresponding to those roles (Li et al., 2010). This mechanism can manage large numbers users with low cost to meet the requirement of cloud services. However, because of its design for closed domains such as centralized servers, RBAC is not a proper way of fulfilling the access control requirement in a highly dynamic cloud scenario. Furthermore, traditional RBAC is deficient for expressing complex access control scenarios due to the lack of specific policy components.

Attribute-based access control

Attribute-based access control (ABAC) extends the role concept in RBAC. It grants access rights to users through a combination of roles and attributes. Compared with RBAC model, ABAC has a more flexible structure to describe policy (Yuan and Tong, 2005). But, like RBAC, ABAC may be deficient in description of distributed cloud access control scenarios.

Other researchers (Danwei et al., 2009, Ruj et al., 2011) have also proposed access control in cloud scenarios. However, their mechanisms did not comprehensively satisfy the requirements analyzed in Section 5.1.1. We propose a novel access control model, CPRBAC, which is based on RBAC, but we extend it to satisfy the demands in the distributed cloud.

Cloud-based privacy aware role based access control

CPRBAC extends the model proposed in the previous work of our research group (Li and Hoang, 2009) and further introduces four new components: *Organization*, *Condition*, *Obligation*, and *Purpose*, to enrich policy description for complex usage requirements concerning authorization delegation, cross-realm role roaming, privacy-awareness, and active auditing. The foundational framework we employ is inherited from the conventional RBAC model that contains *Subject*, *Object*, *Role*, and *Operation*. However, it addressed the shortage in the traditional RBAC model that is not an inadequate way of fulfilling access control requirements in a highly dynamic cloud scenario due to its design for closed domains such as centralized servers. In the dynamic and distributed cloud environment, our access control model contributes to supporting heterogeneity of services, multi-tenancy, resource sharing and service collaboration, and sophisticated security context and cloud management. The following section will describe the detailed construction of CPRBAC model and its novelty and contribution.

5.2.2 Construction of CPRBAC Model

The model is illustrated in Figure 5.1. *Subject*, *Role*, *Object*, and *Operation* comply with the traditional definitions in the RBAC model. Next, specific descriptions of each component in the CPRBAC model are presented.

First, we consider the following scenarios:

1) Patients store their Electronic Health Record (EHR) in a healthcare cloud. Only the patient's general practitioner can access that EHR in the hospital's intranet.

2) A research student in University A wishes to access all patients' EHRs for medical research in the hospital. However, this permission is only assigned to the senior practitioner role in that hospital. But University A has a cooperative relationship with the hospital as long as the student has a valid research ID. In that case, the hospital can issue a temporary senior practitioner role for a limited time for the research student.

3) Different roles can access different data fields in the EHR simultaneously.

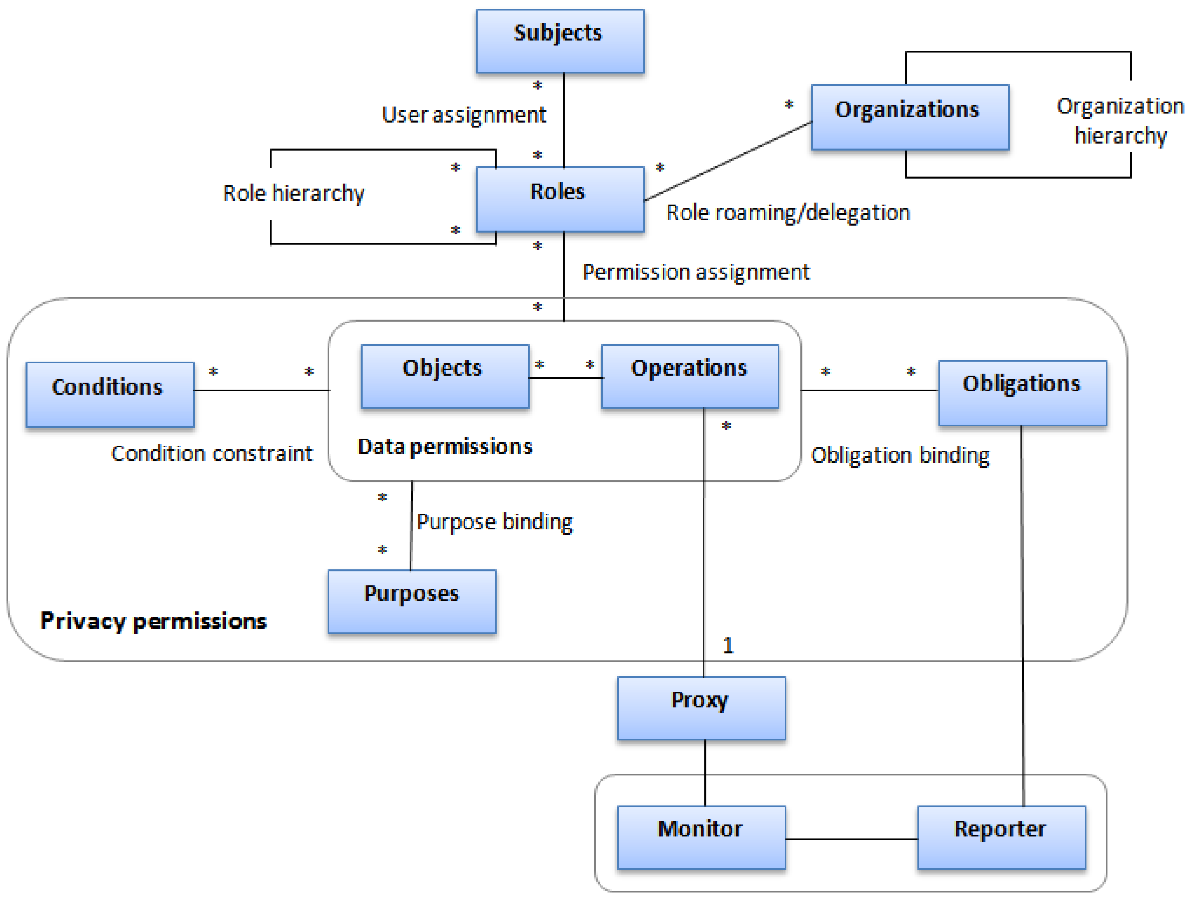


Figure 5.1 CPRBAC model construction

The above scenarios are very difficult to implement by current MAC, DAC, RBAC, and ABAC models. To achieve a comprehensive security control layer that supports fine-grained data protection in distributed healthcare cloud across multiple domains, we propose the CPRBAC model. Table 5.1 explains the notations used in our scheme.

Compared with the access control systems discussed by (Qun et al., 2009, Sandhu et al., 1996, Liu, 2010), our model extends the model proposed in the previous work of our research group (Li and Hoang, 2009) and further introduces four new components: *Organization*, *Condition*, *Obligation*, and *Purpose*, to enrich policy description for complex usage requirements concerning authorization delegation, cross-realm role roaming, privacy-awareness, and active auditing. The foundational framework we employ is inherited from the conventional RBAC model that contains *Subject*, *Object*, *Role*, and *Operation*.

Table 5.1 Notations used in the scheme description

Notation	**Description**
Subject (S)	Entity that accesses relevant objects
Object (O)	Information or data that relates to the identified S
Role (R)	Functional entity that associates someone with specific authority and responsibility within an organization
Operation (Op)	Operation that binds O and consists of a set of actions that S can execute
Organization (Org)	Domain identifier
Condition (Con)	Prerequisite to be met before any Op can be executed
Purpose (Pur)	Specifies the intended reason of the Op
Obligation (Ob)	Functions that must be executed before an Op is executed on O or after the execution

Condition, *Obligation*, and *Purpose* constitute the context-aware description set in cloud. Among these, *Purpose* describes a mapping relationship between a predefined universal unique identifier (UUID) and its corresponding property. It presents the action's intended purpose on data. For example, a practitioner can access the EHR of the patient's cancer history only when the purpose is for emergency treatment. *Obligation* consists of several parallel tasks and the corresponding tags indicating the order of execution for current tasks. It stipulates that a specific execution must be fulfilled at a certain time. For example, the obligation might be that data access must connect to the report service for auditing or logging when data is accessed. Data privacy protection relies tightly on the access *Purpose* and fulfilling of *Obligation*. Another vital component, *Organization*, is a domain identifier in the cloud environment. We generally map a user to the set of *Role-Organization* pairs to fulfill role roaming and role delegation in the distributed cloud context. *Condition* expresses the execution constraints to achieve eventual data operations. The action scope may be time (from 9AM to 5PM), age (more than 18 years old), network address (between 192.168.1.20 ~192.168.1.39), and so on.

An example of how the CPRBAC describes a scenario is:

"Practitioner role can read patient's EHR contents for treatment purpose only if the current time (CT) is between 9AM and 6PM, and patients will be informed by email"

This can be expressed as follows in CPRBAC policy:

<Practitioner, read, patient.EHR, Treatment, [CT (9AM- 6PM)], inform (Email)>

5.2.3 Data Permission Assignment

Definition 1: Data Permission Set and Assignment

•Data Permission set is denoted as $\mathrm{DP} \subseteq \mathrm{O} \times \mathrm{Op} \times \mathrm{Con} \times \mathrm{Pur} \times \mathrm{S(Ob)}$, where $\mathrm{S(Ob)}$ denotes the set of all subsets of Ob.

•Data Permission Assignment set is denoted as $\mathrm{DPA} \subseteq \mathrm{R} \times \mathrm{DP}$, which is a many-to-many mapping relationship between R and DP.

5.2.4 Role Assignment and Hierarchy

Role Assignment is an automatic procedure of authenticating user information and allocating a corresponding role to the user. Our approach in terms of role assignment is based on the notion of ABRA, which expresses the fact that the role of user is associated with a 2-tuple set including *Subject* and *AttributeSet*. Compared with conventional subject-role assignment, ABRA is capable of satisfying dynamic role assignment based on a number of context information items such as organization, time, etc., rather than simply achieving a corresponding mapping relationship between the subject and the role.

Definition 2: Attribute-Based Role Assignment

Let Subject be S which is defined as subject identifier, and AttributeSet AS is a set of context information such as Organization Or, Time T, etc., and the corresponding Role is R. Role assignment RA can be denoted as $\mathrm{RA} \subseteq (\mathrm{S}, \mathrm{AS}) \times \mathrm{R}\ (\mathrm{AS} = [\mathrm{Or}, \mathrm{T}, \dots])$.

Role Hierarchy is an important notion in the RBAC model that efficiently reduces the total number of permission assignment costs. Normally, it defines an inheritance relationship among roles, which is similar to an object-oriented programming term. From a mathematical perspective, role hierarchy is a partial order that is a reflexive, transitive, and asymmetric relation (Qun et al., 2009).

Definition 3: Role Hierarchy

• Role Hierarchy: $\mathrm{RH} \subseteq \mathrm{R} \times \mathrm{R}$, a partial order on roles

• If $\mathrm{R} < R'$ then all subjects assigned to R' are (implicitly) assigned to R

• If $\mathrm{R} < R'$ then all permissions assigned to R are (implicitly) assigned to R'

• If $\mathrm{R} \leq \mathrm{R}'$ then it implies that $\mathrm{P(R)} \subseteq \mathrm{P(R')}$, $\mathrm{P(R)}$ denotes the permissions of R

5.2.5 Organization Hierarchy

Organization Hierarchy defines the inheritance relationships among organizations, similar to role hierarchy in the conventional RBAC model. It can efficiently reduce the total number of permission assignment costs. From a mathematical perspective, organization hierarchy is a partial order that is a reflexive, transitive, and asymmetric relation, like role hierarchy (Qun et al., 2009). Figure 5.2 shows an example of a possible management hierarchy in the cloud healthcare context.

Role assignment in that context could include:

Users from the surgery department are only allowed to obtain roles related to this department.

Users from the emergency department are allowed to obtain roles related to this department and the sub-department below it.

Users from the emergency department are not allowed to obtain roles related to the in-patient department unless the users satisfy a temporary delegation requirement.

Users from hospital A's management level are allowed to obtain all related roles under the hospital.

Users from a lab in university B are not allowed to obtain related roles in hospital A unless hospital A has a correlation with University B and also the users satisfy a temporary delegation requirement.

Based on analysis of the above scenarios, role assignment may need to be delegated in internal or external organizations to meet the dynamic access control requirement in cloud.

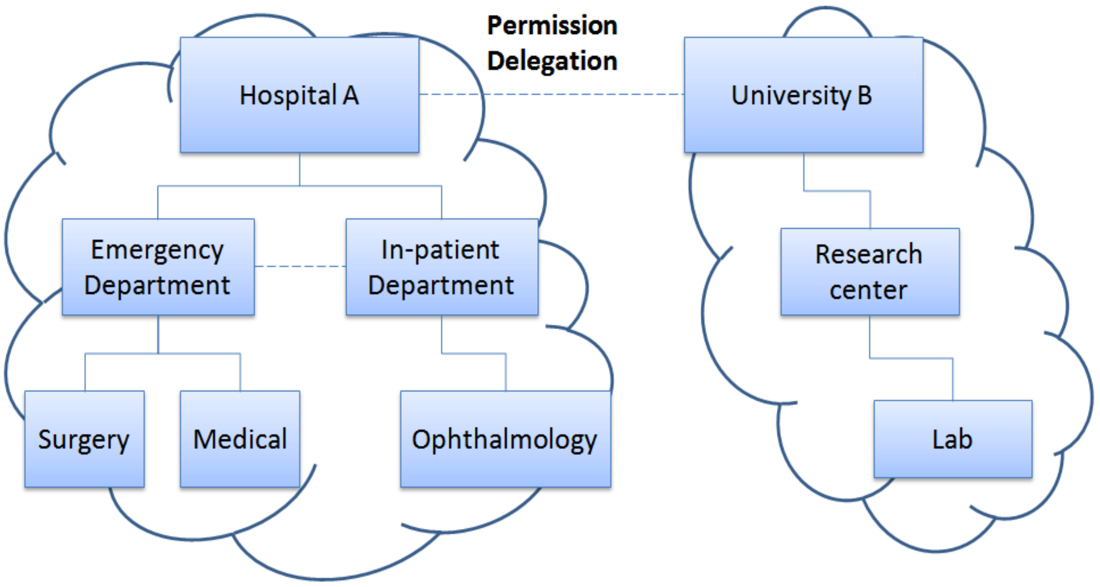


Figure 5.2 A possible management hierarchy in the cloud healthcare context

5.2.6 Role Delegation and Role Roaming

Role Delegation denotes permitting users to delegate certain capabilities to others under a number of restricted conditions within the same organization. For instance, one specialist can delegate permission to an idle general practitioner when he/she is busy or unavailable. Note that the role delegation scheme is only conducted in a single-domain scenario.

Definition 4: Role Delegation

Let original Subject be OS, delegated Subject be DS, the Role of OS be Ros, the Role of DS be Rds, Condition be Con. Role Delegation is expressed as $\mathrm{RD} \subseteq (\mathrm{OS}, \mathrm{Ros}) \times \mathrm{Con} \times (\mathrm{DS}, \mathrm{Rds})$.

Role Roaming refers to a role delegation scheme in the multiple-domain healthcare cloud. The scheme allows a user in an organization to gain external authentication information specified in another organization through a mirroring cross-domain role.

Definition 5: Role Roaming

Let $\mathrm{R1}$ be the role in Organization $\mathrm{O1}$, $\mathrm{R2}$ be the role or role set in Organization $\mathrm{O2}$, the roaming delegation condition be Con. The role roaming scheme can be expressed as $\mathrm{RR} \subseteq (\mathrm{R1}, \mathrm{O1}) \times \mathrm{Con} \times (\mathrm{R2}, \mathrm{O2})$, which indicates that $\mathrm{R1}$ in $\mathrm{O1}$ can map to a corresponding role or one of the role set $\mathrm{R2}$ in $\mathrm{O2}$ when the delegation meets the condition Con.

Here is an example of a healthcare scenario: Let *R1* be a PhD research student in the medical school in University B, who would like to analyze some patients' EHRs for research, but *R1* only has a role account as a PhD research student. The hospital organization *O2* has some roles *R2* that have permission to read patients' EHRs, but these roles are hospital local roles. Now, *R1* can log in the system of EHRs in *O2*, after verifying that *R1* is an external trusted role. The EHRs system in *O2* delegates one of *R2* for this research student, who is then able to enforce relevant permission as *R2* authorizes.

5.3 Implementing CPRBAC Prototype

CPRBAC policy definition is an important component that delivers a consistent authorization service for provisioning data access in our research context. The proposed profiles in the CPRBAC model can handle sophisticated security and privacy protection requirements and provide flexible mechanisms for users to specify the access rules for their data. To implement these demands, we extend the traditional XACML model (XACML, 2011). In the following sections, we introduce the specifications related to implementation of the CPRBAC model.

5.3.1 Introduction of XACML

XACML stands for eXtensible access control markup language. It has become a standard for defining a declarative access control policy language on the XML basis (Wikipedia, 2013i). The standard also specifies a processing model that evaluates requests for resources against a set of rules defined in policies (Dinh et al., 2012). The authorization result returns a permit or a deny decision on the access. Due to the common terminology and inter-operability of XACML, it is suitable to fulfill distributed authorization across multiple domains. A similar technology P3P system, in which the privacy statements are also formatted using XML and a common vocabulary can be automatically retrieved and interpreted, has been well established in many websites to express privacy regulations (Ni et al., 2010, Petković and Jonker, 2007). However, P3P was designed only to support coarse-grained high-level privacy declarations and thus it cannot distinguish different types of access. Another representative technology, enterprise privacy authorization language (EPAL) (Ashley et al., 2003), has been proposed to encode enterprises' privacy-related data handling policies and practices enforced by a privacy-enforcement system. Both EPAL and XACML can provide flexible, high-level policy orchestration, but EPAL adopts a first-applicable mechanism that may cause problems while processing access requests (Barth et al., 2004). Compared with EPAL, XACML provides multiple conflicting resolution schemes to reach a final decision.

5.3.2 Elements of XACML

Figure 5.3 is an example of a XACML policy indicating that a subject with the Physician role can read cenling40's health record between 9am and 5pm local time.

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        PolicyId="1"
        RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
algorithm:ordered-permit-overrides">
  <Description>
    Between 9am and 5pm local time, allow Physician to read cenling40's
```

```
health record.
  </Description>
  <Target>
    <Subjects>
      <AnySubject/>
    </Subjects>
    <Resources>
      <Resource>
        <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#anyURI">cenling40.tdfs</Attribute
Value>
          <ResourceAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#anyURI"

AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"/>
        </ResourceMatch>
      </Resource>
    </Resources>
    <Actions>
      <Action>
        <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-
equal">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
          <ActionAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#string"

AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"/>
        </ActionMatch>
      </Action>
    </Actions>
  </Target>

  <Rule RuleId="EveryoneDuringBusinessHours" Effect="Permit">
   <Target>
        <Subjects>
          <Subject>
                    <SubjectMatch

MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                        <AttributeValue

DataType="http://www.w3.org/2001/XMLSchema#string">Physician</AttributeValue
>
                        <SubjectAttributeDesignator

AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"

DataType="http://www.w3.org/2001/XMLSchema#string"/>
                    </SubjectMatch>
                </Subject>
        </Subjects>
        <Resources>
          <AnyResource/>
        </Resources>
        <Actions>
          <AnyAction/>
        </Actions>
      </Target>
    <Condition
FunctionId="http://research.sun.com/projects/xacml/names/function#time-in-
range">
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-
only">
        <EnvironmentAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#time"

AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"/>
      </Apply>
      <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#time">07:00:00</AttributeValue>
      <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#time">20:00:00</AttributeValue>
    </Condition>
  </Rule>

  <Rule RuleId="DenyAllOthers" Effect="Deny"/>
</Policy>
  </Rule>
  <Rule RuleId="DenyAllOthers" Effect="Deny"/>
</Policy>
```
Figure 5.3 Example of a XACML policy

The definition of XACML core specification (Parducci et al., 2010) is structured into three levels of elements: *PolicySet*, *Policy*, and *Rule*. A *PolicySet* can contain multiple *Policy* elements or sub-*PolicySet* elements. A *Policy* can contain multiple *Rule* elements. As the most elementary unit of *Policy*, *Rule* usually expresses an isolated access-control unit. It is evaluated on the basis of its contents. However, to fulfill multiple access-control rules on one subject, we can encapsulate them in a *Policy*. Notably, due to the fact that the evaluation results of rules are not exchangeable among system entities (Parducci et al., 2010), to generate an eventual evaluation result without generating a possible conflict it is necessary to apply the rule-combining algorithm which is specified as a standard by (Parducci et al., 2010). As well as the rule-combining algorithm, the policy-combining algorithm is applied in *PolicySet* to create the eventual results of multiple *Policy* evaluations without conflict.

5.3.3 Combining Algorithms for Executing XACML Evaluation

In the CPRBAC evaluation process, six rule- and policy-combining algorithms are applied in our framework:

(1) Deny-override:

This indicates that a deny decision has higher priority than a permit decision. If there is a deny decision in the *PolicySet* or *Policy*, the evaluation result is “Deny”.

(2) Ordered-deny-override:

This is identical to the “Deny-override” combining algorithm except for the following: the evaluation of a collection of policies or rules must match the order listed in the PolicySet or Policy.

(3) Permit-override:

This indicates that a permit decision has higher priority than a deny decision. If there is a permit decision in the PolicySet or Policy, the evaluation result is “Permit”. It is exactly the reverse of the Deny-override algorithm.

(4) Ordered-permit-override:

This is identical to the “Permit-override” combining algorithm except for the following: the evaluation of a collection of policies or rules must match the order as listed in the PolicySet or Policy.

(5) First-applicable:

This indicates that each policy or rule is evaluated in order in the policy set. If the evaluation result is a determinate value of “Permit” or “Deny”, then the evaluation will halt and the eventual result will be the effect value of that policy or rule. If the evaluation result is “NotApplicable”, then the next policy or rule in the order will be evaluated. If no further policy or rule exists in the order, then the eventual result will be “NotApplicable”. If an error occurs when evaluating policies or rules, then the eventual result will be “Indeterminate”.

(6) Only-one-applicable:

This indicates that if only one policy or rule in the policy set is evaluated as “Applicable”, then the result of evaluation of that policy will be the eventual result. If no policy or rule is considered applicable, then the result of evaluation will be “NotApplicable”. If more than one policy or rule is considered applicable, then the result will be "Indeterminate". If an error occurs when evaluating policies or rules, then the eventual result will be "Indeterminate".

5.3.4 Execution Modules of CPRBAC Service

CPRBAC provides legislation verification in the transaction of executing an active auditing service. It executes the validity analysis of user requests in terms of the designated policy in the CPRBAC service. A CPRBAC service resembles a security firewall that obstructs unauthorized entities and actions. It presents a simple and unified interface for authorizing users to access and process correct objects based on stored policies. Figure 5.4 depicts a standard workflow diagram of the CPRBAC service in which the procedures from the step 1 to step 5 comply with XACML architecture standard (XACML, 2011).

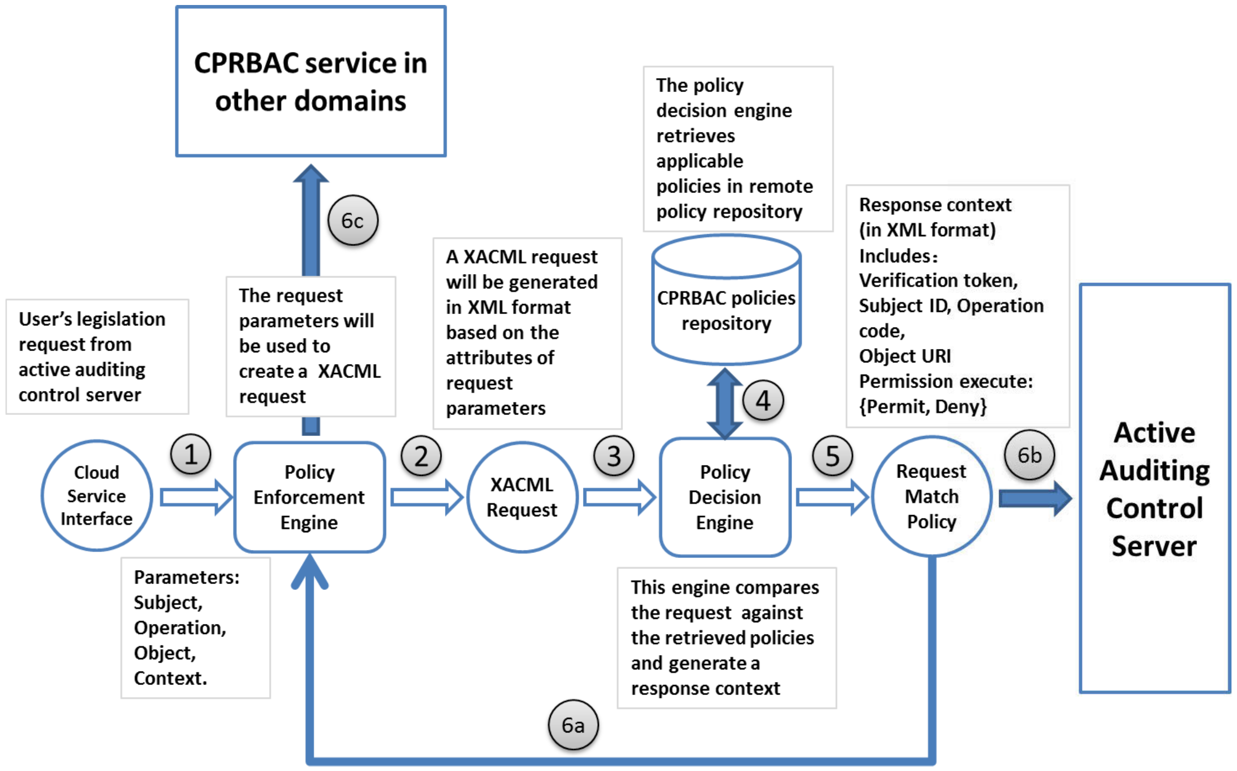

CPRBAC service in other domains
The policy decision engine retrieves applicable policies in remote policy repository
Response context (in XML format) Includes: Verification token, Subject ID, Operation code, Object URI Permission execute: {Permit, Deny}
6c
A XACML request will be generated in XML format based on the attributes of request parameters
CPRBAC policies repository
User's legislation request from active auditing control server
The request parameters will be used to create a XACML request
4
Active Auditing Control Server
1
2
3
5
6b
Cloud Service Interface
Policy Enforcement Engine
XACML Request
Policy Decision Engine
Request Match Policy
Parameters: Subject, Operation, Object, Context.
This engine compares the request against the retrieved policies and generate a response context
6a

Figure 5.4 CPRBAC service workflow

The requester first sends an access request to the *policy enforcement point* (PEP) module, which must enforce the access decision that will be taken by the *policy decision point* (PDP).

The PEP module then sends the access request to *context handler* that translates the original request into a canonical format as a XACML request. The request context provides attribute values such as *subject, resource, purpose,* and *action*. The *environment* module provides a set of attributes that are relevant to *condition* information such as *time, organization, etc*.

The context handler sends the XACML request to the PDP to identify applicable policies or rules. The PDP then evaluates the policies and returns the XACML *response context* with *obligation* set information.

Granted response contexts are issued a verification token (VT) in step 6b. To achieve uniqueness, atomicity, dynamism, and confidentiality of the VT, we calculate it as follows:

VT= Hash (timestamp, random code, subject ID, object URI, action code)

(Time-to-live (TTL) depends on the service requirement, random code is calculated by GUID)

A valid VT representing the current request is authorized by the CPRBAC service. The TTL will restrict the valid period of the VT. Furthermore, to exchange security and privacy policies, the policy enforcement engine also possesses a service interface that can interact with CPRBAC services in diverse domains if federated authorizations are required. This response option is shown in step 6c.

5.3.5 Evaluation Performance Improvement

Although XACML specifications provide a high-level access control language to protect resources based on XML technology (Demchenko et al., 2009), evaluation of policy would suffer significant bottleneck along with the increase of complexity and scale of policy semantics. Hence, performance optimization of XACML is significant. In related work, (Kolovski et al., 2007) improved performance through policy optimization that used descriptor logic (first-order predicate logic) to formalize XACML policy. The work AXESCON XACML (ax2e) also enhances the policy retrieve and cache function in the evaluation engine to improve evaluation performance. However, because the CPRBAC model in this thesis is designed to solve cloud-based access control demand, such performance enhancement is beyond the scope of this thesis. This issue will be addressed in our future work.

5.4 Active Auditing Service

Audit is regarded as the internal and external processes implemented by an organization to monitor or inspect whether the configured policies, procedures, and processes are consistently followed according to the identified requirements that are driven by business objectives, laws and regulations, customer contracts, internal corporate policies and standards, or other factors (Mather et al., 2009). In the cloud data storage context, audit management is an indispensable and important strategy to ensure compliance with established access control policies and corresponding operational procedures on outsourcing data (ADCus). Audit management plays a significant role in maintaining a trustworthy relationship between CSPs and users.

To implement a sustainable audit mechanism that satisfies the requirements discussed in Section 5.1.2, we propose an AAS that works on a transaction-based mechanism. In the entire security architecture, policy compliance (provided by CPRBAC service), validity of processes and procedures (provided by cloud access interfaces), external OS-level monitoring (provided by the supervisor instance), and internal operation logging (provided by ADCus) are the essential audit factors for identifying whether the requirements of cloud data protection are fulfilled. Each of the components (the CPRBAC service, cloud access interfaces, supervisor instance, and ADCus) actively manages and informs its own audit data to the audit service. We import a transaction management mechanism in the audit session to detect any inconsistency against policy, process, or procedure while executing data access on ADCus.

5.4.1 Audit Data Operations in Active and Transactional Manner

Several researchers (Wang et al., 2003, Wei et al., 2010a) have adopted a periodic diagnostic scheme to verify audit records to further detect unauthorized usage or intermediate violations. However, these approaches lack immediacy even though appropriate algorithms can be applied to intelligently reduce the latency of diagnosis. Once a violation occurs, it is difficult to ensure that the audit components can respond correctly. Hence, we employ an AAS to ensure that all participants in a cloud transaction session adopt the push mechanism to actively synchronize audit information with the audit control modules. Furthermore, to avoid the audit process introducing any new vulnerability of unauthorized information leakage or audit information fraud, we employ a distributed transaction scheme in the auditing service to provide data operation consistency. The transaction manager maintains surveillance of the progress and coordinates the eventual commit or rollback according to the transaction requirements. We observe that data transaction management is fundamental to ensure that only access-agreed, consistent, and acceptable state changes are allowed to execute in cloud systems. The transaction scheme can roll back data or resource status to the previous acceptable state if intermediate violations or system failures occur. As the final decision of permitting access to a specific ADCu, the transaction module only commits the action when all participating components behave correctly and the commit policy is reached.

In the cloud environment, the main challenge of providing transaction support is to provide ACID satisfaction: atomicity (A), consistency (C), isolation (I), and durability (D), without compromising the scalability and distribution features of cloud services (Zhou et al., 2011). Compared with conventional transaction mechanisms (Mikalsen et al., 2002), there is no common transaction context representation, semantic, and coordination protocol in the cloud service environment. Diverse and incompatible transaction models or other third party middleware may be involved in a transaction session. Moreover, service coordination, management, and context representation must be implemented in a decentralized, decoupled manner as autonomous participants. We address these challenges through a cloud transaction framework cooperating with the AAS, as illustrated in Figure 5.5.

An active audit transaction flow consists of five interdependent states: Request analysis >> Transaction preparation >> Legislation process >> Ballot process >> Commit process. These five states are defined in an ordered set and executed in sequence. For instance, before arriving at a transaction preparation state, the request analysis must accomplish and respond correctly. A false or unreachable state causes a rollback operation. Each state is performed atomically. Either a true or a false outcome is generated at the end of the state. The participators in each transaction flow are responsible for the individual task in the isolated domain, and eventually, their analysis results and statuses are gathered for a ballot process. Notably, the time for processing each state is limited. If the duration of executing the current state exceeds the threshold, an exception outcome is given. In the following sections, we discuss each of the states in detail.

5.4.1.1 Request Analysis

A request on a specific ADCu first requires an availability analysis. It is necessary to ensure that the new transaction will not generate conflict with a previous active transaction on the ADCu. Normally, conflicts occur in update-type transactions (such as create, update, and delete operations) and external-type transactions (copy, move, and remove). Update-type operations result in internal data modification inside ADCu and their action scope is usually constrained to the data blocks, whereas external-type operations deal with the whole ADCu as a unit. When two transactions attempt to conduct such manipulations on the same scope of a data block, a conflict may occur. This would create inconsistent ADCus when data operation commits.

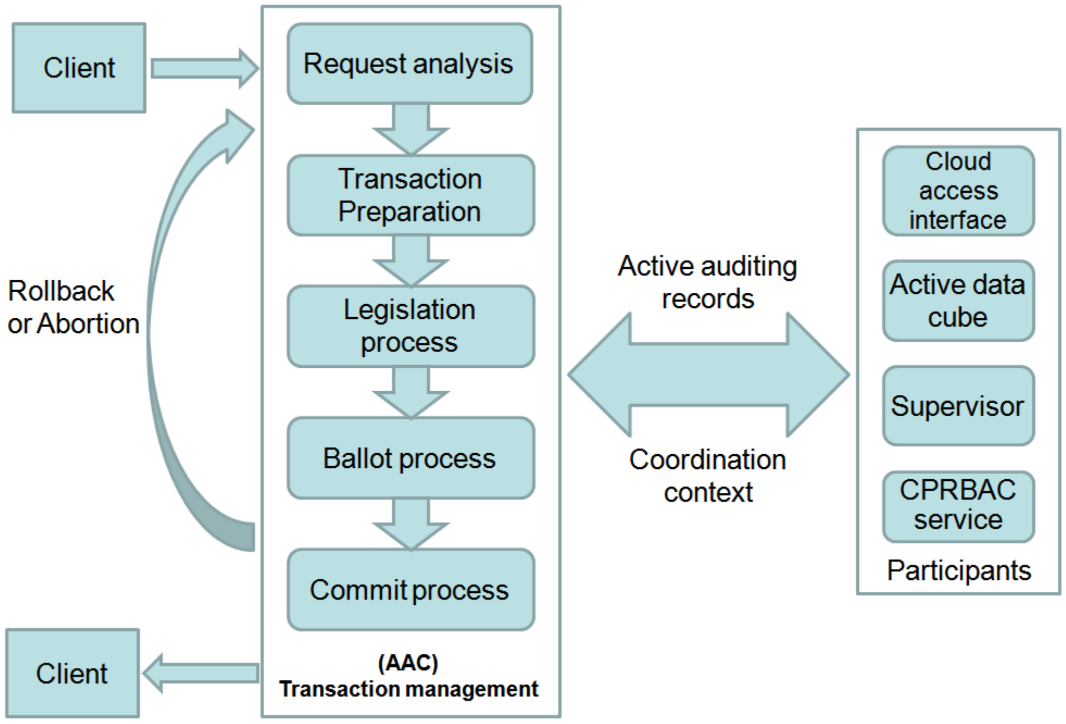


Figure 5.5 Active audit control transaction flow

However, the ADCu supports data update operations in parallel if the requests work on different scopes of data blocks or read-only type data operations. Furthermore, scalability can be simply implemented by creating multiple replicas of the ADCu and distributing them across decentralized cloud servers. Read-only operations have no data item update during the commit phase. Hence, a replica of the ADCu can be destroyed automatically after the transaction is committed. Update-type operations must execute a merge operation with the old ADCu. The conflict detection algorithm is described below:

```
------------------------------------------------------------------------------------------------
Algorithm. Conflict Detection
Input: String operation, String objectURI, String attributeTag
Output: Boolean (True is existing conflict in current transaction session)
------------------------------------------------------------------------------------------------
isExistConflict=false;
Synchronized(this) {
if (!TransactionMap.getInstance().TDFS_V_TEMP.isEmpty()) {
   Iterator it = TransactionMap.getInstance().TDFS_V_TEMP.entrySet().iterator();
        while (it.hasNext()) {
         try{
         Map.Entry entry = (Map.Entry) it.next();
         String TDFS_URL = (String) entry.getValue();
         if (TDFS_URL.equalsIgnoreCase(objectURL)) {
           // first check its isLock
              if (TransactionMap.getInstance().lock_map.containsKey(entry.getKey())) {
             boolean isLock = TransactionMap.getInstance().lock_map.get(entry.getKey());
             if (isLock) {// if locked, return conflict exists
               isExistConflict = true;
               return isExistConflict;
             }
            }
                 String txID = TransactionMap.getInstance().reflection_map.get(entry.getKey());
           MoCAsH_Transaction mt =
TransactionMap.getInstance().transaction_map.get(txID);
           try {
             JarResources JR = new JarResources(TDFS_URL);
   int requestTagCode = Integer.parseInt(JarOperation.getInstance().getCode(mt.
arguments[5], JR));
   int attrTagCode = Integer.parseInt(JarOperation.getInstance().getCode(attribute- Tag, JR));
                      if (isAncestorDescendant(requestTagCode, attrTagCode)) {
                    isExistConflict = true;
                       return isExistConflict;
              }
            } catch (IOException e) {e.printStackTrace();}}
         }catch(Exception ex){}}
```

If a conflict is detected in the request analysis, the transaction manager will deliver the request to the reservation queue corresponding to the specific ADCu. Once the conflicted transaction is committed, the transaction manager executes the request in the reservation queue.

5.4.1.2 Transaction Preparation

This is the second stage of a transaction session. It involves an *activation service*, a *registration service*, and a *coordination service*. After the request is validated, this preparation phase first activates an *activity* that contains the required *participants* in the current transaction session. It is the entry point of an active transaction. In designing a registration framework, we define a participants list (PL) as shown in the following list for different requirements according to transaction types. Normally, an update-type operation requires the following set of *participants* engaged in the transaction session. Due to the fact that multiple participants may be involved in a transaction, we adopt a select algorithm to optimize the transaction process.

$$< (\text{CPRBAC})_i,\ (\text{Supervisor})_j,\ (\text{Cloud access interface})_k,\ \text{ADCu}_x >$$

where i, j, k, x indicate the available serial number of current services or instances that have the optimal selection result, which is decided by cloud VM status, such as CPU usage, memory, throughput, and concurrent tasks. Each *participant* in the transaction is certified by a SHA-1 thumbprint.

Upon completion of the *activation service*, the transaction session contacts the required *participants* in the list to register the session with their specific context information. After registration, the *coordination service* takes charge of the transaction management task. Audit records that contain transaction context information are pushed asynchronously to the *coordinator* to commit a final decision. To avoid sensitive data leakage from transaction context information we deploy the secure communication tunnel. Possible solutions include SSL, SSH, etc. Furthermore, as a hundred thousand transactions may run per second in a large scalable cloud service, to identify different *participants* from different transactions and avoid possible conflicts we generate signatures for each *participant*.

5.4.1.3 Legislation Process

If the transaction session arrives at this point, an active transaction will be established. It executes the validity analysis for the request in term of the designated policy in the CPRBAC service. A detailed introduction of the legislation process was introduced in Section 5.3.4.

5.4.1.4 Ballot Process

Once the legislation process is completed, the granted request proceeds to the ballot process which is the final stage before committing the transaction session. In this stage, the transaction manager must ensure that all the *participants* behave legally and consistently with policy and procedure. We now explain this process with an example of an update-type operation.

Transaction management: We assume that the activity is an data update operation. We define the ballot-set as < $(\text{CPRBAC})_i$, $(\text{Supervisor})_j$, $(\text{Cloud access interface})_k$, ADCu_x >. The given active transaction is committed only if the following condition is satisfied:

$$\textit{Eval}\ (\textit{CPRBAC}_i(\textit{args}[\,])) \cap @\textit{-consistent}\ (\textit{Supervisor}_j) \cap \$\textit{-consistent}\ (\textit{ADCu}_x) \cap \textit{Work}\ (\textit{Cloud access interface}_k) \cap t<T(m) == \textit{true}$$

Eval $(\textit{CPRBAC}_i(\textit{args}\ [\,]))$ indicates the evaluation result towards the request from the CPRBAC service. The result consists of <Verified token, Session_GUID, CPRBAC_GUID, CPRBAC_certificate, Operation, Subject, Object> and returns TRUE if these elements are consistent.

@-consistent $(\textit{Supervisor}_j)$ represents the supervisor's status on the data update operation, and returns TRUE if the correct code is received.

$-consistent $(\textit{ADCu}_x)$ refers to whether the ADCu is intact during storage, which can be verified by recomputing the checksum and comparing it with the stored checksum in its header.

Work $(\textit{Cloud access interface}_k)$ is the working status of cloud entry points (usually refers to the status of cloud web servers, such as tomcat, apache servers).

$T(m)$ is the threshold of transaction time. We set it in order to constrain the maximum execution time for ballot process.

In the transaction preparation phase, different commit condition-sets may be defined for different security levels. All participants attending the transaction must also forcibly log the operation information. Normally, a safe transaction is finally committed, but an unsafe transaction is forced to roll back and the replicated ADCu is discarded.

5.4.1.5 Commit Process

Once this stage is reached, the transaction is committed. For a read-only operation, the ADCu is triggered by its supervisor to execute the data query operation and return the data requested by the user. After commit, the transaction is permanently destroyed. For an update-type operation, the supervisor first executes data update by external data operation interfaces, and then recalculates the header and refreshes it. External-type operations are required to invoke the OS's data operational command. It is worth noting that the commit process cannot roll back, to ensure eventual consistency in data operation. All other states can roll back the state to previous one that has a mutual dependency relationship.

5.4.2 Attestation Record of Audit Participants

In this section, we describe the structure of the attestations used by CSPs to defend themselves against false accusations from clients. When users consume cloud data services, the cloud generates corresponding attestations that are the key elements for attesting the behavior of each party while executing data audit management. If users' active data detects an exception or violation, the data is triggered to inform its bundled user that a violation has occurred on its cloud data. Details of the violation are generated by the ADCu. Hence, an attestation cannot be obstructed by third parties. On the basis of these details, users may attempt to accuse the CSP in order to obtain compensation for the CSPs' misbehavior. From the perspective of CSPs, if they cannot prove that the violation was not deliberate, they may face false accusations. In our framework, to avoid such circumstances, the key mechanism we use is that each participant must produce corresponding attestation records that keep the cloud accountable and prove whether the behavior of the CSPs has satisfied the SLA.

Figure 5.6 illustrates the structure of the attestation. Each attestation generated from the participants involved in one transaction session is made up of several concatenated data fields, and a signed hash value of these data fields. The *participant ID* is used to identify the participant in a transaction session, and each participant generates a unique attestation to prove that it has accepted and executed the corresponding request and the response has been given. The *certificate key* is the fingerprint of a participant, which is a 128bits/256bits SHA-1 message. The *operation code* is the function code that participants have executed. The *new hash* is computed over the concatenation and then the result is signed after a new activity. The *nonce* is a random value created by the participant to increase the difficulty of decrypting attestation records.

PARTICIPANT ID	CERTIFICATE KEY	OPERATION CODE	NEW HASH	NONCE

Figure 5.6 Structure of attestation records

A complete transaction receives a collection of attestations from each participant involved in the transaction session. To avoid adversaries from inserting false attestations between the collections of attestations and producing a false proof, we propose a chained hash scheme over an entire transaction session. It is a hash over the participants in the current attestation and the chained hash value of previous attestations. The chained hash is computed by the following equation:

$$\textit{Chained hash} = \textit{hash (current participant, (previous chained hash))}$$

For instance, we have three participants (*A, B, C*) involved in a transaction session and the chained hash in participant *C* is calculated by participant *B* and its chained hash (computed by participant A and the chained hash of A). Due to the fact that the chained hash represents the history of all the attestations, it is computationally infeasible to insert an attestation between these three participants. It is theoretically impossible for adversaries to penetrate the attestation structure. When cloud users want to accuse CSPs, the stored attestations can provide the proof that the cloud behavior has satisfied the SLAs.

5.5 Summary

In this chapter we have presented two key data security control schemes. The CPBRAC service is proposed to meet fine-grained access control requirements in the distributed cloud context. The AAS is proposed to execute active auditing of users' data operations, and it collaborates with the CPRBAC service to maintain consistency and security of data manipulation in the trust-oriented framework. These two key data control components play significant roles on maintaining data authentication, authorization, auditability, and accountability in the peripheral environment of the ADC framework.

Chapter 6 Data Operation and Management Layer – Full Mobility Management and Data Replica Management

Chapter 5 detailed the data security control layer that mainly focuses on addressing authentication, authorization, and auditability of ADCus. It is tightly incorporated with the data core protection layer to provide access control and active auditing based on the ADC framework and distributed cloud context.

This chapter focuses on the data operation and management layer. We address data full mobility management that efficiently solves the issues of accountability and mobility of data and users. The proposed active binding framework enables CSPs to provide trustworthy transparency of data usage and data security warranty for data owners. Another significant work is addressing data replica management issues related to the ADC framework. The proposed adaptive replica management scheme presents good performance and efficiency to adapt to fluctuating request traffic and can provide more flexible adjustment of replicas at low operational cost. Moreover, the adaptive consistency scheme is more flexible to meet dynamic data update requirements with diverse traffic. Accountability, availability, and scalability can be satisfied via the configuration of this layer.

6.1 Cloud Access Interfaces

In the outermost layer of cloud data storage services, cloud access interfaces provide and specify a protocol for accessing, administering, and provisioning cloud storage. There are three main aspects related to these interfaces:

Request recognition: Cloud resource access services that are normally delivered through web service invocation interfaces such as RESTFUL stateless and SOAP interfaces described in a machine-processable format (Wikipedia, 2013a).

Request distribution: To achieve workload balance, we distribute users' requests to web servers with relatively light workload through optimal selection.

Communication security: Network security in consuming cloud resources can be guaranteed by deploying SSL/TLS and https protocols (Wikipedia, 2013c).

6.2 Addressing Data and User Mobility Challenges in Cloud

Although cloud computing brings many potential benefits, it unavoidably exposes vulnerabilities related to data security, privacy, and control issues, due to its outsourced nature (Wang et al., 2010). Cloud users are reluctant to transfer sensitive data to the cloud for storage unless CSPs can provide trustworthy transparency of data usage and warranty of data security.

6.2.1 Issues Statement

So far, little investigation has focused on data mobility management. In a realistic cloud scenario with complex and dynamic hierarchical service chains, data handling may be delegated from one CSP to another for business reasons (Sundareswaran et al., 2011). Often these CSPs do not employ the same protection schemes and standards (Foster et al., 2008). Existing solutions lack support for data delegation in a secure manner. Furthermore, it is conceivable for CSPs to accidentally violate the SLA established between users and CSPs by moving a user's data to data centers located outside the particular geographic boundaries, to cut costs (Albeshri et al., 2012). Users are concerned that their data should remain in a location that conforms to the restricted scope of their SLA with CSPs. When a violation against the SLA of the cloud occurs, general schemes to detect this violation in CSPs rely on third-party auditors to periodically monitor and audit data. However, this may not satisfy users' requirements. On the one hand, CSPs may assert that a smaller amount of damage was done or even bury the violation incident to pacify their customers; on the other hand, a periodical audit scheme makes it difficult to guarantee timely detection of violation. Adversaries may exploit this vulnerability to compromise data and depart without a trace. Hence, ensuring that CSPs compulsorily, promptly, and truthfully inform data owners of a violation is regarded as a potential challenge. Realistically, it is impossible to fully prevent data from violations or attack, but if CSPs are required to report any evidence of data violation to their users, they would be more inclined to adopt cloud solutions for their businesses, as they can build more acceptable SLAs with their subscribed CSPs in a more trustworthy relationship. We believe that such a transparency scheme would be acceptable to users. Therefore, ensuring that users receive evidence of their data's violation from anywhere in cloud, at any time, also presents a real challenge.

6.2.2 Full Mobility Management Framework

This subsection addresses the above concerns from a novel perspective, offering full mobility management of cloud data to enhance transparency and security. To tightly bind data to users, the proposed full mobility management scheme focuses on two aspects: data mobility management (DMM) and user mobility management (UMM). DMM deals with the physical location changes of users' data in cloud, and ensures that users know whether the new physical locations conform with the SLA established at the time of subscription to the CSP. UMM is related to the user's location, which can be regarded as the location of the user's mobile terminal. Even when the users are roaming in a hybrid wireless network environment, they need to be reliably informed about the status of their data. Considering the above requirements, we propose a pervasive informing (PI) framework and a pervasive knowing (PK) framework. The PI framework enables any data violation against the SLA to be compulsorily informed by the data per se to the users. The PK framework keeps user's knowing of data consistency and data status. Once the consistency or knowing is broken, it indicates that the data is under risk, and the user can pursue the CSP via legal processes to obtain corresponding compensation. In normal circumstances, the PK framework can be regarded as a data state monitor. The data state cannot be forged technically by adversaries or CSPs.

Hardware infrastructure has recently matured with virtualization and high flexibility to achieve on-demand resource provision. One of the research goals is the provision of full mobility management capability on unstructured data based on ADCus. Figure 6.1 illustrates the full mobility management framework fulfilled on ADCus.

Cloud side: Taking advantage of the ADC framework, data is empowered to intelligently analyze network, location, and security properties independently through embedded scripts, rather than relying on third party services. This efficiently reduces the risk of users being cheated by compromised cloud services. Moreover, data mobility and live migration in the federate cloud environment can be actively detected by data per se rather than by third party auditors. Data can analyze the timely information related to its surrounding environment. Once a violation occurs, the data activates an inner probe to raise the alarm and disseminate the message to the bundled mobile terminal through the active binding framework.

Mobile terminal side: We assume that each user has a trusted mobile device, such as a smart phone or tablet that can compute and communicate with cloud services. We also assume that the user can physically possess the mobile device and no one can disrupt its privacy. Through the device, we can provide data usage monitoring, alert message receiving, full log information query, and a self-adapting network handoff scheme in mobile terminals to guarantee network service availability and ubiquitous informing when violations occur.

Security and privacy concern: This scheme can achieve a full transparency infrastructure on both mobile platforms and cloud platforms between clouds and users.

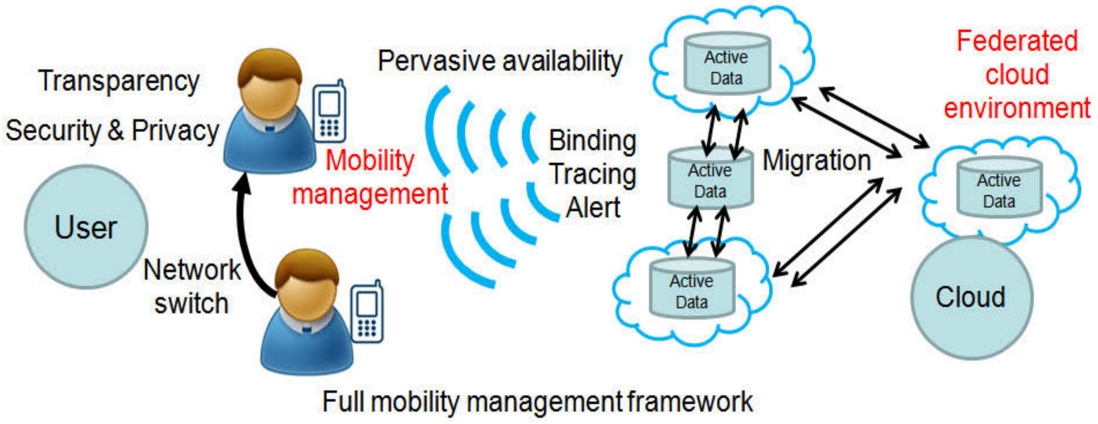


Figure 6.1 Full mobility management framework

6.2.3 Related Work

Related work has investigated data mobility aspects on geo-location. (Albeshri et al., 2012) proposed a new approach for geographic location assurance via a proof of storage protocol and a distance-bounding protocol. It allowed the user to verify the location of cloud data without relying on the word of CSPs. (Ries et al., 2011) introduced a way of verifying the location of virtual cloud resources and detecting geographical node movements by using network coordinate systems. (Benson et al., 2011) worked on proofs of retrievability and on provable data possession and data geo-location. Cloud users can verify that a cloud storage provider replicates the data in diverse geo-locations.

These efforts all relied strongly on the assumption of a linear relationship between network latency and distance in the realistic environment. Yet it is nearly impossible to ensure a linear relationship due to the complex network environment and other associated factors. Moreover, to retrieve information as to the origin of the data, third parties auditors are relied upon to check the data integrity and confidentiality. Our active data-centric framework utilizes an active binding framework between the active data and the mobile device to retrieve sensitive information and location. Through the external verification monitor, the detection of data movement can also be obtained immediately. The geo-location of the data relies on an inner network and location analysis module, which is more active and accurate.

Related work on the mobility management of mobile nodes can be found by different technologies. (Arakawa et al.) proposed an application-layer active wireless network switching mechanism on the android platform. It is a light-weight scheme with low computational overhead that assists wireless connection management on the existing android platform. However, this application relies on a fixed threshold, causing a Ping-Pong effect. (Hou and O'Brien, 2006) proposed a novel fuzzy-logic-based decision-making algorithm for a vertical handoff scheme. However, it can cause computational overheads: it may produce more accurate handoff time, but sacrifices energy cost and computational costs. The handoff scheme introducing hysteresis and a dwelling timer by (Pahlavan et al., 2000) could satisfy the mobility feature of mobile nodes, but static parameter setting might not function in various network environments, and dynamic parameter setting might reduce the matching ratio of the algorithm. Moreover, the hysteresis-based algorithm and dwelling-timer-based algorithm may display low accuracy when the mobile nodes move frequently in a complex HWNE combining multiple WIFI APs and cellular coverage (Liu et al., 2007). Our network switch decision relies on network change trends rather than a fixed threshold. We apply pre-switch and pre-join mechanisms, rather than switching or joining a new network once the context information reaches the configured threshold. We also propose a self-adapting handoff algorithm to dynamically adjust the sampling interval to improve the matching ratio.

6.2.4 Data Mobility Management

Data and users are geographically separated in cloud. Generally, this geographic isolation could affect users' ability to control their data when is stored and processed in the outsourced cloud. Without an appropriate transparency mechanism, users would never know what happens to their data and whether the current data location violates their SLA with the cloud.

6.2.4.1 Active Binding Framework

In such a scenario, users are interested in verifying the location of their data. The user sends a verification request to the cloud, then the cloud service triggers the data attached with the authorized parameters. The data analyses the request and then returns the requested information. As is well known, CSPs normally conceal the access location of users' data for security reasons. We cannot allow data to be accessible by external entities, only by the CSPs. Hence, the trustworthiness of CSPs is significantly important. Here are possible challenges that may occur in CSPs:

- Supposing the cloud service is compromised, the user's request may be directed to an incorrect service interface and a forged result may be produced by an adversary. How can we ensure that the user knows the result is fake and the subscribed cloud service is compromised?
- Cloud users are interested in verifying the location of their data. How to ensure that only the data owner, not anyone else, can verify the location?
- The data per se can actively analyze the network environment and geographic location of the data, without relying on third party services. How to ensure that the analysis result is correct and is sent to the user without error or violation.

Table 6.1 Notation used in the scheme description

Notation	Description
BID, UID, DID, RID	Binding ID, User ID, Data ID, Registration ID in Google cloud messaging framework
MK_{data}, MK_{user}	Master key for data, Master key for user
PrK_{user}, PrK_{data}	Private key for user, Private key for data
D_{index}	Data index in the cloud
H(M,Salt)	Hash function to the Message and Salt
UNT, UNA	User network type, User network address
DONA	Data original network address
DPNA	Data present network address
DGL	Data geographic location
Oper	The operations on the data

In order to construct a secure protocol for the proof of data location and data integrity, we establish an active binding framework between the user (mobile terminal) and the data (encapsulated by the ADC framework), based on a public key infrastructure. The user retains control over the data and the CSP ensures transparency of data usage. Embedded tracking and communication modules within the ADCu enable the user to be informed through a bound mobile terminal when any violation occurs.

In the following section we present the specification of the active binding framework established between users and data. Table 6.1 lists the notations used in the framework.

6.2.4.2 Configuration Phase

Since data is an active entity stored in cloud, we regard it as a functional entity as well as mobile terminals. Hence, an active binding relationship can be set up when a user subscribes to cloud services. When the user wants to verify data integrity or data location, the user is the *verifier*, and the data is the *prover*. When the data accomplishes integrity verification or location analysis, it sends the response to the user which is the *receiver*. In the stage of building the active binding relationship, the user and data are issued the following set of parameters:

$$User= <UID, MK_{data}, PrK_{user}, RID, Salt, UNT, UNA>$$

$$Data= <DID, MK_{user}, PrK_{data}, Salt, DONA, DPNA, DGL>$$

The *UID* and *DID* are described in the form of GUID (32-character hexadecimal string). *RID* is used to identify the mobile terminal that registers the message push service with the cloud infrastructure. *Salt* is utilized to increase the difficulty of cracking the digest message in the one-way hash function. The user and the data keep a consistent *Salt*. We assume that the data and the user can safely encapsulate the PrK_{user}, MK_{data}, PrK_{data}, MK_{user}. Only the paired data and user can successfully verify each other by the signature technology. Neither the cloud providers nor adversaries can reveal the verification message or forge a valid one to deceive the user or the data. *UNT* and *UNA* are the user's network type and address, which are used to describe the wireless environment of the user. *DONA* and *DPNA* are the data's original and present network address, which are represented by a set of <Web server address, VM address, VLAN address, Physical address> in a large cloud environment. *DGL* is the data's geographic location which is a sequence of physical location information. In cloud, the cloud provider stores a collection of data binding sets which are depicted as the following set:

$$Cloud\ binding\ set= <[BID, UID, DID], D_{index}, RID>$$

The *BID* is the binding ID to map the *UID* and the *DID*. In the binding set, the cloud does not own any useful information associated with the data and the user. Once the configuration is accomplished, the binding relationship between the user and the data is established.

6.2.4.3 Proof of Data Physical Location

- User→Cloud

The user sends the request to the cloud. The request parameters can be described as follows:

$$Request= <Sign\ (H\ (M,\ Salt),\ PrK_{user}),\ M>$$

$$M= <UID,\ RID,\ Salt,\ Oper>$$

The request message consists of User ID, Registration ID, Salt, and Operation on the data. The message is first calculated by a one-way hash function with the initialized Salt, and then the user signs the digest message with the user's private key. Finally, the user sends out request with the signature and message.

- Cloud→Data

The cloud receives the request from the user and then searches the data's accessible location in cloud via the binding set. Having obtained the data index, the cloud generates a triggering sequence utilized to access the data. Because the data is functional entity, it is capable of encryption and decryption while encapsulating corresponding cryptographic components. After the data is called by the cloud, it begins to decrypt the *Sign (H (M, Salt),* PrK_{user}*)* by the matching master key of the user. The *H (M, Salt)* is obtained, and then the data uses the same hash functions to calculate the *M* in the request by the data's *Salt*. By comparing the new digest result with the *H (M, Salt)*, we can verify that the request derives from the user. If the request passes the verification of the data, the data will analyze the ambient network environment and geographic location via the embedded modules, as discussed in detail in the next section.

- Data→Cloud→User

The data executes a similar procedure to sign the *H (Result, Salt)* with the private key of the data, and then responds to the user through the cloud services interfaces. The user verifies the *Result* via decrypting *Sign (H (Result, Salt),* PrK_{data}*)* by the MK_{data} and recalculating the hash message of *Result* and the user's *Salt*. If the outcome matches, it indicates that the result from the cloud providers is truly derived from the requested data, rather than a forged result by the cloud providers or adversaries.

In the overall procedure, if compromised cloud providers or adversaries maliciously tamper with the request or result, it is simply detected by the user or the data through signature verification. In normal circumstances, the cloud does not obtain any useful information associated with decryption of the signature. The user and data both enclose the corresponding key pair for inter-verification. Furthermore, it is extremely difficult for adversaries to crack this asymmetric cryptographic scheme (Pieprzyk et al., 2003). However, several exceptions may also occur:

If the request sent from the user does not receive any response from the data within a certain period of time, we regard the data as being in danger. There may be several reasons: the data is violated; the data is in the wrong location; the data is inaccessible, the data behaves incorrectly, and so on.

If the response cannot pass the verification of the user that means it may be intercepted by adversaries. The data is also in danger.

If the data cannot be triggered to an active state to periodically report its status, it may indicate that the runtime environment of the data is compromised or the network connection is unavailable.

6.2.4.4 Actively Analyze Network Location and Physical Location

The active data is capable of analyzing the network environment and geographic location by its embedded network and location analysis module. Compared to analysis by third party services, analyzing network and location information by data is more trustworthy. Third party services may behave dishonestly once they are compromised. The active data-centric framework prevents the data from disclosing any useful information related to the data even when it is under threat. In the data, the network and location analysis can only be triggered by a request with the user's identity. Certainly, the user's identity is protected by the digital signature. After verifying identity, the data sends the web service request to the location service provider to fetch the physical location of the current network environment. Through sending the web service request, the network topologic information of the request will be disclosed to the web service provider. With that information, the web service provider can generate a realistic physical location of the current data and send it back to the data. In our framework, the data requests and receives the location and network information through the network address trace. Therefore, it is a more realistic way to obtain the physical location and network environment of the data.

6.2.4.5 Pervasive Informing Framework

The pervasive informing framework ensures that any data violation against the SLA of the cloud is compulsorily informed by the ADCu to the bundled user. As is well known, this compulsory informing mechanism must employ push technology. Unlike pull technology, the request for a specific transaction by push technology (Wikipedia, 2013f) is initiated by the cloud servers or data per se in our context. Therefore, the push mechanism represents lower energy consumption and instant messaging capability on mobile devices. At present there are three mainstream mobile platforms: iOS, Android, and Windows mobile. They all provide standard a cloud messaging framework on a push basis. On the Android platform, for example, Google cloud messaging (GCM) is the official push framework, and developers are not required to implement push servers by themselves, but rely on Google cloud messaging servers. Two other well-known protocols, XMPP (Wikipedia, 2013j) and MQTT, can be used to customize push schemes on Android phones based on a third party's push server.

For simplicity, we employ the GCM framework to execute the message push scheme in our cloud platform. A detailed description of the architecture can be found in (Google, 2013a). After registration with the GCM service, the ADCu and user can establish a communication tunnel. When the ADCu is moved to a different location, this behavior can be triggered by the external supervisor. Through activation of the data by the supervisor, the data autonomously begins to analyze the geographic location and network information, and sends a signed message with the registration ID to its owner via the GCM service.

6.2.5 User Mobility Management

In the emerging cloud computing paradigm, users want to have ubiquitous access to their data and to receive alert messages instantly in cloud. This requirement places a demand on user mobility as well as data mobility. In the HWNE (mobile network (3G/4G), WIFI network, non-network coverage), smart phones or tablets can access the internet via these wireless technologies. However, considering the mobility of users, the user may move across the border of different wireless access points (APs).

To guarantee the pervasive data monitoring and tracing in such HWNEs, the smart mobile terminal should be capable of switching the network connection to maintain satisfactory quality of service (QoS) of monitoring in cloud. However, with the switch mechanism for improving availability, the ubiquity of cloud services may present some challenges:

By default, a smart phone operation system such as Android would connect to the last connected AP even if another AP presents a stronger radio signal or higher link speed.

By default, the Android system would maintain a WIFI connection even if the WIFI AP's signal is too weak to provide good QoS of the cloud services. In this case, a stable 3G/4G mobile network may be faster and more stable to provide good QoS.

By default, if both 3G/4G mobile network and WIFI are available, the Android system gives preference to the WIFI. In real circumstances, however, it is more effective to depend on the user's context such as location, movement, and cloud services, and making an optimized selection is also a challenge.

By default, if the Android system switches to a heterogeneous wireless network, it must disconnect the current network. This handoff mechanism likely results in loss of the data packet when the Android system continuously executes a service.

When the smart phone switches the current network connect, the traditional threshold handoff scheme easily causes a ping-pong effect. Ensuring a seamless switch also presents a challenge.

6.2.5.1 Wireless Network Handoff

Our network switch decision relies on network change trends rather than on a fixed threshold. We apply pre-switch and pre-join mechanisms rather than switching to or joining the new network once the context information reaches the configured threshold. We also propose a self-adapting handoff algorithm by calculating the movement trend of the mobile node. To improve the accuracy of the handoff, it is feasible to append some context information, such as location, condition, WIFI connection security requirement. However, there is a limitation in the current Android platform where the Android mobile phone OS allows only one network interface working on the board. That means that when we switch the network connection between a cellular network and a WIFI network, the Android OS must disconnect the current network interface in order to connect to the other one. In that circumstance, data communication may be lost. To guarantee continuous service, our solution is that once the network interface of mobile nodes changes, the internet service in the mobile node will reestablish the connection with the cloud service to notify the change in the client's address.

In the following section, we illustrate the user mobility management scheme in heterogeneous wireless networks through the self-adapting network handoff method. In the realistic scenario, the HWNE can be divided into three types of coverage:

Coverage 1: 3G/4G mobile network coverage zones (which cover most places around the country, but blind spaces without signal may exist)

Coverage 2: the overlay coverage zones of the mobile network and WIFI network. It can also appear as the following two sub-zones:

Coverage 2.1: the QoS of the WIFI network is higher than that of the mobile network

Coverage 2.2: the QoS of the mobile network is higher than that of the WIFI network

Coverage 3: there is no signal coverage.

6.2.5.2 Self-Adapting Handoff Algorithms

In terms of handoff features, we can basically define three handoff scenarios:

Scenario 1: the horizontal handoff whereby the mobile node switches to the better WIFI AP if the current AP cannot provide good QoS. This could occur when the mobile node moves within a building from one AP to another.

Scenario 2: the vertical handoff whereby the mobile node switches network connection in a heterogeneous wireless environment. This could occur when the mobile node moves from an outdoor cellular network coverage zone to a building providing WIFI access.

Scenario 3: the offline state when the mobile node stays in a signal blind zone.

To accurately calculate the movement trend of mobile nodes, we employ a dynamic sampling interval scheme in which the sampling interval is calculated according to the parabolic equation when the mobile node goes to the pre-trigger or pre-join stage:

$$\textit{Equation 1: } S=1-(|(L_{rssi} - N_{rssi})|^2)/M^2\text{ , } N=abs(S)*P$$

(S is the weight; L_{rssi} is the Rssi value in the last sampling; N_{rssi} is the current Rssi value; M is the preset variation value of Rssi, normally set as 10; abs is the function obtaining the absolute value' P is the preset sampling interval; N is the new sampling interval)

By adjusting the sampling interval during the pre-trigger or pre-join stage, the switch timing can be significantly improved. When the variation between the current sampling value and the last sampling value is high, the sampling interval automatically shortens to capture the new value. When the variation is steady, the sampling interval also remains stable. In the meantime, the movement trend of the mobile node can be simply observed through the recent sampling outcomes.

```
while(true) // do the sampling of RSS looply
 getRSS(current netID) //get the RSS value of the current connection
 if $RSS_n >= Th_p$
  then do nothing // the QoS is good enough to satisfy the network service
 else if $RSS_n < Th_p$
 then start the pre-trigger procedure
if $RSS_n > Th_t$
   Use the equation 1 to recalculate the new sampling interval
     if the absolute value of the $RSS$ of new AP - $RSS_n > RSS_n - Th_t$ && the movement
trend is far away the current AP and approaching the new AP //if the new AP offers a
much better QoS than the current AP.
     if it is the second recommendation to switch to the new WIFI AP
         then connect (new netID) //connect to the new WIFI AP
       else the first recommendation to switch
else if $RSS_n <= Th_t$
       if no available AP existed
    then disconnect(current netID) //disconnect the current WIFI AP
       handoff (3G/4G) //trigger the cellular network connection
       break; //breaking out the loop
else if the $RSS$ of new AP > $Th_t$ if the new AP can satisfy the network service
 then connect (new netID) // connect to the new WIFI AP
break; //let the service thread start to sample the new WIFI AP
Thread.sleep ($T_{si}$) // Let the service thread sleep $T_{si}$
```

Figure 6.2 Handoff algorithm for Scenario 1

The handoff trigger conditions of the self-adapting handoff algorithms can be described as follows:

Scenario 1: We assume that the mobile node currently connects to a WIFI AP and is moving within the coverage of multiple WIFI APs. We set the sampling interval to the *RSS* as T_{si}, the timestamp of the N_{th} sampling to the *RSS* is T_n, the timestamp of the $(N\text{-}1)_{th}$ sampling to the *RSS* is T_{n-1}, the *RSS* at T_n is RSS_n, the *RSS* at T_{n-1} is RSS_{n-1}, the pre-trigger threshold of the *RSS* is Th_p, the trigger threshold of the handoff is Th_t; when the RSS_n is between Th_t and Th_p, we smartly adjust the sampling interval to increase the accuracy of sampling. When the sampling RSS value is much larger than the preceding sampling result, we decrease the T_{si} in terms of the difference of change of the RSS. The algorithm is illustrated in Figure 6.2:

```
while(true) // do the searching WIFI APs loopy
 if the scan result of WIFI APs is null
   then do nothing
 else sort the scan result by the RSS of the WIFI APs
   if $RSS_{best} > Th_j$ // if the best RSS is better than the join threshold
     if the $Sig_{GPS}$ is lost && the AP has been configured in the security profile && the AP has better frequency than the cellular network && the movement trend is approaching the new AP //that means the mobile node is moving into an indoor environment, and connecting to this AP can receive better QoS than the cellular network
       if it is the second recommendation to switch to the WIFI AP
         then connect to the AP which has best RSS in the scan result and keep signal surveillance by using the algorithm from Scenario 1
       else the first recommendation to switch
     else continue
   else if $RSS_{best}$ is between $Th_{pj}$ and $Th_j$
 then start the pre-join procedure
 Use the equation 1 to recalculate the new sampling interval
   else if $RSS_{best} < Th_{pj}$
     then do nothing
 Thread.sleep($T_{si}$) // let the service thread sleep $T_{si}$
```

Figure 6.3 Handoff algorithm for Scenario 2

Scenario 2: We assume that the mobile node currently connects to a cellular network either 3G or 4G and it is moving from a public outdoor environment without WIFI coverage to a zone covered with WIFI connection. We assume that the WIFI connection does not require a security authentication. The security profile was stored in the mobile phone at the most recent connection. The cellular network maintains a constant sampling frequency. We set the GPS signal to Sig_{GPS}, the sampling interval to the *RSS* of WIFI APs is T_{si}, the pre-join threshold is Th_{pj}, the join threshold is Th_j, and the AP which has the best RSS signal is RSS_{best}. The algorithm is illustrated in Figure 6.3.

Scenario 3: Disconnection is an inherent property of mobile computing since we cannot guarantee a continuous connection with internet service for an entire session. The key technology to manage such disruption and achieve consistent operation is to relocate relevant data through embedded database storage which is SQLite in the Android system and cache processing. Once network coverage appears, the application will immediately switch to an online pattern to resume the operation.

6.2.6 Implementing Data and User Mobility Management

Our experimental environment has been set up as follows: the heterogeneous networks involve Vodafone Australia's HSDPA mobile network, multiple Wi-Fi (802.11g) APs (all configured in the mobile terminal). On the client side, we established a mobility service on the Android device (SAMSUMG Galaxy SII) which installs Android 4.1.2 OS to process user mobility management and mobile portal application on the phone to display the user's cloud data location and violation message. As a test, we experiment only the mobile push service on the Android platform. Discussion of the differences between the push services on iOS and Windows Mobile platforms is beyond the scope of this work. Hence, we employ the GCM framework as the push service. The experiments are based on following assumptions: we assume that the runtime environment (JVM) of the active data behaves correctly and the data is not violated, and we assume that the data will be activated when it moves to another cloud host. We also assume that a safe location service provider (LSP) is available on the Internet. When the user subscribes to the cloud service, the active bundle relationship will be established. Only the bundled user can receive the message from the data and verify its originality. Generally, in our framework, since the data is active, it normally takes charge of disseminating the message. Before sending out the message from the data, the message is protected by the signature technology. The GCM server then pushes the signed message to its bundled user. In normal circumstances, the signed message cannot be forged by third party tools as they do not own the corresponding key in order to sign the forged message. In other words, even though an adversary might intercept the message and modify the content to send back to the user, as in a man-in-middle attack, the user would detect it when verifying the message with the user's stored key. As the data is active and the mobile device possesses the keys locally, when the data is activated by violation attacks, the bundled user will receive an immediate notification message from the data. The specific experimental outcome is illustrated in Section 7.1 in detail.

6.3 Addressing Data Replica Challenges in Cloud

Although the ADC framework proposed in the previous chapters imparts data with self-defending and self-protecting capability in a secure container, physical failures and errors (e.g. power failure, hardware failure, network failure and maintenance of data servers) do exist, as discussed by (Bonvin et al., 2009, Pinheiro et al., 2007), and need to be considered in the overall framework. When such uncontrollable accidents occur, data replicas can be used to enable not only data recovery but also server recovery in cloud (Squicciarini et al., 2013). A data replication scheme can be considered in cloud to support large-scale parallel read/query operations through deploying multiple data replicas and distributing them across different cloud hosts. However, data replication in cloud entails the following challenges:

6.3.1 Issues Statement and Challenges

- The degree of data replicas:

Data replication can improve access efficiency and availability in distributed storage systems with diverse storage capability and network status. But excessive replication would increase the complexity of data update consistency and cause storage space overconsumption. Hence, flexible adjustment schemes for data replication, such as creating new replicas when a data hotspot appears and removing redundant replicas when the hotspot cools down, or allowing request traffic fluctuation to avoid unnecessary replica adjustments, are beneficial in terms of data consistency and efficient storage.

- The distribution of data replicas:

The distribution of data replicas directly impacts replica storage, query, and update costs. In a highly dynamic cloud system, data may be distributed to different cloud hosts to achieve better query load and higher efficiency. A traditional cloud storage system such as Amazon's Dynamo (DeCandia et al., 2007) statically distributes replicas at a fixed number of physical nodes. Others do not consider geographical diversity, access cost, and replication cost issues (Qu and Xiong, 2012). These types of distribution may result in high access costs and unbalanced workload.

- The consistency of data replicas:

Data replica management always faces data consistency issues when data is updated or deleted. Normally two types of update strategies are used to maintain data consistency (Xu et al., 2004): strong consistency (updating all replicas promptly) and eventual consistency (allowing a certain amount of diversity in a limited time). Strong consistency provides access to up-to-date data but at high operational cost, and reduces replica availability. Eventual consistency is achieved with low operational costs, but data content may be obsolete at times when users attempt to access their data. Hence it is necessary to consider consistency, availability, and system performance in any data replication scheme.

To address these challenges, we propose an adaptive data replication management scheme that adjusts the data replication process based on dynamic-window analysis (referring to the analysis of the collected average value and standard deviation in a dynamic set of array) with respect to the response time of requesting data resources. The proposed scheme can achieve better workload balance and efficiency than the request-oriented method (referring to the prompt execution of data replication management according to request traffic) and the random scheme (referring to the fixed and static data replication schemes). Notably, the proposed adaptive data replication mechanism operates on the ADC framework which encapsulates active scripts on sensitive data to achieve self-protection and self-defense in an independent data unit. Each active data unit (ADCu) contains a collection of data files with similar features and properties.

Furthermore, this packaged data structure demonstrates better performance in data replication management than traditional data files due to the compression feature. Furthermore, we propose an adaptive data update consistency scheme that is more flexible than strong consistency and eventual consistency in fulfilling diverse update scenario requirements.

6.3.2 Related Work

Data replication technology has been widely investigated in various contexts, such as grid systems (Tatebe et al., 2002, Chervenak et al., 2002, Lee and Weissman, 2001), distributed file systems (Ghemawat et al., 2003, Weil et al., 2006), and recent data clouds (Bonvin et al., 2010, Qu and Xiong, 2012, Wei et al., 2010b, Bonvin et al., 2009). In grid systems, Tatebe et al. (2002) and Chervenak et al. (2002) proposed static replication strategies in which the number of replicas and the placement of replicas are static at the start of the life cycle. No further replicas are created or adjusted. Lee and Weissman (2001) proposed a dynamic replication technique that adapts to changes in user request pattern, storage capability, and bandwidth. The dynamic strategy allows more flexible adjustment and placement of data replicas depending on the varied context of the grid environment. However, these researches are based on grids. Data storage services are more user-oriented in cloud data storage systems and hence higher mobility and immediacy of data replicas may be required. In distributed file systems, the Google file system (Ghemawat et al., 2003) and Ceph file systems (Weil et al., 2006) maintain only a fixed number of data replicas and apply static distribution mechanisms, which do not satisfy the requirements of new cloud data storage. In data clouds, Wei et al. (2010b) proposed a dynamic distributed cloud data replication algorithm based on the Hadoop Distributed File System. They discussed dynamic determination of the number of replicas according to availability requirements and data replica placement according to changing workload and node capacity. Bonvin et al. (2010) proposed a dynamic data replication scheme on a self-managed key-value store. Using the data replicas' popularity and their client locations, the scheme dynamically adapts to a varied query load by analyzing the most cost-efficient locations for data replicas.

This paper differs from the above related work in several respects. First, this work addresses data replica management based on the active data-centric framework in cloud. Compared to the conventional data storage structure, our framework encapsulates a collection of data in an independent container (ADCu) for active protection. This data structure supports simultaneous data updates as long as the update field does not conflict. Moreover, the ADCu can communicate with third party entities via its inner communication module. Hence, replica update consistency can be implemented through a mutual push or pull mechanism. There is no related work that discusses a data replication scheme in the data-centric framework.

Secondly, we discuss an adaptive data replica management scheme that dynamically maintains the amount of data replicas, increases the number of replicas when a hotspot develops, allows fluctuations of request traffic, and removes redundant replicas when the hotspot cools down. Furthermore, the scheme supports dynamic distribution of replicas across cloud virtual servers.

Thirdly, update consistency is flexible and satisfies various data access scenarios in cloud by deploying the adaptive data update consistency scheme.

6.3.3 Data Replica Network Establishment

In our work, replicas are divided into secondary replicas, primary replicas, and source replicas in terms of update sequence. Secondary replicas are used to improve access performance. Primary replicas are used to avoid a single point of failure. Source replicas are normally used for archiving data. An update is first processed in the primary replicas, and last propagated to the source replicas. New secondary replicas are created from their connected primary replicas. The update consistency scheme is enforced from primary replicas to secondary replicas, or else secondary replicas pull updated data from their adjacent primary replicas.

Initially, a user subscribes to a CSP and stores data in the CSP. A new supervisor instance will be initialized and registered with the ADNI registry to manage data replicas. Each user's data is classified by its data types such as image files, text files, designated data files, and so on. The user data is then partitioned into several portions. Each portion can be regarded as an ADCu. For fault tolerance, each portion is designed to have at least three replicas: a source replica, a primary replica, and a secondary replica. Figure 6.4 shows the data replica network topology diagram for one portion of user data. The metadata for a data replica is illustrated in Figure 6.5.

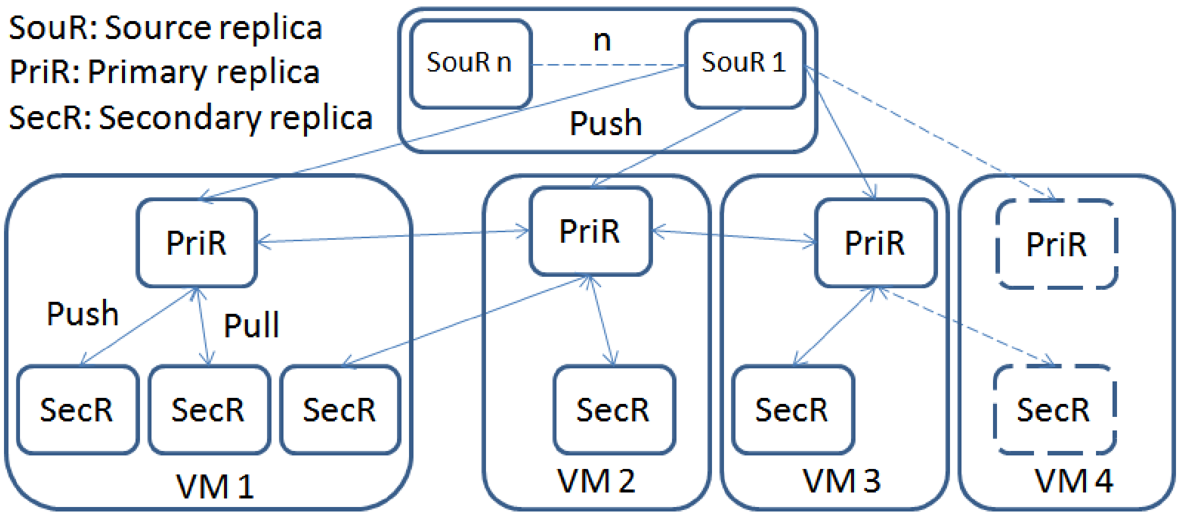


Figure 6.4 Data replica network topology

```
 <Replica>
<ID>
  <PID>fdasfEASD324ssadfs</PID>  <!--Portion ID-->
  <PType>images</Ptype><!—Portion type-->
  <RID>WFHsdfhakfdh23423s</RID>  <!--Replicas ID-->
</ID>
   <Version>
     <LT>24124324234234</LT>  <!--Latest timestamp-->
     <VID>23124322</VID>  <!--Latest version ID-->
     <CT>Primary</CT>  <!--Copy type-->
   <Checksum>fdksafkajf3234sdfdskjs4334adjfiejoawfsladfAJELJLAJS3232</Check
Sum>
   </Version>
 <Provenance>
  <Location>192.238.232.12:8080/Cloud/d/s2s4hfsakdfhsd</Location>
  <CreatorID>3243122238</CreatorID>
  <TimeStamp>2342354333222353</TimeStamp>
 </Provenance>
 <Supervisor>
  <SID>sdfsadfasdf234sSFD</SID>  <!--Supervisor ID-->
  <SL>192.238.232.12:8080/Cloud/d</SL>  <!--Supervisor Location-->
 </Supervisor>
 </Replica>
```

Figure 6.5 Metadata of a data replica

6.3.4 Replica Adjustment and Distribution

The rules for deciding to adjust replicas are as follows:

Definitions: Two types of observation windows (referring to an array with historic value in a period) are used for analyzing data hotspots (referring to how often the data is accessed): a short-term time window (T_s), and a long-term time window (T_l) (that meets $T_s < T_l$). The value of T_s requires that the system makes a promptly response to a sharp and steady increase of access traffic. T_l allows for temporary fluctuations of access traffic on active data to make a cautious replica adjustment. To analyze the growth trend and the hotspot value on a portion of users' data, we calculate the average value $W_{average}$ and the standard deviation value W_{sd} of the current window, as illustrated by the following formulas:

$$W_{average} = \frac{1}{n} * \sum_{i=1}^{n} T_i \qquad W_{sd} = \sqrt{\sum_{i=1}^{n} (T_i - W_{average})^2}$$

Based on the response time T_{resp} on accessing one portion of user's data, we now discuss the replica adjustment process. Figure 6.6 shows the decision tree for adjusting replica numbers. Each portion of users' data is regarded as an independent and active data unit. We can improve the new response time $T`_{resp}$ by creating new replicas and distributing them to light-load virtual hosts. To achieve high replica utilization rate, high access efficiency, and low operational cost, the adjustment algorithm increases replicas when the access traffic presents an obvious growth trend and decreases replicas when the hotspot demonstrates a real cool-down. Through adjusting the observation windows and adding dwelling-timers, the timing of adjustment replicas can be significantly improved.

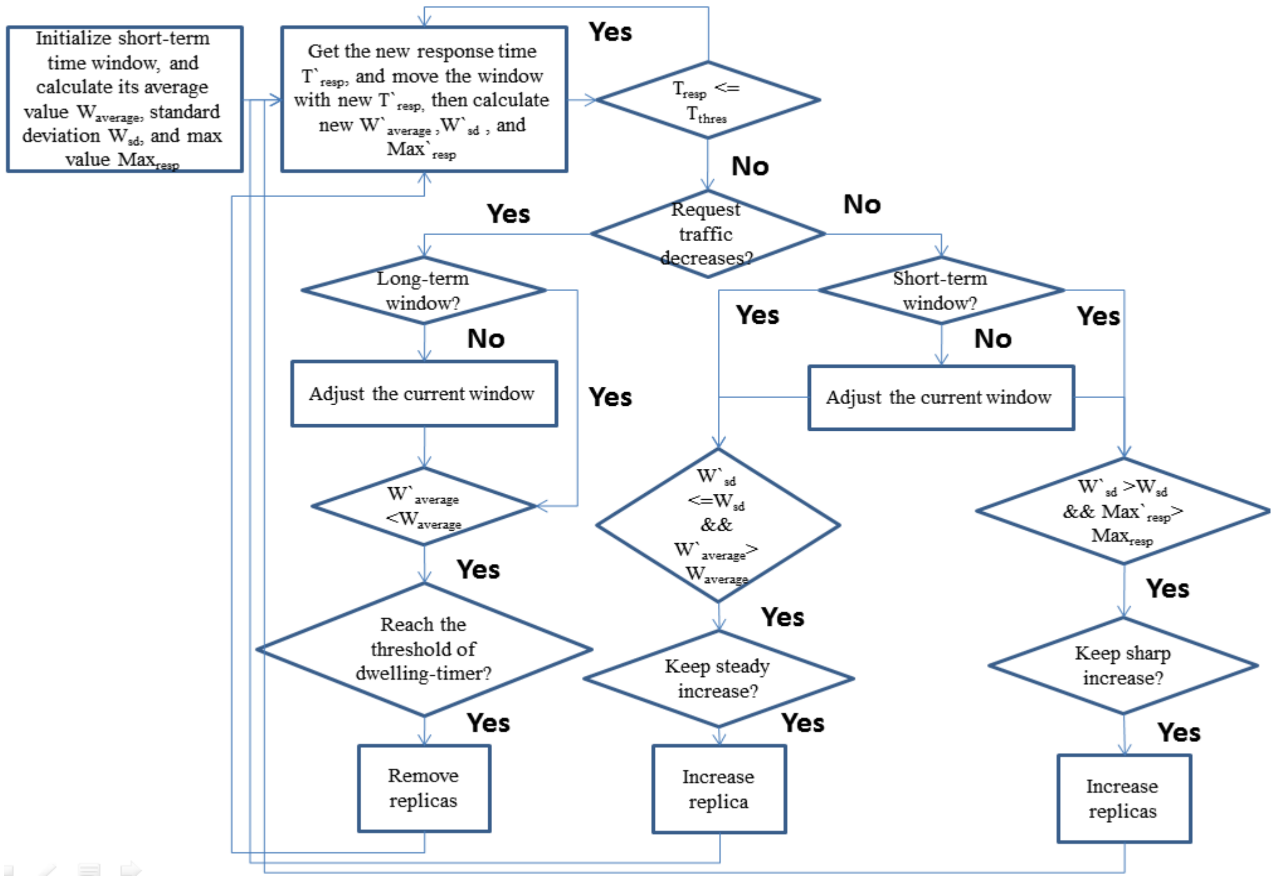
Once a new replica is created, the replica network is updated. As each server can simultaneously process a limited number of requests due to the capability constraint (Wei et al., 2010b), each new replica is distributed to a relatively light-load virtual host to balance the workload and obtain better response performance. The dynamic workload of each virtual cloud server can be obtained from cloud management interfaces. We create a workload set = [L_1, L_2,

L_3 ... L_i] (L_i represents the workload of the server i at the timestamp t_i), where L_i can be calculated by the equation:

$$L_i(t_i) = \alpha * T_{resp} + \beta * P_{broadband} + \gamma * P_{storage} + \delta * P_{cpu}$$

where T_{resp}, $P_{broadband}$, $P_{storage}$, P_{cpu} are the response cost, broadband usage, storage usage, CPU usage, respectively. $P_{broadband}$, $P_{storage}$, P_{cpu} are expressed as the percentage of the full capability, and $\alpha, \beta, \gamma, \delta$ represent weights of T_{resp}, $P_{broadband}$, $P_{storage}$, P_{cpu} respectively. We then rank the workload set from light to heavy order.

We set the servers in the first third of the above workload set as light-load servers. When a new replica is generated, the supervisor randomly chooses a light-load server to store the new replica. The replica then connects to the supervisor and its primary replicas logically.



Initialize short-term time window, and calculate its average value $W_{average}$, standard deviation W_{sd}, and max value Max_{resp}
Get the new response time $T`_{resp}$, and move the window with new $T`_{resp}$, then calculate new $W`_{average}$, $W`_{sd}$, and $Max`_{resp}$
Yes
$T_{resp} <= T_{thres}$
No
Request traffic decreases?
Yes
No
Long-term window?
No
Adjust the current window
Yes
Short-term window?
Yes
No
Yes
Adjust the current window
$W`_{average} < W_{average}$
Yes
$W`_{sd} <= W_{sd}$ && $W`_{average} > W_{average}$
$W`_{sd} > W_{sd}$ && $Max`_{resp} > Max_{resp}$
Yes
Reach the threshold of dwelling-timer?
Yes
Keep steady increase?
Yes
Keep sharp increase?
Yes
Remove replicas
Increase replica
Increase replicas

Figure 6.6 Decision tree for adjusting replica numbers

6.3.5 Reconfiguration of Replica Network

In the proposed data replica network, primary replicas initially deal with update operations and secondary replicas deal only with query operations. Initially, there is just one primary replica and one secondary replica. When update operations increase, secondary replicas can switch to primary replicas to improve update performance. If the query task grows rapidly, primary replicas join to assist the query process. If one of the primary replicas breaks down, the supervisor selects one of the secondary replicas to replace it. The ADCu encapsulates a pile of data with similar properties and features. Because the ADCu employs object-oriented technologies, it supports simultaneous update of different data content inside the ADCu through multi-threading manipulation. However, if the ADCu receives an update request on data content that is just being processed, the program lock is activated to avoid update conflict. The supervisor records the data update along with the version. Synchronization work is carried out in terms of update versions.

6.3.6 Adaptive Data Replica Consistency Mechanism

Cloud data consistency addresses the multiple replica synchronization issue after the data content update. The strong/immediate consistency aims to ensure that all replicas are immediately updated, and each access fetches the latest data. However, this strategy involves high operational cost that can cause performance decline. Eventual consistency allows some inconsistencies in replicas within a limited time period. Hence, better access performance can be obtained with low operational cost. However, the access may receive obsolete data. In the cloud storage environment, the requirement for data consistency varies. It is not flexible to carry out a fixed consistency strategy. For this reason, we propose an adaptive data replica consistency mechanism that dynamically adjusts consistency strategies to achieve balance in availability, consistency, and operational performance.

The detailed description of the update consistency mechanism is as follows:

We first check whether the ADCu is processing an update operation; if so, we then check whether the new data requesting update is the data currently undergoing the update operation within the ADCu. If so, we suspend the update until the program lock is released from the ADCu.

The ADCu supports simultaneous update operations as long as the update fields do not conflict. The first updated replica is the primary replica, and then we utilize the following algorithm to broadcast the update to other replicas.

We choose the update frequency (UF) and access frequency (AF) of each ADCu as the characteristic variables for selecting consistency strategies. We calculate UF $= \frac{C_U}{T}$ (referring to the number of update C_U in T period of time, where T can be dynamically selected based on real cases), AF $= \frac{C_A}{T}$ (referring to the number of accesses of C_A in T period of time). We calculate the consistency requirement (CR) by the following formula, where α is the impact factor for the update frequency and β is the impact factor for the access frequency. The value of CR is between 0 and 1 (1 indicates strong consistency; 0 indicates eventual consistency).

$$\mathbf{CR = \alpha * UF + \beta * AF}$$

We set the minimum threshold for UF as Min_{uf} and the maximum threshold as Max_{uf}, and the minimum threshold for AF as Min_{af} and the maximum threshold as Max_{af}. If UF reaches Max_{uf}, and AF reaches Min_{af}, the supervisor employs the eventual consistency strategy due to the high update frequency and low access frequency. Applying strong/immediate consistency would cause a sharp increase of operational cost. The source replica only synchronizes all primary replicas. When a secondary replica receives an access request, it fetches the latest data from its connected primary replica by the pull mechanism. If UF reaches Min_{uf} and AF reaches Max_{af}, the supervisor executes the strong consistency strategy. A low update frequency does not result in high operational cost while executing synchronization from the primary replica to all secondary replicas by the push mechanism and each access on the ADCu can fetch the latest data.

From the above two cases, the value of α and β can be determined. When the update frequency and the access frequency are between the maximum value and the minimum value, we can determine CF and hence the degree of consistency. As the threshold values for update frequency and access frequency vary dynamically, α and β also vary accordingly. The update consistency strategy varies depending on various set-threshold values. If the update frequency and the access frequency on one ADCu exceed the defined thresholds, the supervisor sets their value to the defined thresholds (Max_{uf}, Min_{uf}, Max_{af}, Min_{af}).

6.4 Simulated Performance Result and Analysis of Adaptive Data Replica Management

We conducted data replication management tests in the simulated cloud scenario by CloudSim, which is the simulator for modeling and simulation of cloud computing infrastructures and services. In this case, we configured two datacenters geographically distributed in cloud. Each host in the datacenter was configured with sufficient RAM, storage, bandwidth, and processing capability to serve a certain number of queries in each epoch. Each virtual machine (VM) was configured as 512MB RAM, 1000 mips (Millions Instructions Per Second), 1000M bandwidth, and single processer. The host allocates query requests to each VM by the VMSchedulerTimeShared policy.

6.4.1 Data Request and Query Test

We simulated the cloudlet (which is the modeling for cloud services and tasks) as the data request task. We regarded the VMs as the active data replicas: executing one data request task on active data replicas can be considered as equivalent to executing a cloudlet task in the specific VM. To observe the workload balance and performance of the adaptive replica management scheme, we compared it with the random scheme and the request-oriented scheme. Figure 6.7 shows the request traffic diagram in terms of epochs. Throughout the epochs, the generated query tasks can be divided into five stages: initialization stage (5%), steady increase stage (20%), steep increase stage (15%), fluctuation stage (40%), and fall after rise stage (20%). Due to the simulated environment, the response time and replication cost had no specific unit. However, through comparison of these three schemes, we illustrate the benefit of our adaptive data replication management mechanism. Figures 6.8 and 6.9 show the response time and the replication cost for the generated request traffic. Initially, 15 replicas are generated with a random scheme. We set the response time threshold for starting to analyze the request traffic trend as 100. There are only 2 replicas initially for the request-oriented and the adaptive schemes. At each epoch, we submit corresponding request traffic to the cloud broker. In the initialization stage, the response time and the replication cost of the request-oriented and the adaptive schemes are steady and at a low level due to the system data initialization and configuration procedure.

The replication cost remains constant for the request-oriented and the adaptive schemes until the response time reaches 100. Hence, the response time increases steadily in terms of the request traffic growth. In the steep increase stage, the replication cost of the adaptive scheme is similar to that of the request-oriented scheme, with a slightly shorter response time than the request-oriented scheme. This illustrates that the response performance of the proposed adaptive method is similar to that of the request-oriented method when the request traffic exhibits a steep increase, and the response time increases in a more stable manner. In the stage of fluctuation, the request-oriented scheme exhibits a similar fluctuation trend to the request traffic. But the adaptive scheme demonstrates a more stable state and shows a slight increase to maintain a rapid response through the longer observation window analysis when the fluctuation occurs. When the request traffic decreases, the request time of request-oriented scheme decreases as well. But the adaptive scheme analyses the overall traffic trend and stabilizes the replication cost at 18. Hence, the request-oriented scheme provides a more stable response time than the adaptive scheme due to the removal of redundant replicas in the request-oriented scheme. However, the adaptive scheme can handle fluctuation situation better and avoid unnecessary removal of replicas.

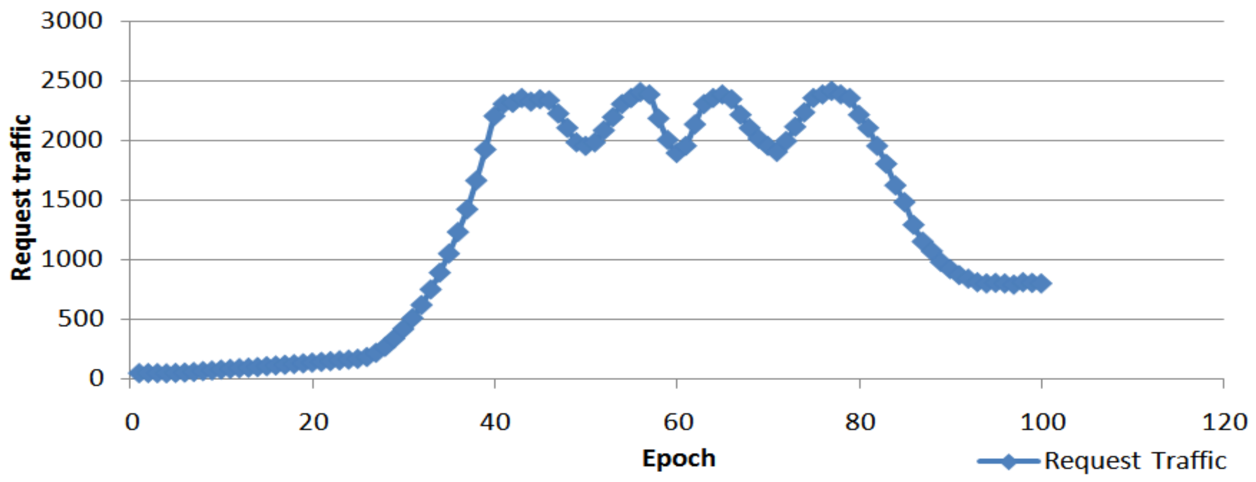


Figure 6.7 Request traffic

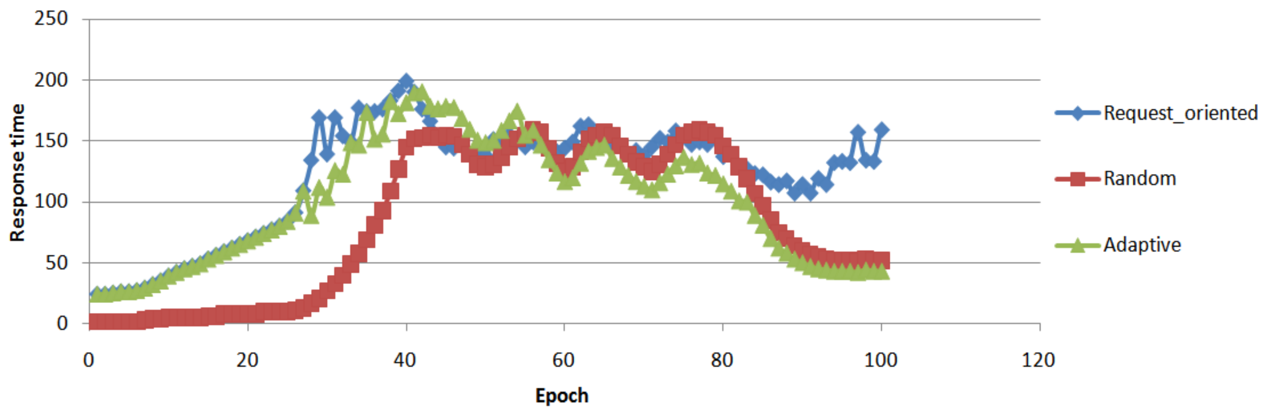


Figure 6.8 Response time with request traffic

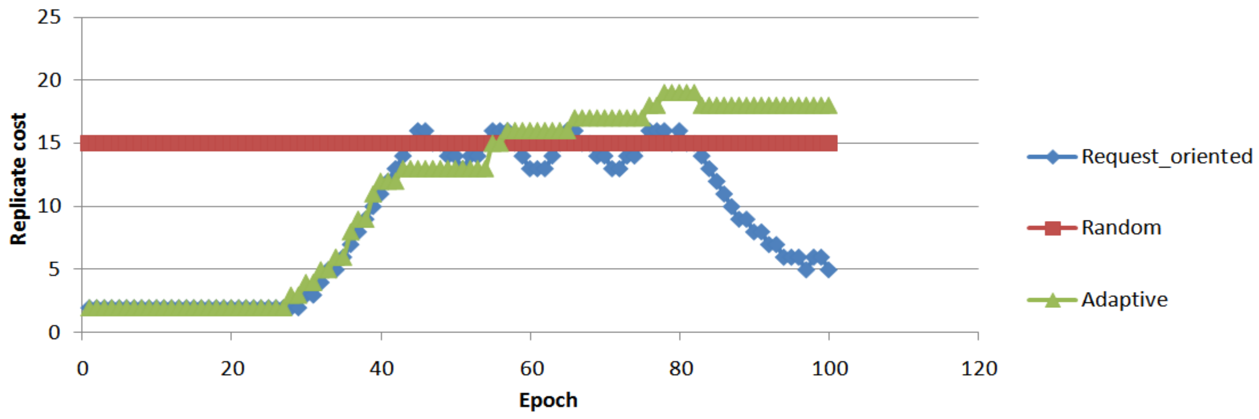


Figure 6.9 Replication cost with request traffic

We assume that creation of a replica takes 200 operations, and removal of a replica takes 100 operations. The maximum processing capability for each VM is 200 query requests per epoch. We calculate the operational cost by accumulating all replica operation costs in a certain period of time, including creation and removal of replicas. We calculate the utilization rate through current processing requests / maximum processing capability. The result illustrates that the random scheme has the lowest average utilization rate (47%), and the request-oriented scheme has the highest rate (73%) because it promptly adjusts replicas in terms of changes in traffic. The adaptive method (60%) is slightly lower average utilization rate than the request-oriented scheme, but its overall operation cost (3400 for creation, 100 for removal) is much lower than that of the request-oriented scheme (5200 for creation, 2200 for removal). The simulation results show that, in the dynamic cloud environment, our method has good performance and efficiency to adapt to fluctuating traffic, and can provide more flexible adjustment of replicas with low operation cost, which usually occur in cloud.

6.4.2 Data Update Consistency Test

We define the update operational cost (OC) as the overall operational cost, and each update costs 100. In the initial replica network, we set up 1 source replica and 3 primary replicas. Each primary replica manages 3 secondary replicas. The minimum threshold for update frequency (UF) is 5, and the maximum threshold is 50. The minimum threshold for access frequency (AF) is 20, and the maximum threshold is 100. For the first 9 epochs, the UF remains at 5, and we gradually increase the AF by 10. From the second 9 epochs, we increase the UF by 5 and continue the AF increase by 10 from 20. Hence, this set of test data covers various consistency scenarios. Figure 6.10 illustrates the OC for the three consistency strategies. As the epochs continue, the OC of the strong consistency scheme continues to grow. The adaptive consistency scheme presents a fluctuating increase and a lower OC than the strong consistency scheme. At the 64th, 73rd, and 82nd epochs there are huge gaps in the OC between the adaptive consistency and the strong consistency schemes. Figure 6.11 shows the percentage of accessing the latest update in the three consistency strategies. The strong consistency can guarantee 100 % of accessing the latest update since it lays emphasis on updating all replicas immediately. The adaptive consistency displays fluctuating variation between strong consistency and eventual consistency. Due to the features of eventual consistency, the OC and percentage of accessing the latest update in this strategy remain constant. The experimental results show that the adaptive consistency scheme can be more flexible to satisfy more data update scenarios and achieve better performance balance.

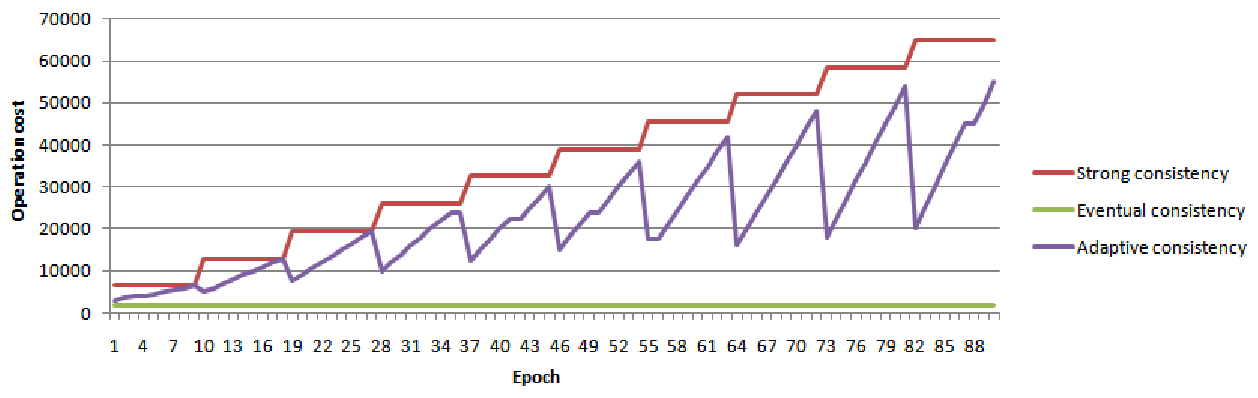


Figure 6.10 Operation cost in three consistency strategies

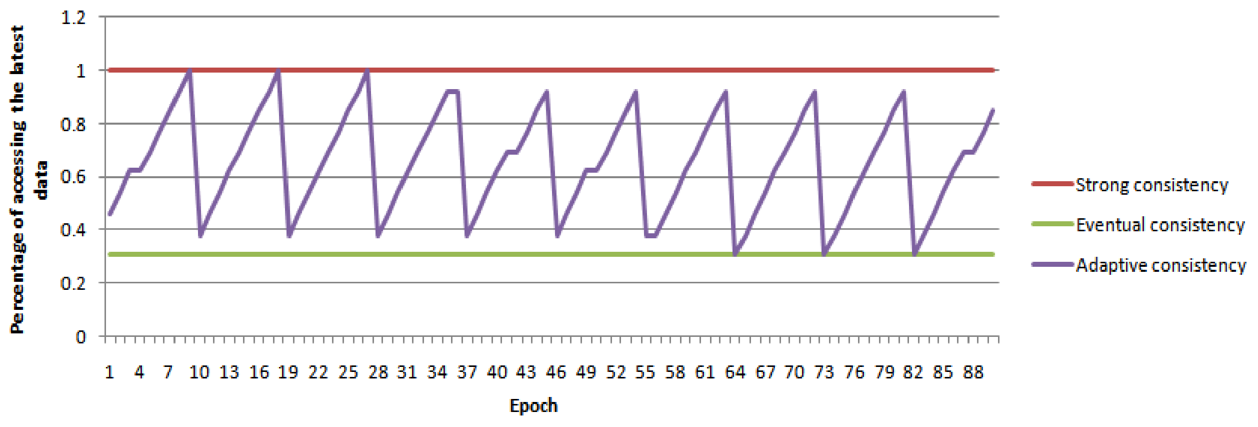


Figure 6.11 Percentage of accessing the latest update in three consistency strategies

6.5 Summary

In this chapter, we have presented two important data operation and management mechanisms. One is for data full mobility management that efficiently addresses the issues of accountability and mobility of data and users. Through deploying the active binding framework between users and their data, CSPs can provide trustworthy transparency of data usage and data security warranty for data owners. Any data violation against SLAs would be compulsorily informed to its bundled owner by ADCus. The second mechanism controls data replica management based on ADCus. The proposed adaptive replica management scheme has good performance and efficiency to adapt to fluctuating request traffic and can provide more flexible adjustment of replicas at low operational cost. Moreover, the adaptive consistency scheme is more flexible to meet dynamic data update requirements with diverse traffic.

Chapter 7 Experiments and Evaluations

In this chapter, we evaluate the proposed architectures, protocols and applications using experiments on the IBM HS20 blades center in our laboratory. The relevant data storage analysis of security, privacy and control in the outsourced private cloud environment is discussed.

The experimental hardware infrastructure is built on the computing facilities in the ARN laboratory of iNext research center in the University of Technology Sydney. The main computing platform is based on the IBM HS20 blades center which comprises 8 blade servers, each with Intel Xeon processor 2.8 GHZ, 2 x 2GB and an internal 36.4GB hard drive. The storage node is built on the IBM TotalStorage DS400 (SAN server) disk subsystem with 4 disk arrays, and each disk has 146.8GB space and 10K cache. We have one fiber-channel switch, one Ethernet switch, and one management switch used to establish the network topology of all blade servers and storage server, as shown in Figure 7.1. As well, we deploy a gateway server to connect to an external public network, and we have an external client PC used to manage the blade center and private cloud platform. For the software configuration, we adopt the open source cloud toolkit OpenStack (OpenStack, 2011) to establish and manage the mini-private cloud platform. Notably, we do not focus on the construction of the cloud platform or on resource scheduling in cloud; rather our emphasis is on establishing and testing a trust-oriented data protection framework adapted to a private cloud environment. From perspective of cloud infrastructure, private cloud infrastructure essentially has no salient difference from public cloud infrastructure. They mainly deliver different services that orient to different clients. Private cloud prones to provide services operated solely for a single organization and public cloud services are rendered over a network that is open for public use. However, security consideration may be substantially different for services (applications, storage, and other resources) that are made available by a service provider for a public audience and when communication is effected over a non-trusted network (Wikipedia). Our research in this stage only focuses on data security mechanisms a mini-private cloud environment. Adapting our active data security model to public cloud platform will be one of important future works.

To improve the scalability and performance of cloud services, we can horizontally extend and deploy more compute and storage nodes to host and schedule VMs to achieve load balance. Hence, for simplicity, we merely create one VM with full computing capability to deploy each required service on a blade server, such as CPRBAC service, AAS, active data management and creation service, GCM service for push messaging to bundled users' mobile devices, and web server for cloud portal. The VMs stores in the configured SAN server are used to simulate a shared storage cloud environment. Each VM installs the Ubuntu 12.04 server version OS. To enable the ADC framework to work, each OS is required to install standard Java 1.62 SDK. We leverage the RMI-SSL protocol to guarantee secure communication between each service.

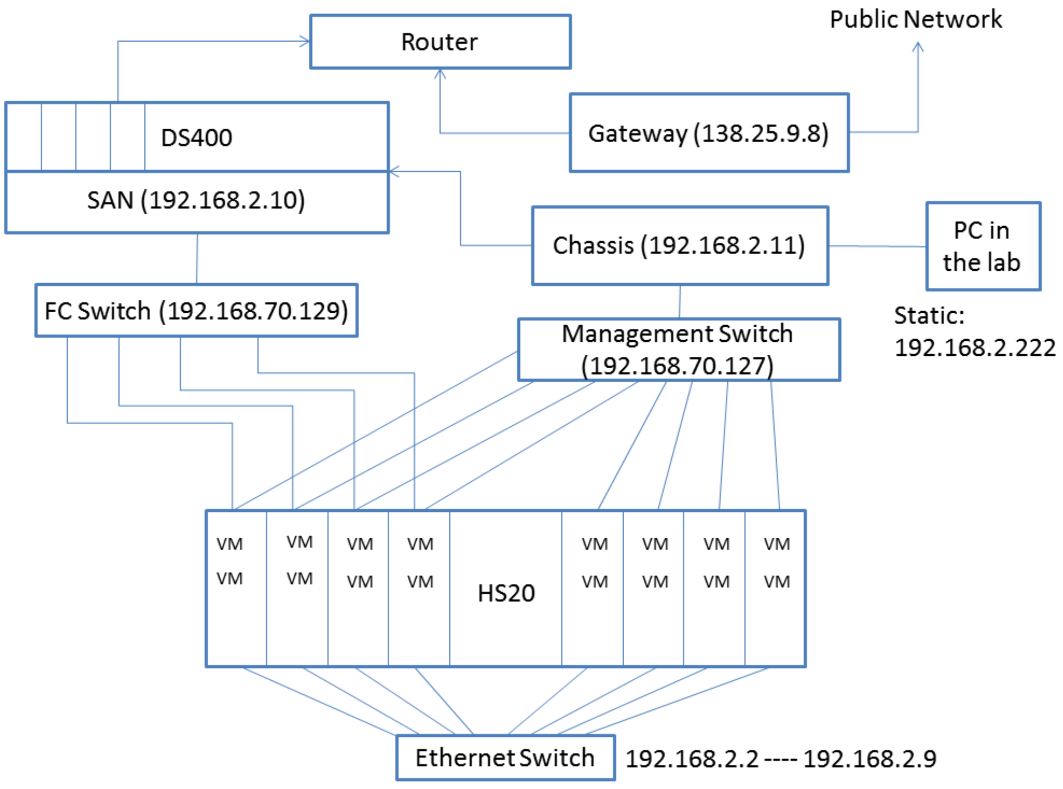


Figure 7.1 Network topology of a mini-private cloud in ARN lab

The structure of this chapter is as follows: Section 7.1 discusses data security evaluation and analysis of the proposed frameworks based on a semi-honest adversary model by a number of assumptions. Section 7.2 conducts general functionality and security tests in terms of the features presented in Chapters 4, 5, and 6 by intrusion attacks, CPRBAC policy conformance test, and full mobility tests including data location violation and wireless network handoff. Section 7.3

mainly conducts performance and cost experiments related to the ADC framework and AAS on the established IBM HS20 blade cloud platform. Section 7.4 is the comparison of different data protection mechanisms applied in cloud as discussed in Chapter 2. Finally, we summarize this chapter.

7.1 Security Evaluation and Analysis

The cloud computing model presents risks and issues related to security and privacy aspects similar to those in traditional computing paradigms such as grid computing, SOA, and utility computing. The risks may emanate from internal or external environments. Internal risks include eavesdropping of data, traffic or trend analysis, data manipulation, altering of data integrity, theft of information, and unauthorized disclosure or operation of data; external risks may consist of diverse attacks such as distributed denial-of-service (DDoS) attacks, malicious intrusion attacks, man-in-the-middle attacks, and insertion of a malicious code or program. Other issues may involve communication and network security in the internet/intranet/extranet environment (L. Krutz and Dean Vines, 2010).

In traditional computing paradigms, users normally possess and operate data in their local environment. Users naturally take responsibility for their data's safety. In cloud environments, however, new challenges arise. Once data is in the hands of a CSP, questions of "who, what, where, when, how and why" related to users' data must be addressed since the cloud paradigm shifts users' responsibility from user-side to cloud-side. These new issues require new measures to provide sufficient transparency and accountability of users' data usage to mitigate users' concern and buy their confidence in using cloud-based data services in sensitive fields such as banking or healthcare.

The ADC framework is the core component corresponding to data in the proposed trust-oriented data protection scheme. We explain the scheme using the metaphor of an ADCu as a new type of land mine. The mine is equipped with a trigger, a probe, a novel wireless communication device (like RMI-SSL), and explosive material (like sensitive data). In normal circumstances, attackers who are not familiar with the access structure of this mine could easily trigger an explosion (like raising an alarm in the ADCu) if they attempt to break into the mine.

Only a sapper who is an authorized entity can open the mine. Consider an attacker who pretends to be an authorized entity but has an invalid access token (access tokens are normally issued if the request has been verified in the CPRBAC and AAS services in our case). In that case the mine would trigger an explosion by virtue of the procedure whereby the mine first verifies the identity of requester through the ZK proof scheme. If the proof cannot be verified, the mine triggers an explosion. Subsequently, the mine would check that any verified token generated conforms to the result from the access control policy and is consistent with transaction requirements. The mine can only be revealed when the entity can pass through verification in these two layers. The ADCu is designed for such a scenario. We aim to enhance security preservation even when data itself is under direct attack. The self-protection capability requires the control or intelligent functions to be embedded inside data to achieve self-describing, alarming, and defending regardless of the surrounding environment. Only an authorized entity that passes the verification in a consistent transaction session in the AAS module can obtain information from inside the ADCu via its registered supervisor. The ADCu supports detection of violations of familiar internal operations such as update, delete, and read-only. Additionally, the supervisor supports detecting violations of external operations such as move, duplicate, remove the entire ADCu on the OS level. In the following, we evaluate the security property of our framework in detail from an attacker's perspective.

Attacker: We assume that the attacker has bypassed the access control layer and gained direct contact with users' data through elevation of privilege. These data are encapsulated in an independent ADCu. Normally, the attacker attempts to reveal the content of the ADCu. However, the ADCu requires a set of corresponding request parameters as keys to trigger the data when a third party service executes operations on it. Since the internal content of the ADCu is invisible for external requesters, it is difficult for the attacker to obtain the construction of the keys. Even if the attacker is familiar with the structure of the keys, the verification module embedded in the ADCu first requires a three-round ZK proof procedure to identify requesters. In principle, only the corresponding supervisor of the ADCu has the keys to bypass the proof procedure. Additionally, even if the attacker is able to make the supervisor do his/her bidding, a verified token must still be presented to prove the validity of the request. This token is theoretically impossible to duplicate as it is issued uniquely and non-reversibly in the legislation stage.

Furthermore, the verified token must be based on a complete transaction session in the AAS module indicating that the current request has been verified and is consistent with the defined CPRBAC policy or the procedures in the update-type operation. Nevertheless, if the *probe* detects a violation, the ADCu will raise the alarm for invalid access.

The attacker may attempt to disassemble the data through reverse engineering or decompilation attacks by virtue of the fact that the ADCu is implemented by JAR technology. Once the ADCu is disassembled, the compiled script files and encrypted data are exposed to the attacker. Through decompiling the script files, the original source code may be extracted. It is highly dangerous if an adversary injects some new code to violate the active functions within the ADCu after obtaining the whole security protocol from the original source code. But this is hypothetical. We have employed code obfuscation technology which greatly increases the difficulty of decompiling the ADCu. Moreover, even if the ADCu is disassembled and the data block is revealed, it is in encrypted form that is still unreadable by the attacker.

Now, we assume that the attacker has decompiled the ADCu and obtained the source code within the ADCu. Then the attacker may attempt to tamper with the scripts logic to conduct a jailbreak operation to bypass the verification and identification procedure and obtain access to the resource in the core. We can simply prevent this tampering through inserting a set of checkpoints throughout the whole ADCu. Any inconsistent outcome created at the checkpoints would be actively detected when the ADCu is running and then cause its termination. Furthermore, JAR technology supports sealing and signature technology which can guarantee the integrity of the ADCu.

We next assume that the attacker attempts use side-channel attacks to violate the runtime environment, which is JRE in our case, to achieve the aim of extracting the content through control of the ADCu, because the active features of the ADCu rely highly on the runtime environment. If the attacker controls the ADCu, it will misbehave. However, due to obfuscated scripted and strong AES encryption of data blocks, although the attacker may have control of the ADCu, nothing can be retrieved unless the attacker possesses the valid cryptographic key. Moreover, we can periodically inspect the validity of the JRE through integrity examination.

For network and communication security, we have deployed the RMI-SSL protocol which guarantees that both entities use a mutually trusted communication tunnel to communicate over the network. Moreover, the cloud data storage system stores users' data in an unordered and invisible form. No sensitive information (such as data index or location information) is revealed. It is extremely difficult for adversaries to eavesdrop on information targeted to a specific user. However, an attacker can still carry out an attack to compromise the whole data storage host. That can happen, just as natural disasters can occur, unpredictably. The only way to reduce the impact of these attacks is to create a multiple-replica mechanism and distribute the replicas to geographically separate hosts.

In summary, the proposed framework demonstrates the compulsory enforcement of verification and identification when data is triggered. This framework enables logging, verification, monitoring, and data operations to be actively enforced along with data per se. To some extent, cloud data usage demonstrates comprehensive transparency to the data owner. Once data is compromised or violated in a subscribed cloud service, data can detect the violation immediately and inform customers by bundled mobile phone using push services. Furthermore, data location changes can be detected by the ADCu and disseminated to bundled users. Any violation breaking the relevant law is compulsorily recorded as evidence that may be delivered to relevant governance, regulation, and compliance organizations. In the following section, we conduct major functionality and security tests to verify the frameworks proposed in this thesis.

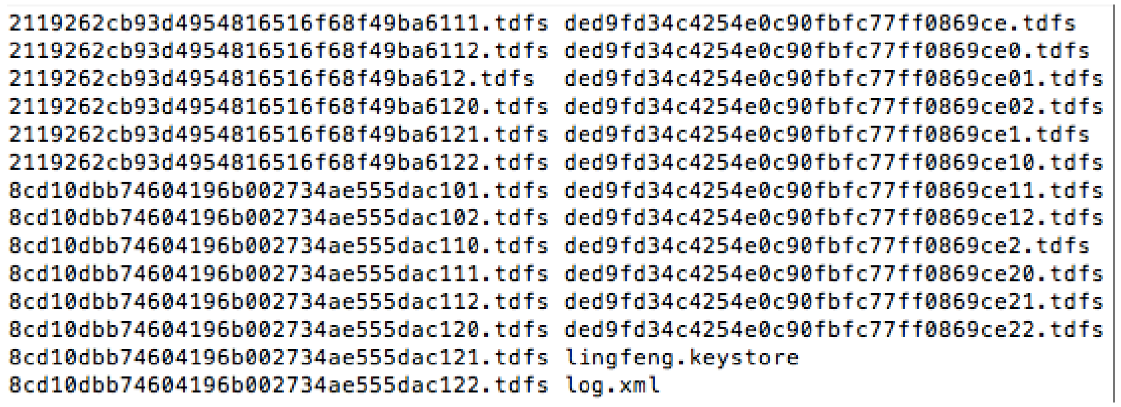
7.2 Functionality and Security Tests

In this section, we perform major functionality and security tests on the proposed functions, protocols, and mechanisms of the trust-oriented data protection framework. First, we conduct a number of intrusion attacks and normal data retrieval operations on created ADCu data. Then we conduct a conformance test to verifying CPRBAC policy structure and evaluation results. Finally, data location variation alert test and wireless handoff test are performed to verify the feasibility and mobility of data security management.

These tests are based on several assumptions: we assume that data owners do not release any sensitive information to unauthorized parties, including the secret keys used to generate signature and encrypt data and personal privacy which can be utilized to crack a user's account; we assume that the supervisor instance, CPRBAC service, and AAS behave correctly; and we assume that an adversary has penetrated the data security control layer and obtained access to the data storage layer. The adversary attempts to carry out direct attacks on the targeted ADCu; we assume that the runtime environment of the ADCu behaves correctly; we assume that the data is activated when it moves to another cloud virtual host; we assume that a safe LSP is available on the Internet.

7.2.1 Test Cases for Active Data-centric Framework

As the core of trust-oriented data protection framework, the ADC framework undertakes the major functionality of protecting data from intrusion, tampering, and violation. To test its functionality and security, we programmably create a set of active data cubes stored in a storage node. Each ADCu is named by a GUID and its replicas are named after it with an additional bit of digital number. In this case, each ADCu contains 12 image files using JAR technology. At the same time, we deploy the supervisor instance in the same VM domain as those ADCus. All context and index information are stored in the ADNI registry when the ADCus are created. Figure 7.2 illustrates the created demonstration ADCus in one of the storage nodes.



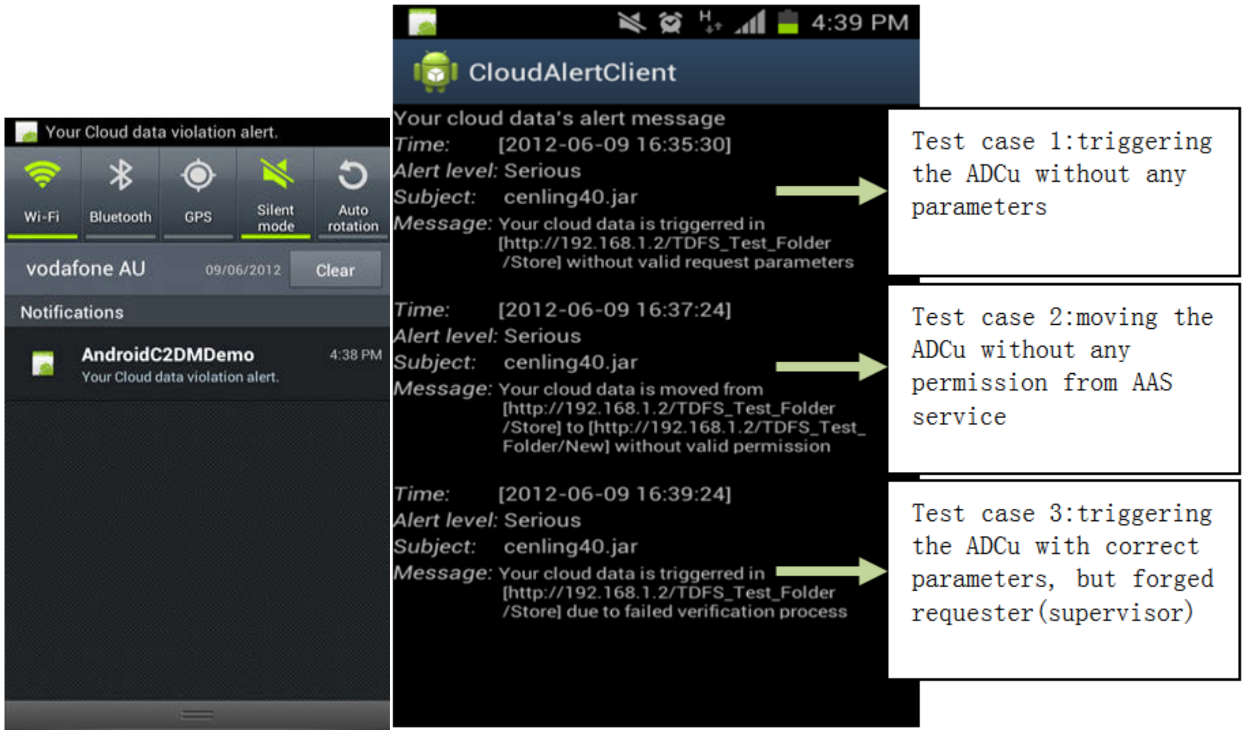
```
2119262cb93d4954816516f68f49ba6111.tdfs ded9fd34c4254e0c90fbfc77ff0869ce.tdfs
2119262cb93d4954816516f68f49ba6112.tdfs ded9fd34c4254e0c90fbfc77ff0869ce0.tdfs
2119262cb93d4954816516f68f49ba612.tdfs  ded9fd34c4254e0c90fbfc77ff0869ce01.tdfs
2119262cb93d4954816516f68f49ba6120.tdfs ded9fd34c4254e0c90fbfc77ff0869ce02.tdfs
2119262cb93d4954816516f68f49ba6121.tdfs ded9fd34c4254e0c90fbfc77ff0869ce1.tdfs
2119262cb93d4954816516f68f49ba6122.tdfs ded9fd34c4254e0c90fbfc77ff0869ce10.tdfs
8cd10dbb74604196b002734ae555dac101.tdfs ded9fd34c4254e0c90fbfc77ff0869ce11.tdfs
8cd10dbb74604196b002734ae555dac102.tdfs ded9fd34c4254e0c90fbfc77ff0869ce12.tdfs
8cd10dbb74604196b002734ae555dac110.tdfs ded9fd34c4254e0c90fbfc77ff0869ce2.tdfs
8cd10dbb74604196b002734ae555dac111.tdfs ded9fd34c4254e0c90fbfc77ff0869ce20.tdfs
8cd10dbb74604196b002734ae555dac112.tdfs ded9fd34c4254e0c90fbfc77ff0869ce21.tdfs
8cd10dbb74604196b002734ae555dac120.tdfs ded9fd34c4254e0c90fbfc77ff0869ce22.tdfs
8cd10dbb74604196b002734ae555dac121.tdfs lingfeng.keystore
8cd10dbb74604196b002734ae555dac122.tdfs log.xml
```

Figure 7.2 Demonstration ADCus created in the storage node

To prove the efficiency and feasibility of defense against intrusion attacks, and of executing external update-type operations, we present four test cases. To test the alert functions when a violation occurs we have developed a third party program to simulate simple violation scenarios.

To implement the violation alert function we have developed a notification feature using the GCM framework in our cloud platform. It actively binds cloud user's data violation events with mobile devices. Test case 1 triggers the targeted ADCu without any parameters. Test case 2 moves the targeted ADCu from one location to another location without permission from the AAS. These two requests are executed by the corresponding supervisor. Test case 3 triggers the targeted ADCu file with correct parameters but a forged supervisor instance. To verify correctness, we conduct Test case 4, which triggers the targeted ADCu with the corresponding supervisor and correct parameters. To observe the execution procedure while the ADCu is active, we have configured four checkpoints in the scripts.

All three invalid test cases triggered the probe inside the ADCu because they failed at the verification and identification stage. Violation messages were generated by the ADCu and sent to the supervisor, and at the same time, the bundled mobile device received an alert message immediately. Figure 7.3 shows screenshots of the notification messages in Samsung Galaxy SII when the violations occurred in the user's ADCu.



Your cloud data's alert message
Time: [2012-06-09 16:35:30]
Alert level: Serious
Subject: cenling40.jar
Message: Your cloud data is triggerred in [http://192.168.1.2/TDFS_Test_Folder /Store] without valid request parameters

Time: [2012-06-09 16:37:24]
Alert level: Serious
Subject: cenling40.jar
Message: Your cloud data is moved from [http://192.168.1.2/TDFS_Test_Folder /Store] to [http://192.168.1.2/TDFS_Test_ Folder/New] without valid permission

Time: [2012-06-09 16:39:24]
Alert level: Serious
Subject: cenling40.jar
Message: Your cloud data is triggerred in [http://192.168.1.2/TDFS_Test_Folder /Store] due to failed verification process

Test case 1:triggering the ADCu without any parameters

Test case 2:moving the ADCu without any permission from AAS service

Test case 3:triggering the ADCu with correct parameters, but forged requester(supervisor)

Figure 7.3 Screenshot of notification messages when violations occurred in user's data

The valid request in Test case 4 evoked a successful response and the mobile device did not receive any alert messages from the ADCu. Figure 7.4 shows the log information generated when the configured checkpoints were executed.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE log SYSTEM "logger.dtd">
<log>
<record>
  <date>2013-07-23T14:59:03</date>
  <millis>1374555543651</millis>
  <sequence>0</sequence>
  <logger>8cd10dbb74604196b002743ae555dac110.tdfs</logger>
  <level>INFO</level>
  <class>arn.inext.Shell_Main</class>
  <method>main</method>
  <thread>1</thread>
  <message>the TDFS is triggered</message>
</record>
<record>
  <date>2013-07-23T14:59:03</date>
  <millis>1374555543750</millis>
  <sequence>1</sequence>
  <logger>8cd10dbb74604196b002743ae555dac110.tdfs</logger>
  <level>INFO</level>
  <class>arn.inext.identification.Verifier</class>
  <method>calculateResult</method>
  <thread>1</thread>
  <message>the verification procedure passed</message>
</record>
<record>
  <date>2013-07-23T14:59:03</date>
  <millis>1374555543791</millis>
  <sequence>2</sequence>
  <logger>8cd10dbb74604196b002743ae555dac110.tdfs</logger>
  <level>INFO</level>
  <class>arn.inext.ProcessRequest</class>
  <method>readOperation</method>
  <thread>1</thread>
  <message>the request has been delegated to core</message>
</record>
<record>
  <date>2013-07-23T14:59:03</date>
  <millis>1374555543854</millis>
  <sequence>3</sequence>
  <logger>8cd10dbb74604196b002743ae555dac110.tdfs</logger>
  <level>INFO</level>
  <class>arn.inext.Core_Main</class>
  <method>main</method>
  <thread>1</thread>
  <message>the data has been loaded</message>
</record>
</log>
```

Figure 7.4 Log information generated by the checkpoints in the ADCu

7.2.2 Test Cases for Verifying CPRBAC Policy

In this test case, we assume that the CPRBAC service and AAS service are trustworthy, because our focus is on constructing a fine-grained authorization policy structure and integrating distributed context components, rather than on the security issues and protection schemes related to these components. However, to verify the policy structure and evaluation result, we conduct a conformance test to determine whether the system meets the specific design requirement of the access control policy. The following tests are conducted on the deployed CPRBAC service by XACML specification on a specific computing node. Table 7.1 illustrates the conformance test cases and results for the CPRBAC service. We create a policy repository which contains the set of correct permission assignments before we conduct the conformance test. Test cases 1 to 10 pertain to the permission assignment, and 11 to 13 pertain to role delegation and roaming. In the first 10 test cases, alternating access requests conform to the policy and violate the policy. A permitted access request is issued with an access token created by the SHA-1 hash calculation of a set of parameters defined in previous section. A denied access request returns a deny response.

Table 7.1 Conformance test on CPRBAC service

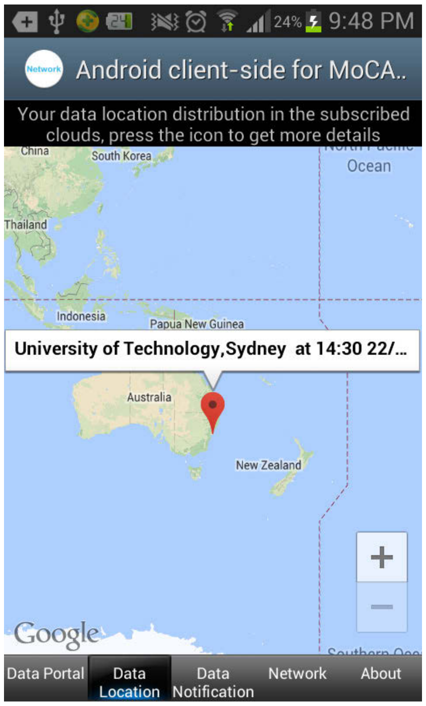
Test case	Policy description	Access request	Access response
1	Cenling40 can access and write the EHR data of lele9	<cenling40,write, ~/lele9/EHR,null,null,null>	<Permit, U5tBVEYfRfDktZS 7WhBKcW46y/0= >
2	Cenling40 can access and write the EHR data of lele9	<cenling50,write, ~/lele9/EHR,null,null,null>	<Deny>
3	Any subject can read any resource with “non-sensitive” tag	<cenling40,read,~/*/#non-sensitive,null,null,null>	<Permit, 93I8hA3kK32puvp+ GOy7+gbLxjg= >

4	Any subject can read any resource with "non-sensitive" tag	<cenling40,read,~/*,null,null,null>	<Deny>
5	The physician can access the EHR data of lele9 if the physician is the designated one	<physician,access,~/lele9/EHR,designated physician,null,null>	<Permit, M//uc/5k0r3JFUnnvWRCNAnKyn4= >
6	The physician can access the EHR data of lele9 if the physician is the designated one	<physician,access,~/lele9/EHR,null,null,null>	<Deny>
7	The physician can access the EHR data of patients when it is in emergency	<physician,access,~/*/EHR,null,emergency,null>	<Permit, 1zSyH9CGOuuTlvuKUQBedB3QXyo= >
8	The physician can access the EHR data of patients when it is in emergency	<physician,access,~/*/EHR,null,null,null>	<Deny>
9	The physician is only allowed to write the EHR data of patients during 9:00AM-5:00PM local time, and meanwhile sending email to the patient	<physician,write,~/*/EHR,14:00,null,sending email>	<Permit, jkcBESoZsfMO19ZT+ihF0xcqiyQ= >
10	The physician is only allowed to write the EHR data of patients during	<physician,write,~/*/EHR,18:00,null,sending email>	<Deny>

	9:00AM-5:00PM local time, and meanwhile sending email to the patient		
11	General physicians can get temporary permission of senior physicians if they are busy and approve it	[general physicians,senior physicians,busy&approval]	<Permit, /sbbM0XEGpGO+MUO5eqLaloR8jM= >
12	Senior physicians can get all permissions that general physicians own	[senior physicians,general physicians,null]	<Permit, /4LYaKS2+Q6Oy1iXCDWIgco7PsI= >
13	A research student in University of Technology Sydney can be assigned a temporary role of physician in the Health center	[(research student,University of Technology Sydney,(physician,Health center),null)]	<Permit, j9PAkJIrMPjY3vsUe5WS295eJ6U= >

7.2.3 Test Cases for Full Mobility Management

In this experiment, we attempt to simulate an invalid movement of a created ADCu from our lab in the University of Technology Sydney to a public cloud environment in Amazon EC2 cloud without permission. When a user subscribes to a cloud service, an active bundle relationship is established. Only the bundled user can receive the message from the data and verify its originality. Generally, in our framework, since data is active, it normally takes charge of disseminating the message. Before a message is sent out from the data, it is protected by the signature technology. The GCM server then pushes the signed message to its bundled user. In normal circumstances, a signed message cannot be forged by third party tools as they lack the corresponding key to sign the forged message. In other words, even if they intercept the message and modify its contents to send back to the user, as in a man-in-middle attack, the user would detect it through verifying the message with the user's stored key. As the data is active and the mobile device possesses the keys locally, when data is activated by a violation attack, the bundled user instantly receives a notification message from the data. In this test, we move the ADCu from University of Technology Sydney to the Amazon EC2 cloud. When the ADCu arrives at one of the EC2 servers, the deployed supervisor detects this OS-level change and then activates the ADCu to report its current physical location. The verification module inside the ADCu first identifies the request. A valid request is delegated to the core of the data to carry out the network environment and physical location query by calling the third party LSP. A web service request from the data is processed by the LSP. The LSP responds with correct network information and physical location to the data by interpreting the client-side web request. The real and accurate network environment and physical location information is pushed to the bundle mobile node as the data movement notification according to the network topology information of the original request from the ADCu. Figure 7.5 shows two screenshots (on the left is the old physical location; on the right is the latest physical location) that were received when we executed this movement operation of the ADCu.


24% 9:48 PM
Android client-side for MoCA..
Your data location distribution in the subscribed clouds, press the icon to get more details
China
South Korea
Ocean
Thailand
Indonesia
Papua New Guinea
University of Technology,Sydney at 14:30 22/...
Australia
New Zealand
Google
Data Portal
Data Location
Data Notification
Network
About

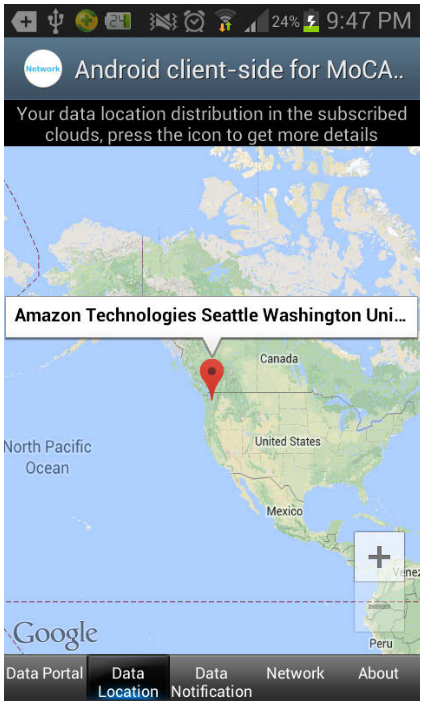

24% 9:47 PM
Android client-side for MoCA..
Your data location distribution in the subscribed clouds, press the icon to get more details
Amazon Technologies Seattle Washington Uni...
Canada
North Pacific Ocean
United States
Mexico
Peru
Google
Data Portal
Data Location
Data Notification
Network
About

Figure 7.5 Data location view when user's active data moves from the University of Technology, Sydney cloud server (left figure) to the Amazon Technologies Seattle Washington cloud server (right figure)

Next we provide test results regarding user mobility management. To test the ubiquity of the service, we developed a mobility service running in the background of the Android OS to evaluate the active network signal value based on timeline. To capture and measure the behavior of the handoff process, we utilized the GCM framework to continuously push messages from the active data cube to the mobile node. By collecting the handoff execution delay, handoff delay, packet loss, and one-way delay, we obtained representative values and performance results. We used the NTP (NTP, 2013) protocol to calculate the one-way delay of message dissemination from the active data to the mobile node in case of time inconsistency on both sides. In Scenario 1, we set up two WIFI APs (with both APs configured in the HTC Desire phone) along the walking path. The distance between APs was around 50m. The initial state of the mobile node was under the coverage of AP1 and retained good QoS of the network. The mobile node walked from AP1 to AP2 at a steady pace. In Scenario 2, the initial state of the mobile node was under the Vodafone Australia HSDPA mobile network and the mobile node started walking towards the APs. We set the pre-trigger, pre-join, trigger, and join thresholds as -85dbm,-110dbm,-110dbm,-85dbm respectively. The regular sampling interval was 1500ms.

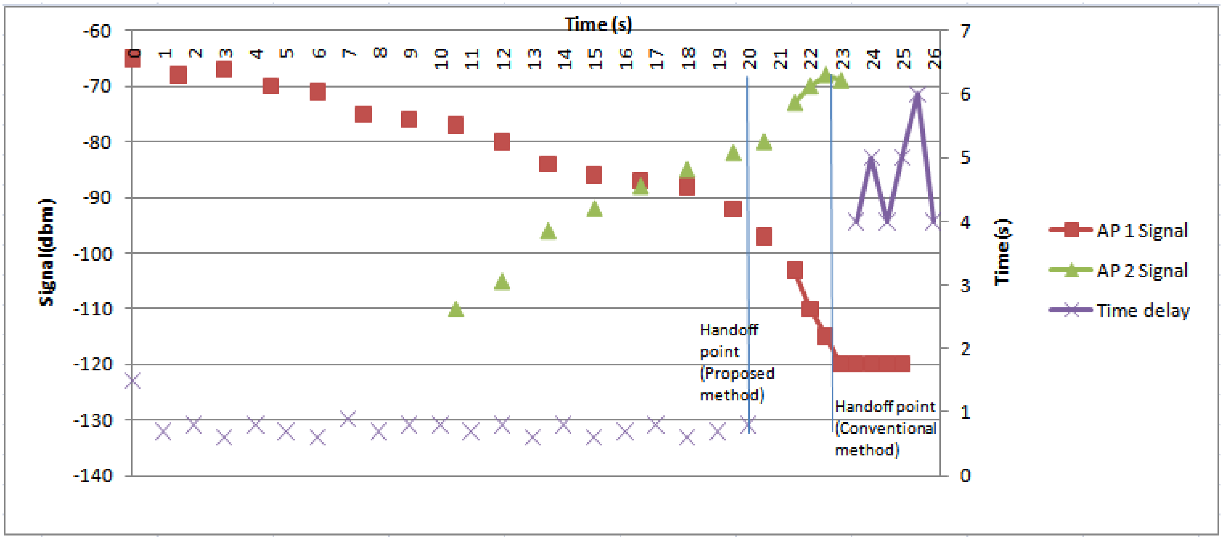


Figure 7.6 Signal of APs and one-way delay diagram in Scenario 1

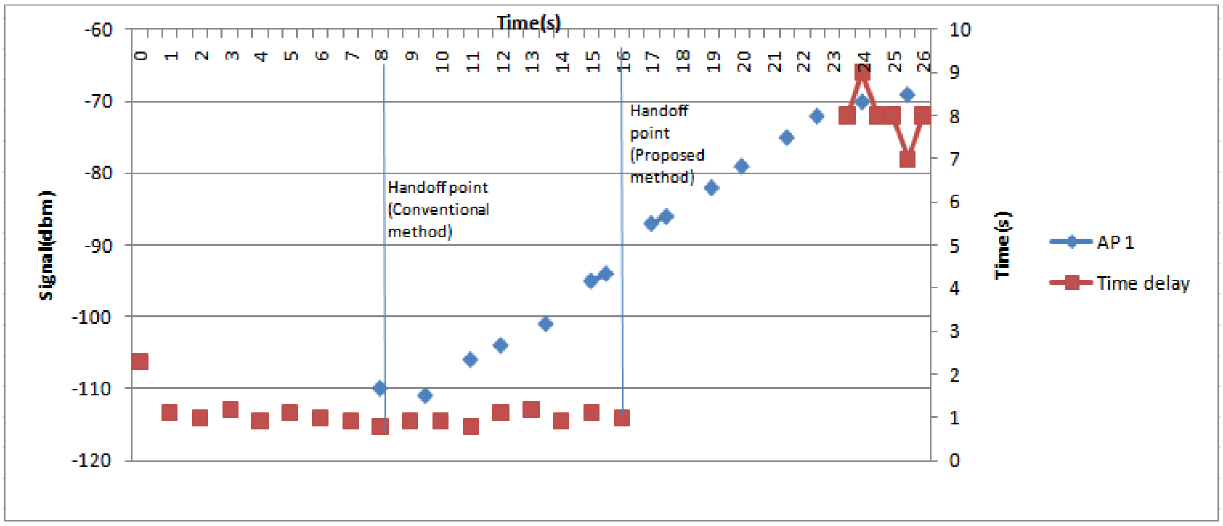


Figure 7.7 Signal of AP and one-way delay diagram in Scenario 2

Figure 7.6 shows the test result of Scenario 1. The one-way delay of the message from the active data to the mobile node is stable before the first handoff occurs at the 20s. The elapsed time can be under 1s except for the first second due to run-up of the service. During the first 20s, the signal of AP1 demonstrates a steady decrease, and the signal of AP2 begins to appear at around 10s. At around 20s, the trigger condition is satisfied (the signal of AP2 is better than that of AP1, and AP1 shows a continuous decline trend, but the AP2 signal keeps increasing, and this is the second time that AP2 can provide better QoS than the AP1). After the handoff procedure occurs, connection with the GCM service is lost, and the first new message received from the data is at around 23s. As the traditional mechanism of the Android OS, the existing WIFI AP is not disconnected until the signal disappears. Hence the handoff timestamp for the conventional approach is at around 22.5s. However, between 20s and 22.5s, the QoS of AP1 may be not good enough to maintain message dissemination. Additionally, if the mobile node stops or slows down during that period, the switch timestamp will be later. But our methods can handle the network handoff more flexibly. The proposed method reduces Scenario 1 by around 2.5s (we suppose that the switch time cost is same in either way, as is the reconnection time with the GCM service).

Figure 7.7 demonstrates the test outcome in Scenario 2. The one-way delay of the message from the active data to the mobile node is stable before the handoff occurs at the 16s. However, the overall time delay in the mobile network is slightly higher than in the WIFI network. The conventional handoff time point is at around 8s because the Android OS connects by default to the configured WIFI network as higher priority than the mobile network even though it is weak. Sometimes, however, a weak WIFI signal may not provide satisfactory QoS. In particular, when the mobile node walks around in the period, the disconnection time will be longer. Our proposed method begins to handoff the network at around 16s, and the first new message received from the data is around 8s later, which is slower than in Scenario 1. The reason may be that the switch time cost is higher and the network address switch mobile-WIFI is longer than WIFI-WIFI.

The self-adapting handoff algorithm concerns the movement trend of the mobile node. It reduces the ping-pong effect and improves the matching ratio by using a double threshold scheme and dynamic sampling interval algorithm. Service ubiquity can be achieved through the *GCMBaseIntentService* which runs in the background in the Android phone. When the mobile node changes its network connection interface, the service will inform the GCM server to relocate the new client address. Furthermore, the associated computation involves elementary calculation which suits low energy and low computational capability mobile terminals.

7.3 Performance and Cost Tests

In this section, we perform stress testing, verification and identification cost testing, operational cost testing; we test the cost of storage on the ADCu, and compare the performance of executing active auditing services on diverse operations on the ADCu.

7.3.1 Stress Testing on ADCu

Stress testing is generally utilized to test the stability and reliability of a given system or entity (Wikipedia, 2013h). We perform stress testing by targeting a large number of concurrent client requests to a single instance of an ADCu. We utilize an external laptop (Apple MacBook Air 13” 1.8GHz/4GB/128GB flash storage MD231X/A) as the client-side request generator to simulate a real scenario of concurrent client requests by thread pool techniques. We allocate 50 working threads to generate user requests in parallel. Notably, we only test the query operation on the ADCu to observe its performance under different request stresses. Figure 7.8 shows the time cost when we gradually increase the request traffic from 10 requests to 320 requests. Generally, the time cost displays linear growth along with the increase in request traffic. This result indicates that the ADC framework can efficiently avoid an exponential increase in response time when a large number of requests are directed to one active data instance, due to the fact that the ADCu runs in the runtime environment as an atomic instance, a massive request would not directly crash the ADCu runtime instance. Each request has to be processed in ADCu one after one to ensure the atomicity. Hence, to achieve better load balance to process user requests, we can simply distribute and deploy multiple replicas of the ADCu to geographically separate hosts.

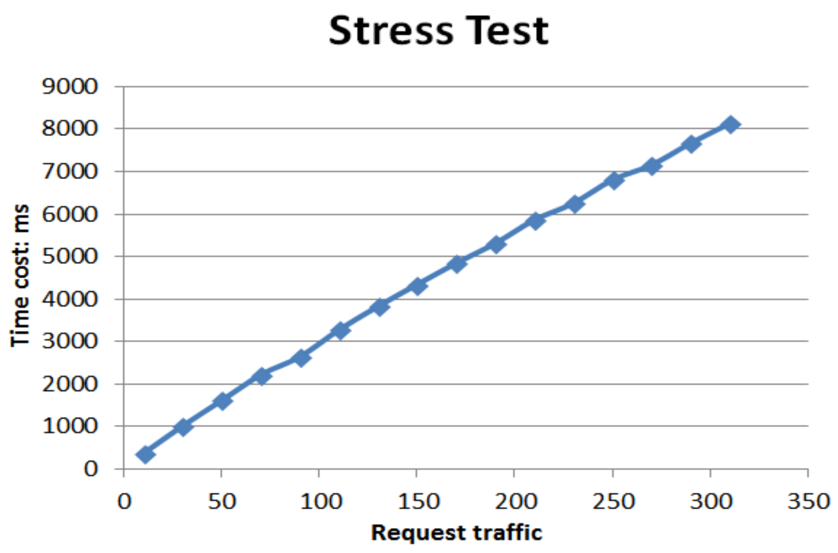


Figure 7.8 Stress test on an ADCu

7.3.2 Verification and Identification Cost on ADCu

The overhead for data retrieval from an ADCu involves verification and identification, data loading, and network communication costs. The network communication cost, however, is not significant and can be regarded as negligible due to the high-speed intranet test environment. Thus we mainly evaluate the verification and identification costs. The result shows that the most important element affecting the cost of verification and identification is the length of the prime number q. Due to the randomness of generating an appropriate prime number with a certain bit length and meeting the condition that p-1 is divisible by q, the time cost for parameter initialization shows obvious variation. Hence, we initialize five sets of tests in which the bit length of q ranges from 30 to 70, and each set of tests runs 10 times circularly. Figure 7.9 illustrates the time cost of executing the ZK proof scheme in the ADCu. The x-axis represents the bit length of q (bits), and the y-axis represents the execution time (ms). As we can observe, the time cost for generating the system's public parameters clearly increases along with the increase in the bit length of q. However, the average time cost for a 40 bit length q (844.9ms) is similar to that of a 50 bit length q (1126.1ms). Considering that the time taken to generate the system's public parameters may be an obstacle to achieving light weight of the ADCu, we suggest that the bit length of q be between 40 and 50 bits. In this case, q will be larger than 2130, and t can be larger than 65. Hence, the probability of success for adversaries to guess the correct e will be less than $2.7 * 10^{-20}$, which is sufficiently secure to resist a discrete logarithm attack. Moreover, the time cost for the process of generation and verification of user parameters remains almost constant with the growth of q, and is approximately 50ms.

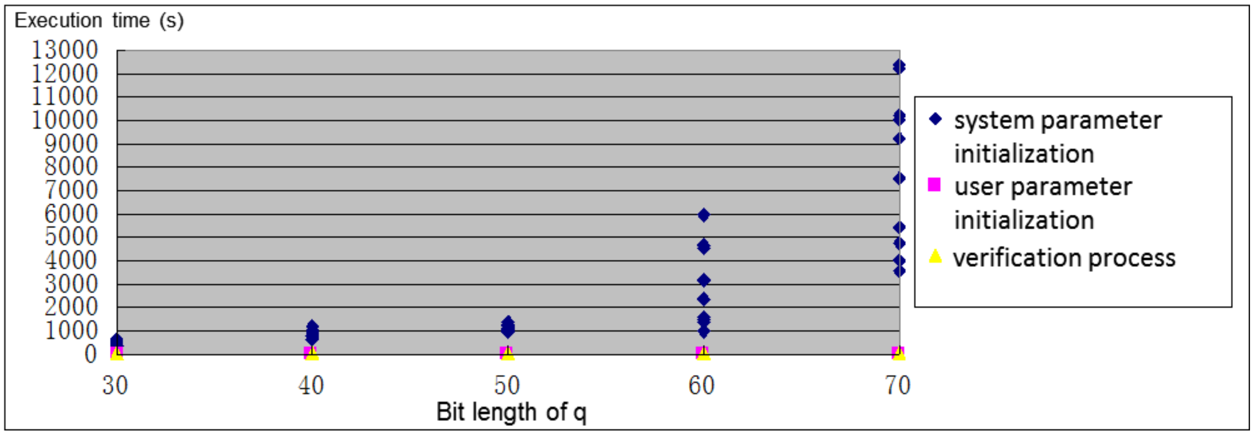
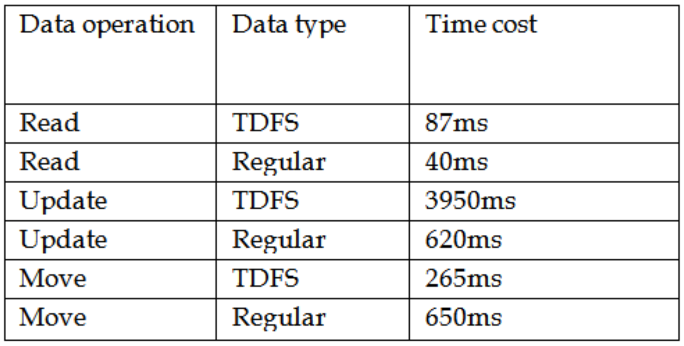


Figure 7.9 Time cost of verification and request identification in the ADCu

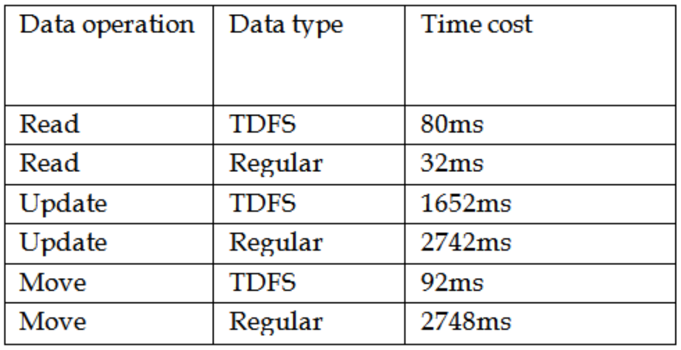
7.3.3 Operation Cost of ADCu

This set of experiments compares the regular data operations in the OS and the data operations in the ADC framework. We skip measuring the decryption of data files due to the fact that the ADCu does not decrypt the data block inside the active cube, but executes it on the fly. Figure 7.10 presents the outcome for two data quantities. Panel a shows the results for small quantities. The ADCu in this panel contains 42 image files with initial 72.8 MB size, and the updated ADCu size is 71.9 MB. Panel b shows the results for large quantities. The ADCu in this panel contains 1811 script files with initial 21.8 MB size, and the updated ADCu size is 15.9 MB. We observe that the performance of the read operation in the ADCu (referring to the TDFS data type in Figure 7.10) is almost 50ms longer than the regular read operations in both data quantities. This can be explained by the fact that the read operation in the ADCu requires the verification and identification procedure (Chen and Hoang, 2012) which takes approximately 50ms, but the regular read operation does not require that procedure. However, if we take security and data protection issues into consideration, the overhead of this procedure is a small price to pay. The overhead of updating a bunch of data into the ADCu results in a much slower performance for small data quantities but a slightly better performance for large data quantities. For the move operation, the ADCu reflects better performance with both data quantities, especially for large data quantities by virtue of the encapsulation feature of the ADCu. In summary, the ADCu presents a slower performance only when executing an update of small data quantities. The excess overhead in read operations can be ignored for security considerations.



Data operation	Data type	Time cost
Read	TDFS	87ms
Read	Regular	40ms
Update	TDFS	3950ms
Update	Regular	620ms
Move	TDFS	265ms
Move	Regular	650ms

(a)



Data operation	Data type	Time cost
Read	TDFS	80ms
Read	Regular	32ms
Update	TDFS	1652ms
Update	Regular	2742ms
Move	TDFS	92ms
Move	Regular	2748ms

(b)

Figure 7.10 Time cost for reading, updating, and moving operations between the regular data file and the ADCu with (a) small and (b) large data quantities

7.3.4 Storage Cost on ADCu

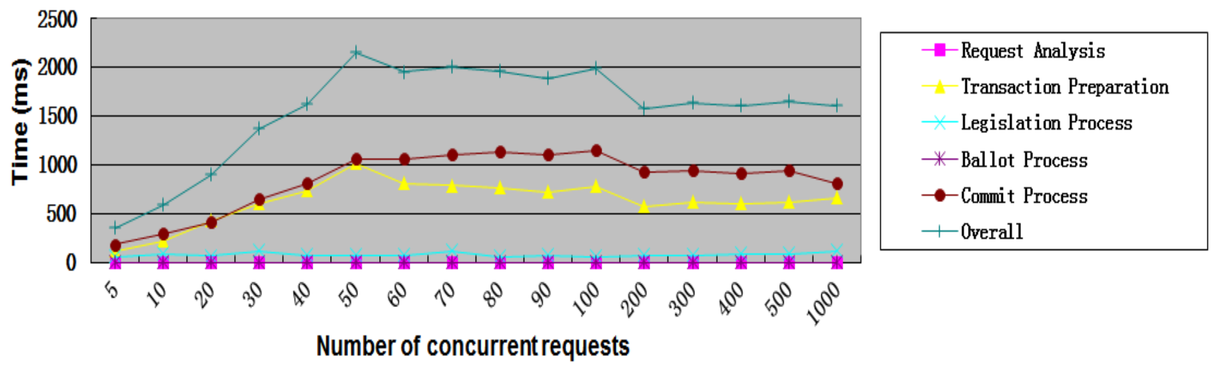
As a result of packaging active scripts on data, storage overhead must be added. We measured the shell size, the core size without stored data, and the eventual ADCu size. Due to the fixed size of shell, core scripts, and header, the entire ADCu only increases in size when the embedded data blocks grow. The shell scripts occupy 78KB storage space and the core scripts including the header occupy 901KB. However, the eventual empty ADCu occupies only 149KB due to the compression feature of JAR technology. In special cases, if the data block files have a high compression ratio, such as text files, the updated ADCu demonstrates better efficiency in storage cost. To further test the average creation cost of an empty data cube, we leveraged the "ant" technique (Kallambella, 2006) to automatically build a smart data cube using the configured build.xml. The average creation time was approximately 150 ms when executed 30 times cyclically. The overall experimental outcome demonstrates the light-weight feature and shows that not too much overhead is added.

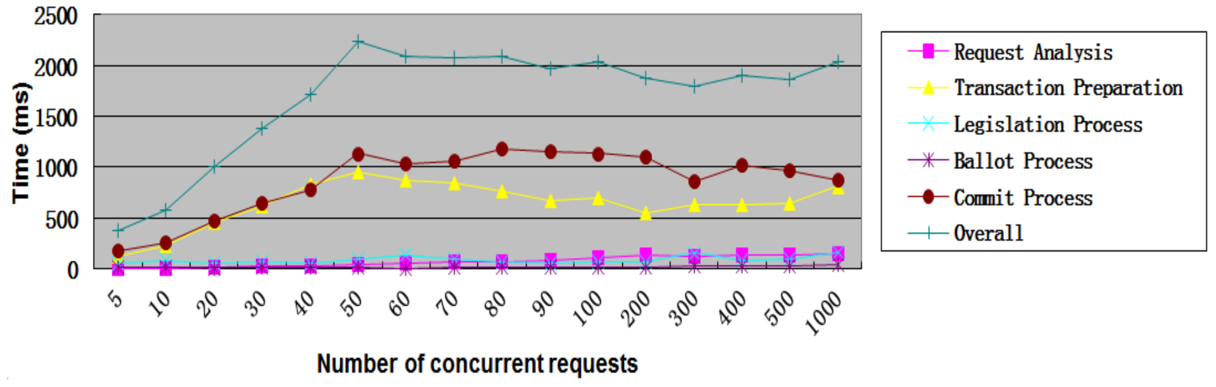
7.3.5 Performance Comparison of Executing Active Auditing Services on Diverse Data Operations on ADCu

To test the performance of executing active auditing services, we utilize an external laptop (Lenovo ThinkPad Edge40 with Intel i3 2.26GHZ CPU in 4 cores, 2GB RAM) as a client-side request generator. We create a separate working queue to evaluate the performance of the framework with read-only processes, update-type processes and move (external-type) processes. The number of concurrent requests varies from 5 to 1000. The CPRBAC service policy is expressed in XACML. We generate only 30 policies in the policy repository, in which each policy contains *Subject*, *Object*, *Action*, and time-type *Condition* properties. To guarantee that all transactions can be eventually committed, we set all properties of the service requests as legitimate requests. Before starting the evaluation, we warm up the RMI-SSL servers, the policy loading and the initialization of components in a transaction. Figure 7.11 indicates the time cost of the above three types of operation from 5 concurrent requests to 1000. The x-axis describes the number of concurrent requests (CR), and y-axis is the average time consumption (ATC) of all transactions which is calculated by the total amount of time cost divided by the CR. The results indicate that our scheme executes efficiently for both read-only and update types of operations.

The highest ATC appears when the CR reaches 50, and the update-type operations are nearly 100ms slower than the read-only type operations on the data. When the CR is less than 50, the ATC demonstrates a steady increase with respect to an increase in the CR. When the CR is greater than 50, the overall ATC demonstrates a slight fluctuation, but stabilizes between 2000ms and 1500ms. However, it is observed that update-type operations require 100-150ms longer execution time than read-only operations. This can be explained by the fact that the update-type operations require more work in detecting conflict, verifying monitor preparation, and modification operation on the data. These steps are not included in the read-only operations.

The move operations demonstrate a different profile. The first maximum ATC appears at the point where the CR reaches 70, which is 1358 ms, and the ATC drops to around 1000ms until the CR reaches 200. The ATC then increases steadily. It is observed that the greatest difference between the move operation and the two internal operations is in the commit process stage. For the move operation, the ATC is relatively short until the CR reaches 200, but for the two internal operations, the ATC rises sharply. This is due to the fact that the move operation merely requires invoking the OS's underlying operations in the commit stage, which is highly efficient, whereas the two internal operations require processing query or modification work on the ADCu.





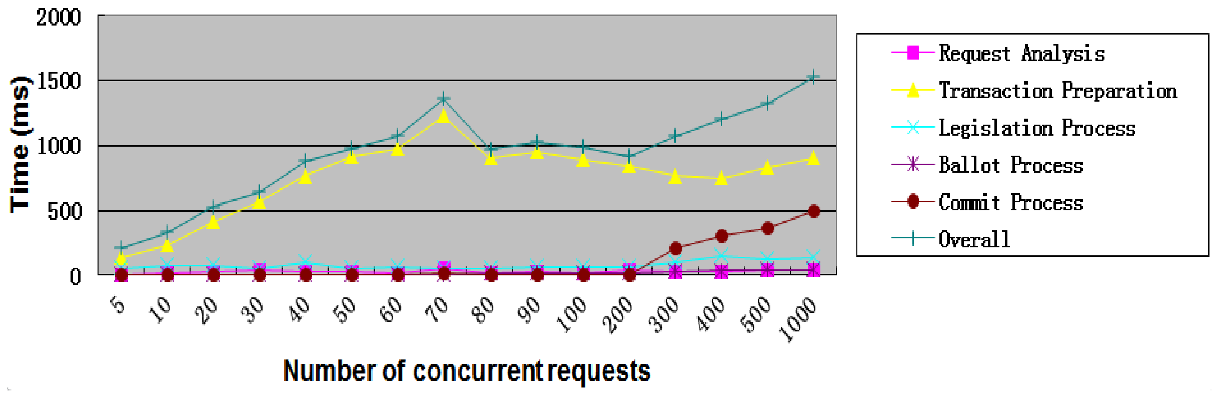


Figure 7.11 Time cost of read-only (top), update-type (middle), and move operations (bottom) in the AAS

7.4 Comparison of Different Data Protection Mechanisms Applied in Cloud

In this section, we compare different data protection mechanisms discussed in Chapter 2 with our trust-oriented data protection framework in cloud from several aspects: performance, focus, intrusion attack, trust, accountability, immediacy, and mobility. Table 7.2 illustrates the comparison results that indicate the benefit of adopting our framework.

Cryptography-based schemes focus on addressing the confidentiality and integrity of outsourced data in cloud. The data must be encrypted before storage in cloud data storage nodes, and data security and privacy depend highly on the quality and robustness of the encryption algorithms. This scheme inevitably introduces high overheads on both storage and computing for encryption or decryption. The trust of data security is based on a highly secure encryption algorithm. Since encrypted data remains static, it cannot prevent an intrusion attack by itself, but relies on third party services. Other relevant properties, such as accountability of data usage, immediacy of violation alert, and mobility management for encrypted data, are not supported when deploying exclusive cryptographic mechanisms.

Policy-driven mechanisms focus on handling authentication and authorization on the usage constraints of users' data. When data is processed, the action must adhere to the policy specification. The performance of this mechanism usually depends on the complexity of policy structure, quantity of policy, and evaluation procedure. This is security control layer strategy. Hence, it cannot provide accountability, instantaneity, and mobility for data usage in cloud, only authentication and authorization. Moreover, it cannot prevent an intrusion attack on data if adversaries penetrate the access control layer. Therefore, it is also difficult to provide trust to users who subscribe to CSPs.

A partially trustworthy relationship between CSPs and users can be provided through trusted third party management systems which may include features such as security ranking, auditing, and accounting. However, such schemes lack a certain protective capability for data per se. They cannot efficiently detect an intrusion attack prior to periodical auditing. Hence, immediacy of data security protection may be lost. Mobility management of data is also not offered by this strategy.

The data policy binding framework was introduced in recent research to address data security and privacy issues through encapsulating data, policy, and functional entities within an independent object. This object is useful for preventing an intrusion attack via the embedded access control module. It can compulsorily process data requests in terms of bundled security procedures. This scheme only provides partial trust for users due to the concern of lack of mobility. Moreover, executing fine-grained authentication and authorization in the object may cause more overheads if the requests are intensive.

Despite the attractiveness of the idea of homologous data policy binding frameworks, our work presents more features and advantages. We introduce mobility management on active data, which improves users' trust through bundled mobile devices. Any violation or ongoing danger can be compulsorily informed to data owners. Moreover, our method can provide regular security functionalities in addition to the data policy binding framework, such as preventing intrusion attack, accountability, and immediacy. However, we move the authorization and authentication work from inside active data to external third party access control servers for reasons of security, flexibility, and performance. The experimental outcomes prove that our proposed framework can establish a more trustworthy data protection infrastructure that can be deployed in sensitive areas that require accountability, auditing, mobility, trust, intrusion-tolerance and reasonable performance.

Table 7.2 Comparison of different data protection strategies in cloud

Properties / Strategies	Performance	Focus	Intrusion attack	Trust	Accountability	Instantaneity	Mobility
Cryptographic-based	Poor, high overhead	Confidentiality Integrity	no	Partially trusted	no	no	no
Policy-driven	Medium	Authentication Authorization	no	no	no	no	no
Trusted computing	Medium	Accountability Trust Auditability	no	Partially trusted	yes	no	no
Data policy binding	Poor, high overhead	Accountability Confidentiality Integrity	yes	Partially trusted	yes	yes	no
Active data-centric and trust-oriented framework	Medium	Trust Accountability Confidentiality Integrity Availability Intrusion-tolerance Auditability	yes	trusted	yes	yes	yes

7.5 Summary

In this chapter we have demonstrate results of experiments conducted on our lab's mini-private cloud environment to evaluate and analyze the proposed trust-oriented data protection approaches, protocols, and applications. First, we analyzed data security of the proposed framework through a number of regular attack cases. Subsequently, we proved the feasibility of the proposed schemes via functionality and security tests. To demonstrate that little overhead was added to the novel data protection framework, we conducted performance and cost tests. Our results indicate that the proposed ADCu can actively repel intrusion attacks, audit data usage, and inform bundled data owners with very reasonable efficiency and light weight. Finally, we compared the mechanisms discussed in Chapter 2 with our strategies in detail.

Chapter 8 Application of protecting healthcare data in cloud

Traditional eHealth technology enables patients and physicians to manage medical records in a centralized way, which greatly facilitates the storage, access and sharing of electronic health records (EHR) under a secure and isolated environment (Li et al., 2010). Although applying eHealth in distributed cloud can achieve elastic resources and lower operational cost, it may entail issues and challenges to security, privacy, and maintenance of control, as discussed in Chapter 1.

The previous chapters introduced a trust-oriented data protection model that combines an active and passive data protection framework in cloud, which comprehensively addresses cloud data security, privacy, and controllability issues. In this chapter, we focus mainly on applying our trust-oriented data protection framework for protecting patients' EHRs in health cloud environment.

8.1 Introduction of e-Health

In the era when information technology was not well developed, healthcare providers (such as clinics, hospitals, etc.) stored patients' medical records on paper. This scheme provided a controllable environment with easy management and protection of data security and privacy (Löhr et al., 2010) via keeping the paper records in a relatively isolated, sealed, and secure space such as a locked archive room. In recent years, with the continuous development of information technology, an increasing number of hospitals and clinics have digitized patients' medical records and stored their health records on independent data servers using eHealth technology (Li et al., 2011). For instance, as reported published in May, 2010 (Government, 2010), national eHealth record systems were regarded as the key building block of the Australian health and hospitals network. The Australian government planned a $466.7 million investment to revolutionize the delivery of healthcare.

eHealth technology addresses more sophisticated and cumbersome recording issues than the traditional paper-based recording approaches (Li et al., 2011). Physicians can query and edit patients' health records via simple web interfaces, which is more convenient and beneficial for patient treatments. Patients can check and query their online health records and treatment history at home without professional medical knowledge. Additionally, eHealth systems such as EHRs can avoid expensive double diagnoses or repetitive drug administration, thus decreasing costs of EHR management, but improving the efficiency of healthcare administration.

Examples of national activities for applying eHealth technology are evident in countries such as Australia, Germany, Taiwan, and the US, that have deployed their national healthcare network (association, 2012). In Australia, eHealth is significant. A report in 2012 (association, 2012) claimed:

"Australia's aging population is placing unprecedented pressure on our healthcare system. In 1970, there were 7.5 working age people to support people over the age of 65; today there are 5.0. By 2050 that figure will fall to 2.7. (Intergenerational Report 2010) • Ageing and health pressures are projected to result in an increase in total government spending from 22.4 per cent of GDP in 2015–16 to 27.1 per cent of GDP by 2049–50. (Intergenerational Report 2010)

In the aged care sector it is estimated that by 2050 the number of people aged over 65 years will increase seven fold, and increase over twelve fold for those aged over 85 years (Intergenerational Report 2010)

Up to 25% of a clinician's time is estimated to be spent collecting data and information about their patients. (Australian Audit Commissions, For Your Information, Canberra, 1995)

Up to 18% of medical errors occur as a result of inadequate availability of patient information whilst the inappropriate use of medicines costs $380 million per year in the public hospital system alone. (Australian Institute of Health and Welfare, Australia's Health 2002)"

A common infrastructure in the above systems is to store medical data in centralized data centers. They adopt healthcare standards such as the Health Level 7 consortium (HL7). Most hospitals and clinics now deploy their own databases to manage patients' EHRs. The exchange and sharing of these EHRs may be easily done with the same electronic healthcare standards and servers. However, some small-scale clinics do not have the capable servers to manage and exchange patients' EHRs (Zhang and Liu, 2010), which may slow down or even block the exchange of EHRs. Nevertheless, exchanging EHRs between different medical centers or research organizations can improve the efficiency of diagnosing disease and support treatment of the disease.

8.2 Problem Statement Regarding Storing EHRs in Cloud

It is exciting to deploy healthcare-related services on cloud platforms. However, the security and privacy risks entailed by the cloud's outsourced pattern could impede its wide adoption in the eHealth area. The main concern is that patients might lose physical control of their own EHRs. They might not know who can gain access to the EHRs, how the EHRs are processed, what details or data are disclosed to others, and whether the procedure of security and privacy protection conforms to well-defined SLAs. Furthermore, existing work lacks appropriate schemes to protect users' sensitive EHRs from illegal disclosure or malicious violation by an employee within the CSPs. Such an incident has occurred: an employee stole the US Department of Veterans Affairs database that contained sensitive personal health information of 26.5 million military veterans and took it home without authorization (Foreman, 2006). This event illustrates that it is extremely difficult to detect internal misbehavior.

To deal with such potential risks, our solution and mechanism focus on giving patients (EHR owners) sufficient control over their outsourced EHRs in cloud. To this end, the raw EHR data needs to be encapsulated in an active data cube (ADCu) created by the ADC framework. The ADCu can provide data operation transparency via inner and external monitoring, compulsory logging, and checkpoint execution. The active and encapsulated data structure also satisfies data confidentiality and integrity requirement. As well, our proposed CRPBAC service and AAS service enable EHRs to be rationally protected via authorization policies that describe the conditions under which users can be permitted access the resources under normal circumstances. Policy or credential inconsistency can be detected by the transaction management in the AAS and a rollback to the previous correct checkpoint can be applied.

8.3 Securing EHRs in Cloud by the Proposed Trust-Oriented Scheme

Figure 8.1 shows an overall blueprint for securing EHRs in cloud by our proposed active-passive combined data protection mechanism. The raw EHR is static in that the data itself lacks the capability of self-protection once it is transferred to the cloud. Furthermore, data owners may lose the control of their data. Additionally, cloud architecture is required to accommodate multiple tenants and to satisfy multi-tenant features. As discussed in previous chapters, the ADC framework is used to encapsulate and secure static raw EHR data of patients in the data storage node. This structure not only provides an isolated secure environment for each cloud user but also achieves a safe resource sharing scheme in a distributed cloud environment. The CPRBAC and AAS services take charge of monitoring, auditing, and controlling the manipulation of active EHRs. A smart phone can be used to execute dual communication between patients and their data. To guarantee the availability of data, we deploy an adaptive data replica management scheme in cloud.

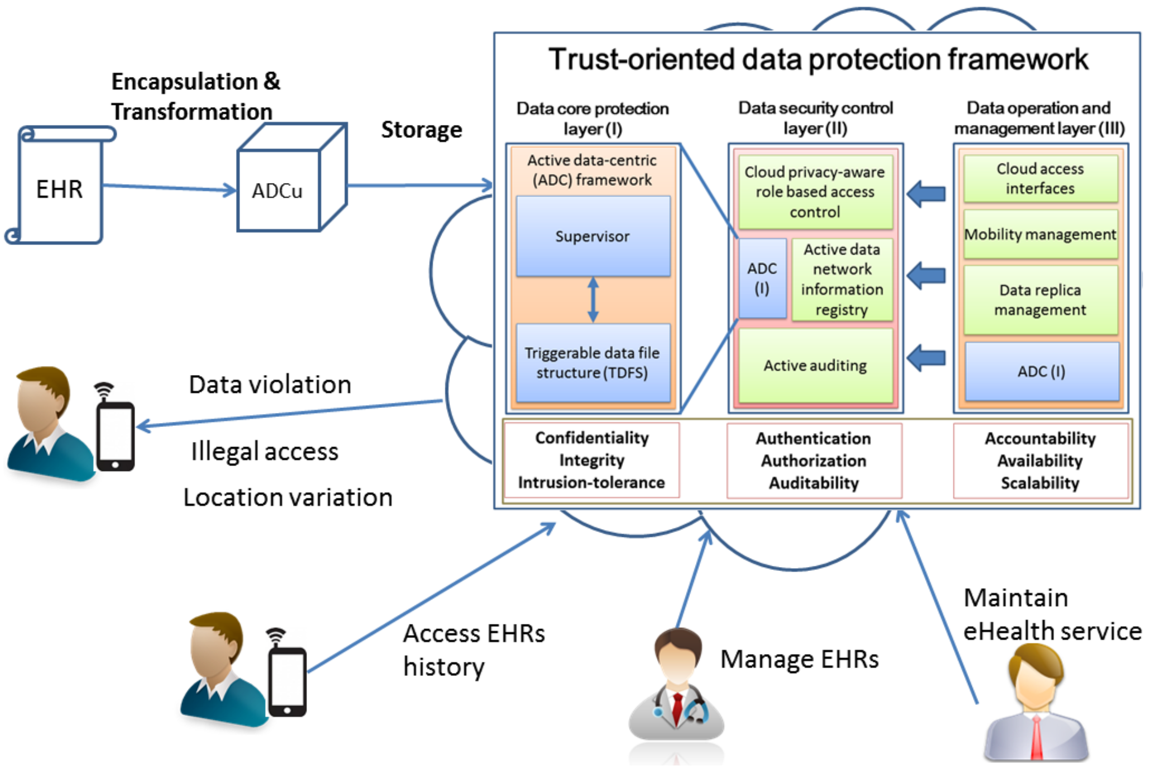


Figure 8.1 Blueprint for securing EHRs in cloud using the trustworthy data protection framework

8.3.1 EHR Transformation and Node Labeling

An EHR in the eHealth context normally exists in the form of an XML-type sequential structure specified by eHealth standards such as HL7. Such a record may include a comprehensive set of relevant health data in a specific format. We propose a tree-type, hierarchical structure with a set of linked nodes (or data elements), to represent a health record in an integrated and hierarchical manner. This tree-type data structure dynamically supports simultaneous data update and delete when a new updated inode does not have a parent-child relationship with the updating inode. Mathematically, it is an ordered directed tree separating data contents into organized and hierarchical sub-contents. The tree begins with a root node. The inode contains a *code header*, *attribute tag* and *pointer* to the next-tier inodes or to direct leaf-nodes which contain the attribute value of their inode. We leverage symmetric data encryption key to encrypt the attribute values. But we only encrypt the leaf-nodes for consideration of light-weight computation overheads.

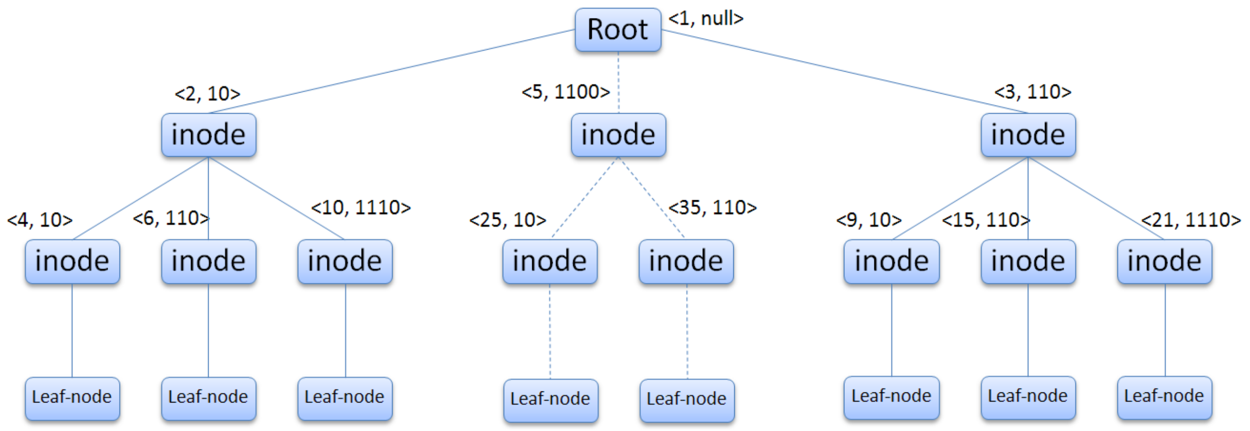


Figure 8.2 Tree-type EHR data blocks

Figure 8.2 depicts a tree structure of EHR data blocks, where the dashed-line branch is a collection of newly inserted data. We index our inodes through an encoding scheme to facilitate rapid determination of ancestor–descendant and parent–child relationships among the inodes. The encoding scheme synthesizes the characteristics respectively by a prime number labeling scheme (Hye-Kyeong and SangKeun, 2010) and binary string approach (Wu et al., 2004). Querying and updating order-sensitive nodes can be achieved with high efficiency. Importantly, the code of an inode only reveals its relationship with other inodes. It does not disclose any sensitive information associated with that inode.

Encoding scheme for EHR-tree inode:

Definition 1: ***Prime set:*** $P_i = <2, 3, 5, 7, 11, 13.......>, 1 \leq i$

Definition 2: ***Ancestor-descendant relationship:*** if and only if inode (A_{prime_BIT}) mod inode (B_{prime_BIT}) =0 or inode (B_{prime_BIT}) mod inode (A_{prime_BIT}) =0

Definition 3: ***Sibling relationship:*** if and only if inode (A_{prime_BIT}) and inode (B_{prime_BIT}) have the same largest prime factor.

Definition 4: ***Lexicographical order:*** given two consecutive binary strings S_1 and S_2, their lexicographical order is as follows:

$S_1 \equiv S_2$ (lexicographical equality) => S_1 and S_2 are exactly the same.

Comparison of S_1 and S_2 is carried out bit by bit from left to right. If the current bit of S_1 is 0 and current bit of S_2 is 1, then $S_1 < S_2$

If length(S_1) < length(S_2), S_1 is the prefix string of S_2, and S_2 removes the same prefix string of S_1, the remaining bits are 1, then $S_1 < S_2$

If length(S_1) > length(S_2), S_2 is the prefix string of S_1, and S_1 removes the same prefix string of S_2, the remaining bits are 0, then $S_1 < S_2$

Definition 5: ***Encoding scheme:*** the code for each node except leaf-nodes in the EHR-tree consists of a tuple with two elements which are a prime number or a prime product and a binary string, which can be <prime/prime product, binary-string>, the prime_BIT denotes the prime value of the inode, the binary_BIT denotes the binary value of the inode. The prime_BIT is used to illustrate the order of creation, the binary_BIT is used to illustrate the order of space in the tree. The generation rules are as follows:

The code for root is <1,null>

On the first layer of the EHR-tree, the i_{th} sub-inode can be encoded as $<P_i , 1^i0>$ (if i=3, the code for the inode is <5, 1110>; if i=2, the code for the inode is <3,110>)

The other i_{th} sub-inode on the EHR-tree can be described as $<N_{father}*P_{k+i-1}, 1^i0>$ (we assume the largest prime factor of the father node is P_k ; N_{father} is the code for its father node)

Definition 6: ***Insertion scheme*****:** the code for the left node at the insertion position is <a,b>, for right node is <c,d>, the largest prime factor among the sibling inodes is P_k, the parent prime_BIT is P_p, the code for new inode is <e,f>

```
If ($P_p$==1)        //parent node is root
e=$P_{k+1}$
Else if ($P_p$>1)      //parent node is not root
e=$P_p \times P_{k+1}$

Case 1: insert from left side (left node is empty, right node is not empty)
f=b $\ominus$ 0 ($\ominus$ denotes concatenation operation)
Case 2: insert from middle (left node and right are not empty)
If length (b) $\leq$length (d), f=d $\ominus$ 0;
Else if length (b) > length (d), f=b $\ominus$ 1;
Case 3: insert from right side (left node is not empty and right node is empty)
f=d $\ominus$ 1;
```

For instance, in the tuple <prime/prime product, binary-string>, prime/prime product is used to sort the relationships of all nodes. In the second tier of Figure 8.2, we have 2,3,5 (they are primes, and the inode with 5 is inserted after the inodes with 2 and 3). So in the third tier, all children of the inode with 2 will have a code using <2*2, 2*3, 2*5>, similarly, we have <5*5, 5*7> for the inode with 5's children, and <3*3, 3*5, 3*7> for the inode with 3's children. Here, we have <2*2>, and the first 2 is its first parent, the second 2 is the order in current tier. Hence, if the tree has next-tier inodes, we can calculate the children for the inode <4,10> such as <2*2*2,2*2*3,2*2*5>. We can easily know the relationship from the prime value. For example, <2*2*2> means that its top parent has prime code 2, its second-tier parent has prime code 2, and it is the first node in the current tier.

The binary-string portion is used to describe the inserted location in current tiers. We can see the first set inodes in the third tier <4,10>,<6,110>,<10,1110>. In the normal case, 10,110,1110 means that the inode is inserted orderly from the right side. The inode <5,1100> means that the inode is inserted from the middle (left node and right node are not empty), and the length of the right inode (110) is longer than that of the left inode (10). So we append 0 at the end of the right inode (110), and we get <5,1100>. If the length is shorter, we get <5,101>. Hence, through <5,1100>, we can know that the new inode is inserted from middle by comparing the left side or right side. 1100 is longer than 110 and 0 is inserted at the end of 110, which indicates that the inode 1100 is inserted from 110's left side.

In summary, this encoding scheme takes advantage of the unique property of prime numbers. High efficiency query of nodes can be simply captured by prime value decomposition. Simultaneously, a complementary binary string scheme enables updating of a highly dynamic ordered EHR-tree without recoding the existing node. In the following section, we demonstrate some examples created for protecting EHRs.

8.3.2 Examples of ADCus Storing Patients' EHRs

For verified requests, the data loader in the ADCu replaces third party services to execute the required data operations. There are two benefits from this scheme: 1) we can protect data from violation even in the compromised cloud environment, since the ADCu encapsulates data and functional scripts that can execute verifications and data operations independently; 2) it also reduces the burden of third party services. Subsequently, EHR data in the resource folder is constructed according to the proposed tree-topology. An XML schema is employed to encapsulate the data and the corresponding codes. In contrast to the normal depth-first traversal of an XML-tree, the proposed scheme can conduct data retrieval that does not require traversing the whole tree. The program can analyze the attribute's code value and obtain its index with complexity *O(1)*. The encoding scheme has proven ability to efficiently support update in an ordered XML-tree without modifying the previously allocated inodes.

It should be noted that the traditional EHR is static data that relies on third party services to manipulate and protect data usage against violation through cryptographic primitives, anonymity, dispersal storage, access control, and so on. Figure 8.3 is a raw EHR example derived from (Dolin et al., 2000), in which we have extracted a portion of the EHR data to illustrate the proposed scheme. The EHR in the example consists of 15 content nodes. The document complies with HL7 standard. After transformation of the raw EHR, it will turn to two parts: one part is the header, recording all metadata and inode code related to the EHR body. Figure 8.4 illustrates the header information in the core of the ADCu. The other part is the body, recording all EHR information, but all nodes are encoded by the encoding scheme and all leaf-nodes are encrypted. Figure 8.5 depicts the body information in the core of the ADCu. Subsequently, a new EHR header and body are encapsulated in an ADCu with active scripts.

eHealth cloud administrators can authorize policy according to either a policy template or an example format. Figure 8.6 shows a policy editing page in our eHealth cloud web portal. Upon completion of policy editing, the CPRBAC service is required to analyze the validity of the policy, including its structure and whether the new policy conflicts with an existing one in the policy repository. This web portal is still under development and we intend to complete it and deploy it in our lab's cloud platform.

```
<?xml version="1.0" encoding="UTF-8"?>
<EHR>
        <header>
                <date.of.creation>16/09/2011</date.of.creation>
                <date.of.study>16/09/2011</date.of.study>
                <PID>123456789</PID>
                <dob>January 13, 1923</dob>
                <name>cenling40</name>
                <sex>M</sex>
                <practitioner.id>24680</practitioner.id>
                <practitioner.name>Dr. Amy A. Fall</practitioner.name>
        </header>
        <body>
                <section caption="History of Present Illness">
                        <paragraph>
                                <content>cenling40, the 7th is a 67 year old male referred for further
asthma management. Onset of asthma in this teens.
                                He was hospitalized twice last year, and already twice this year. He
has not been able to be weaned off steroids for the past several months </content>
                        </paragraph>
                </section>
                <section caption="Medications">
```

```
                        <list>
                                <item caption="M1">
                                        <content>Proventil inhaler 2puffs QID PRN</content>
                                </item>
                                <item caption="M2">
                                        <content>Prednisone 20mg qdc</content>
                                </item>
                        </list>
                </section>
                <section caption="Physical Examination">
                        <subsection caption="Vital Signs">
                                <list>
                                        <item caption="BP">
                                                <content>118/78</content>
                                        </item>
                                        <item caption="Resp">
                                                <content>16 and unlabored</content>
                                        </item>
                                        <item caption="HR">
                                                <content>86 and regular</content>
                                        </item>
                                </list>
                        </subsection>
                        <subsection caption="Skin">
                                <paragraph>
                                        <content>Erythematous rash, palmar surface, left index
finger.</content>

                                        <observation_media_value MD="image/jpeg"
                                                ADR="rash.jpeg"></observation_media_value>
                                </paragraph>
                        </subsection>
                </section>
        </body>
</EHR>
```

Figure 8.3 Raw EHR data of a patient

```
<?xml version="1.0" encoding="UTF-8"?>
<TDFS_header>
    <SID>
        <Revision>0.0.1</Revision>
        <Domain_Identifier>132.145.2.14:8080/CloudCluster1</Domain_Identifier>
        <Data_Identifier>13F8C8AF-CB87-4261-1F2E-8985A06DA0DE</Data_Identifier>
    </SID>
    <DDe>
        <Content_des Description="This is a EHR Example">
            <Practitioner_ID>24680</Practitioner_ID>
            <Practitioner_Name>Dr. Amy A. Fall</Practitioner_Name>
        </Content_des>
        <Owner_des PID="123456789">
            <Name>cenling40</Name>
            <DOB>January 13, 1923</DOB>
            <Sex>M</Sex>
        </Owner_des>
        <Data_checksum>
            <Checksum>-bf77jl3odafibkhpler7uatkua5t3sdi</Checksum>
            <Key>ZJeewbxzgEM=</Key>
            <Salt>-7768707957630818095<Salt>
        </Data_checksum>
        <Policy_index>
            <Index>sdfaes3sfsed23</Index>
        </Policy_index>
        <Caption-Code>
            <Attribute code="1,0">TDFS_body</Attribute>
            <Attribute code="2,10">History of Present Illness</Attribute>
```

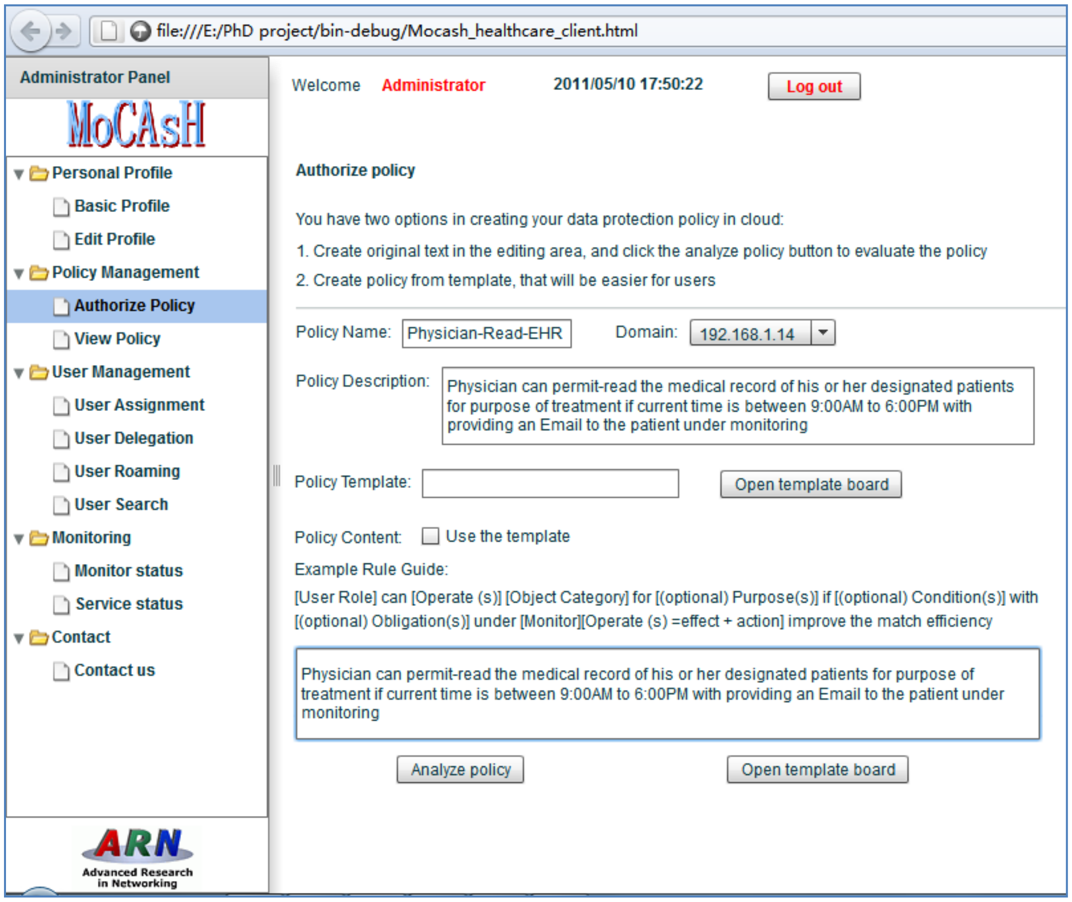
```xml
            <Attribute code="4,10">paragraph</Attribute>
            <Attribute code="3,110">Medications</Attribute>
            <Attribute code="9,10">M1</Attribute>
            <Attribute code="15,110">M2</Attribute>
            <Attribute code="5,1110">Physical Examination</Attribute>
            <Attribute code="25,10">Vital Signs</Attribute>
            <Attribute code="125,10">BP</Attribute>
            <Attribute code="175,110">Resp</Attribute>
            <Attribute code="275,1110">HR</Attribute>
            <Attribute code="35,110">Skin</Attribute>
            <Attribute code="245,10">paragraph</Attribute>
            <Attribute code="1715,10">observation_media_value</Attribute>
        </Caption-Code>
    </DDe>
    <Timestamp>
        <Creation_time Time="16/09/2011" />
        <Expire_time Time="15/12/2011" />
        <Last_Modified_time Time="16/09/2011" />
    </Timestamp>
</TDFS_header>
```

Figure 8.4 EHR header in ADCu

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TDFS_body code="1,0">
    <section code="2,10">
        <paragraph code="4,10">
            <content>Q8Vqw1bgFspZr80HDlyE9g8i6h/C9/SAlP1Xiy8Mxqk5//srDOOFhbU1wHpepKHF1z+2asRMEaq8
CqOO2+9lt/1nVHXCq74pMVdJCWfOHJIjpQ0Wpt/RuWee14bC7CuBiF2OLezZ6Edvq4Gr5iJ54B4+
Ji85bdcUvPHOQHH4dPK0tPMoJrmTUA4Y0/nZM9q7vV95eUA9GPWrnbIN+Q0EFpU4+5RyCfg89B+1
R7Eilsl0qouNxYRwfdJSGUJdElBRg7Okc0YXhz4f78muoed4m4xKoNpH0eKamo0hj0hssetCCt+e
QHzT4i+cbIemaalaIE4YO8TYVapHV2aDKxjpynSMonXOQLUR</content>
        </paragraph>
    </section>
    <section code="3,110">
        <list>
            <item code="9,10">
                <content>QXw2FXETCSxJ8dNDE2+F7lHA3SEiYVq/sURIFohvmuyKPH/UXgLTTQ==</content>
            </item>
            <item code="15,110">
                <content>bLMzp/Q/g1FUxg+Hle8E+IJYhL0YRMlK</content>
            </item>
        </list>
    </section>
    <section code="5,1110">
        <subsection code="25,10">
            <list>
                <item code="125,10">
                    <content>VOVeSj7cYrc=</content>
                </item>
                <item code="175,110">
                    <content>cU74jxM3auOxAEqmtV08joo8f9ReAtNN</content>
                </item>
                <item code="275,1110">
                    <content>w4622e2oTtilz8WAwAdjbg==</content>
                </item>
            </list>
        </subsection>
        <subsection code="35,110">
            <paragraph code="245,10">

<content>8sHY8inwcbedyFkoHNughOAz2tMLAmAuzPP8ZBc9O5G4XcRNGWjlBoJCfgTo4eGZSx1gs43RCEg=</content>
                <observation_media_value code="1715,10" MD="image/jpeg" ADR="rash.jpeg" />
            </paragraph>
        </subsection>
    </section>
</TDFS_body>
```

Figure 8.5 EHR body in ADCu


file:///E:/PhD project/bin-debug/Mocash_healthcare_client.html

Administrator Panel

Welcome Administrator 2011/05/10 17:50:22 Log out

Authorize policy

You have two options in creating your data protection policy in cloud:

1. Create original text in the editing area, and click the analyze policy button to evaluate the policy

2. Create policy from template, that will be easier for users

Policy Name: Physician-Read-EHR Domain: 192.168.1.14

Policy Description: Physician can permit-read the medical record of his or her designated patients for purpose of treatment if current time is between 9:00AM to 6:00PM with providing an Email to the patient under monitoring

Policy Template:

Policy Content: Use the template

Example Rule Guide:

[User Role] can [Operate (s)] [Object Category] for [(optional) Purpose(s)] if [(optional) Condition(s)] with [(optional) Obligation(s)] under [Monitor][Operate (s) =effect + action] improve the match efficiency

Physician can permit-read the medical record of his or her designated patients for purpose of treatment if current time is between 9:00AM to 6:00PM with providing an Email to the patient under monitoring

Figure 8.6 Authorize policy in the administrator page

8.4 Future Direction: eHealth Cloud

With the emergence of cloud computing technology, eHealth cloud has become very appealing. Cloud-based healthcare solutions raise new possibilities. Users can access and manage their outsourced EHRs easily and ubiquitously via the cloud's web portals or mobile portals. All confidential EHRs of patients stored in outsourced clouds simplify the complex exchange procedure, promoting collaboration and exchange between different services, organizations, and other third parties. Such eHealth cloud architecture also promises more cost-efficient service and improved service quality (Löhr et al., 2010).

Microsoft HealthVault (2013) is a valuable eHealth cloud service that helps users gather, store, use, and share their health information with permitted doctors. Apart from regular editing and management of healthcare records, the HealthVault service offers wonderful interfaces for patients to prepare for doctor visits, unexpected emergencies, and even to archive their fitness goals. Additionally, Microsoft does not charge users for using HealthVault. These benefits are attracting a growing list of laboratories, pharmacies, hospitals, and clinics that use the platform to improve their healthcare quality.

There is no doubt that the eHealth industry will shift to cloud-based solutions in the near future despite the existence of some impediments and challenges. Figure 8.7 is a vision of a future eHealth cloud. In this infrastructure, the eHealth cloud consists of a collection of VM resource pools, EHRs storage pools, healthcare service pools, and cloud management pools. These resource pools and services including EHR data storage, management, medical treatment, billing, insurance management, remote monitoring service, medical treatment interaction, and so on are provided to medical research centers, health insurance companies, hospitals, and clinics, via eHealth service interfaces. The greatest advantage for institutions or organizations is that they do not need to invest in hardware, software, and maintenance cost for hosting services. They can subscribe to the eHealth cloud provider and delegate their management and services on the provider's infrastructure. All they are required to pay for is their usage. Within the eHealth cloud, data and resources can be shared and exchanged collaboratively between medical research centers, hospitals and other institutions under stated policy regulations and security schemes. On the one hand, this architecture efficiently reduces operation and deployment costs for organizations. On the other hand, it improves healthcare quality because higher throughput, efficiency, and QoS can be provided by high-performance cloud servers, and medical collaboration can be carried out with centralized resource management in cloud. Furthermore, eHealth cloud can provide more than one interface for users to consume cloud resources, such as smart phone, cloud web portal, tablet, and so on. This well satisfies the ubiquity requirement of cloud.

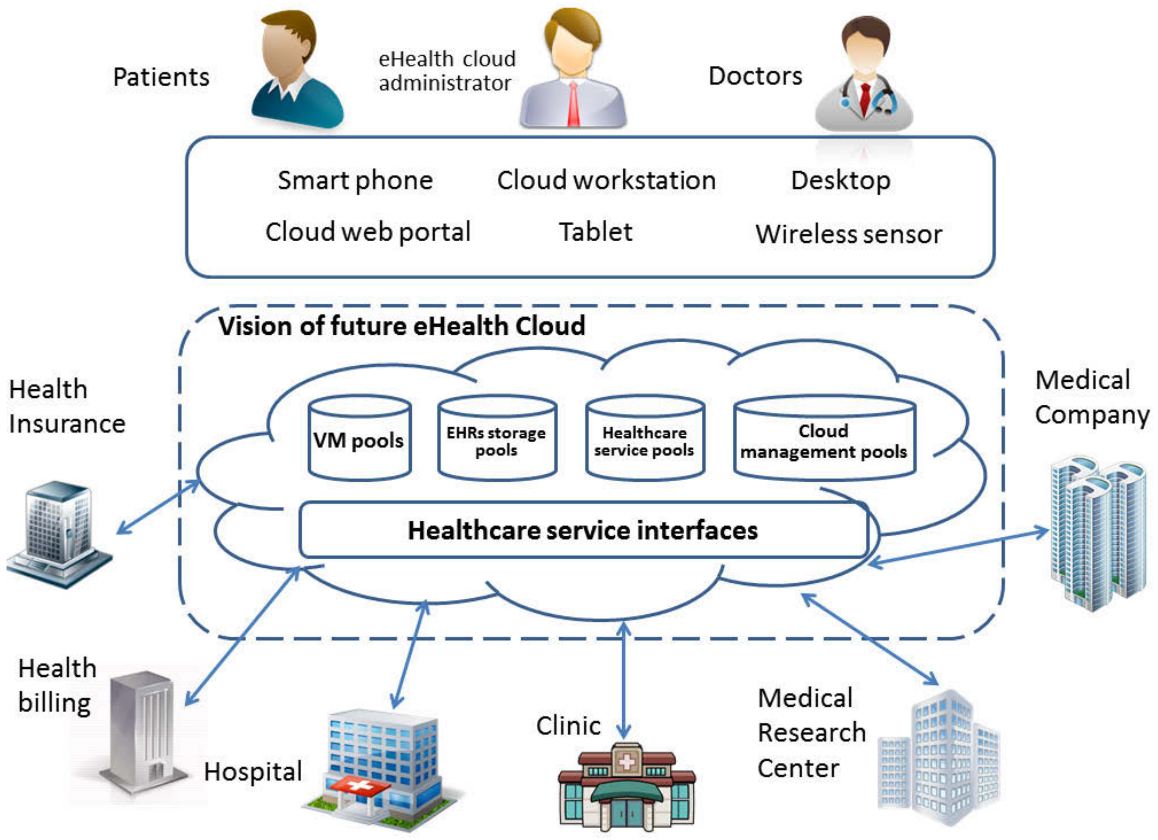


Figure 8.7 Vision of future eHealth cloud

Chapter 9 Conclusion and Future Work

In this chapter we first summarize the thesis and then outline its main contributions. Finally, we suggest future work in this field.

9.1 Summary and Contribution of This Thesis

Cloud data storage faces serious challenges in sensitive industries or organizations where users require more transparency, more trust, and more confidence on their outsourced data. This thesis primarily investigates a trust-oriented data protection framework. The aims and objectives are to mitigate users' concerns and promote broader adoption of cloud-based data storage services. As the research outcome, we propose a framework combining active and passive data protection. The active feature is reflected in an innovative active data-centric framework that is designed to make data active and able independently and autonomously to analyze risks and violations, rather than relying on third party security schemes. The passive protection aspects mainly involve the actions and mechanisms taken to monitor and audit data manipulation and status in the outer environment of the active data-centric framework. We also discuss full mobility management and data replica management to further connect users' data with its owner and to guarantee its availability.

The novelty of this work lies in protecting cloud data in an ADCu that is not highly reliant on strong encryption schemes and third-party protection schemes. By proposing innovative structures, concepts, algorithms, and services, the major contribution of this thesis is that it helps cloud providers to deliver trust actively to cloud users, and encourages broader adoption of cloud-based solutions for data storage services in sensitive areas.

The research contribution of this thesis can be summarized as follows.

- We comprehensively and systematically investigated issues of outsourced data security, privacy, and control in cloud. While analyzing and researching related work on these issues, we propose a novel data protection model adapted to cloud. This model is trust-oriented and can establish more trustworthy relationships between cloud providers and users.

- We proposed an innovative ADC framework that achieves data security and privacy protection with platform-independent and multi-tenant support. This new framework changes the conventional data protection pattern that is highly reliant on third party protection mechanisms and strong encryption.
- We fulfilled ubiquitous monitoring and tracking of data usage based on the ADC framework with light workload but high efficiency. Data operations are processed inside ADCu rather than being executed by a third party interface that can guarantee minimum information disclosure of data.
- We situate two vital services functioning on the proposed ADC framework: a cloud privacy-aware role-based access control (CPRBAC) service and an active auditing service (AAS). The former governs the access control operations on given active data that meet the requirement of cloud data access. The latter executes active auditing of users' data operations, collaborating with the CPRBAC service to maintain consistency and security of data manipulation.
- We also comprehensively investigated and discussed full mobility management related to the ADCu. In our framework, active data can maintain a bundle relationship with users via a mobile device that can ubiquitously receive alert messages generated from an ADCu when violation occurs. Moreover, adaptive data replica management is proposed to address three challenges to satisfying the availability of cloud data: the degree of data replicas, the distribution of data replicas, and the consistency of data replicas.
- Experimental results generated on a real cloud test bed demonstrated the efficiency, dependability, and scalability of the proposed data protection framework. Although a reasonable overhead may be incurred during data structure and operation, the portable, active and independent features of the ADCu can provide more trustworthy data storage status than conventional mechanisms.
- Finally, we conducted a case study of a real application of protecting EHRs in the eHealth cloud, demonstrating the practicability of the research outcomes presented in this thesis.

9.2 Future Work

It is of interest to deploy the trust-oriented data protection in cloud via the combination of active and passive mechanisms (ADC framework, CPRBAC service, AAS service, and full mobility and replica management). No matter what happens with users' data, the owner of the data will be informed through the active binding framework. Smart data can even survive on a compromised host. Future research can be conducted in the following aspects:

The current work and experiments in this thesis were based on a laboratory's mini-private cloud platform using an open source cloud toolkit. All the frameworks and security services are stored and deployed within a single cloud cluster. We still need test results from a large-scale and distributed cloud environment. In the future, we intend to test and improve our security infrastructure under public cloud and hybrid cloud which can represent distribution features.

The work in this thesis focuses on application-level data protection mechanisms. We did not discuss protection schemes for lower layers such as VM level, OS level, hypervisor level, and hardware level. These layers may evince different security issues for data protection. The cloud platform is enormous system, and each component can significantly affect data security and privacy. Hence we intend to investigate deeper-layer data security and protection issues.

The work in this thesis proposed a CPRBAC service designed specifically for the cloud access control context. However, we did not investigate its evaluation performance. XACML is a declarative access control policy language on an XML-basis. It can help to meet a fine-grained access control requirement. However, the performance of policy evaluation requires improvement, which can be addressed in the future.

The work in this thesis discussed adaptive data replica management challenges based on the ADC framework. We experimentally tested our strategies and mechanisms on a simulated cloud platform. In the future we can test in a real distributed cloud context.

The current work makes assumptions that CPRBAC services, AAS services, and supervisors are trustworthy, and we did not discuss how to guarantee the trust and security of these security components. In future, such work can be a new direction.

The current work is protecting unstructured data in the cloud environment. In future, we will specially focus on health data records. We will investigate existing types, sources, formats of various types of EHRs in order to deal with them effectively. We will examine and consolidate the requirements of an atomic framework for the protection of an EHR fully.

More importantly, we will focus on the management of big data health records. The problems of big data will be investigated: why it is beyond the ability of existing and commonly used software tools, storage and computational platforms to capture, manage and process within a reasonable amount of time; how EHRs should be organised and distributed for efficient storage, fast processing and processing in real-time; what distributed infrastructure are appropriate; and how big data centres such as Google and Amazon develop their infrastructure for their ultra-scale services.

References

ABADI, D. J. 2009. Data Management in the Cloud: Limitations and Opportunities. *IEEE Computer Society Technical Committee on Data Engineering,* 32**,** 3-12.

ABAWAJY, J. Establishing Trust in Hybrid Cloud Computing Environments. IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 16-18 November 2011. 118-125.

ALBESHRI, A., BOYD, C. & NIETO, J. G. GeoProof: Proofs of Geographic Location for Cloud Computing Environment. 2012 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW), 18-21 June 2012. 506-514.

ALHAQBANI, B. & FIDGE, C. Access control requirements for processing electronic health records. Business Process Management Workshops, 2008. Springer, 371-382.

ALMUTAIRI, A., SARFRAZ, M., BASALAMAH, S., AREF, W. & GHAFOOR, A. 2012. A distributed access control architecture for cloud computing. *IEEE Software,* 29**,** 36-44.

AMAZON. 2013a. *Amazon Elastic Compute Cloud (Amazon EC2)* [Online]. Available: http://aws.amazon.com/ec2/ [Accessed May 6 2012].

AMAZON. 2013b. *Amazon S3 Availability Event: July 20, 2008* [Online]. Available: http://status.aws.amazon.com/s3-20080720.html [Accessed March 4 2013].

AMAZON. 2013c. *Amazon Simple Storage Service (Amazon S3)* [Online]. Available: http://aws.amazon.com/s3/ [Accessed September 23 2012].

ANGIN, P., BHARGAVA, B., RANCHAL, R., SINGH, N., LINDERMAN, M., BEN OTHMANE, L. & LILIEN, L. An Entity-Centric Approach for Privacy and Identity Management in Cloud Computing. 29th IEEE Symposium on Reliable Distributed Systems, 2010. 177-183.

ARAKAWA, Y., MIYAKE, H., YAMAGUCHI, Y., TAGASHIRA, S. & FUKUDA, A. Application-Layer Active Wireless Network Switching on a Smartphone. International Workshop on Smart Mobile Applications, 2012 Newcastle, UK. ACM.

ARRINGTON, M. 2006. *Gmail Disaster: Reports Of Mass Email Deletions* [Online]. Available: http://techcrunch.com/2006/12/28/gmail-disaster-reports-of-mass-email-deletions/ [Accessed June 20 2012].

ASHLEY, P., HADA, S., KARJOTH, G., POWERS, C. & SCHUNTER, M. 2003. Enterprise privacy authorization language (EPAL 1.2). *Submission to W3C.*

ASSOCIATION, A. I. I. 2012. *AIIA and eHealth* [Online]. Available: http://www.aiia.com.au/?page=eHealth [Accessed August 20 2013].

ATENIESE, G., BURNS, R., CURTMOLA, R., HERRING, J., KISSNER, L., PETERSON, Z. & SONG, D. Provable data possession at untrusted stores. 14th ACM conference on Computer and communications security, 2007 Alexandria, Virginia, USA. ACM, 598-609.

AX2E. 2013. *Java implementation of OASIS XACML 2.0 standard* [Online]. Available: https://code.google.com/p/ax2e/ [Accessed September 2 2013].

BARTH, A., MITCHELL, J. C. & ROSENSTEIN, J. Conflict and combination in privacy policy languages. 2004 ACM workshop on privacy in the electronic society, 2004. ACM, 45-46.

BENSON, K., DOWSLEY, R. & SHACHAM, H. Do you know where your cloud files are? 3rd ACM workshop on cloud computing security workshop, 2011 Chicago, Illinois, USA. 2046677: ACM, 73-82.

BLAISDELL, R. 2012. *Multi–tenancy in the cloud: Understanding its benefits* [Online]. Available: http://www.cloudtweaks.com/2012/06/multi-tenancy-in-the-cloud-understanding-its-benefits/ [Accessed March 27 2013].

BONVIN, N., PAPAIOANNOU, T. G. & ABERER, K. Dynamic cost-efficient replication in data clouds. 1st workshop on Automated control for datacenters and clouds, 2009. ACM, 49-56.

BONVIN, N., PAPAIOANNOU, T. G. & ABERER, K. A self-organized, fault-tolerant and scalable replication scheme for cloud storage. 1st ACM symposium on Cloud computing, 2010. ACM, 205-216.

BOWERS, K. D., JUELS, A. & OPREA, A. HAIL: a high-availability and integrity layer for cloud storage. 16th ACM conference on Computer and communications security, 2009a Chicago, Illinois, USA. ACM, 187-198.

BOWERS, K. D., JUELS, A. & OPREA, A. Proofs of retrievability: theory and implementation. 2009 ACM workshop on Cloud computing security, 2009b Chicago, Illinois, USA. ACM, 43-54.

BUYYA, R., CALHEIROS, R. N. & GROZEV, N. 2013. *CloudSim: A Framework For Modeling And Simulation Of Cloud Computing Infrastructures And Services* [Online]. Available: http://www.cloudbus.org/cloudsim/ [Accessed December 23 2012].

CHEN, L. & HOANG, D. Active data-centric framework for data protection in cloud environment. 23rd Australasian Conference on Information Systems (ACIS 2012 : Location, location, location), 2012 Geelong, Vic., 1-11.

CHEN, Y., PAXSON, V. & KATZ, R. H. 2010. What Is New About Cloud Computing Security? EECS Department, University of California, Berkeley.

CHERVENAK, A., DEELMAN, E., FOSTER, I., GUY, L., HOSCHEK, W., IAMNITCHI, A., KESSELMAN, C., KUNSZT, P., RIPEANU, M. & SCHWARTZKOPF, B. Giggle: a framework for constructing scalable replica location services. 2002 ACM/IEEE conference on Supercomputing, 2002. 1-17.

CHRISTENSEN, J. H. Using RESTful web-services and cloud computing to create next generation mobile applications. 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications, 2009 Orlando, Florida, USA. 1639958: ACM, 627-634.

CISCO. 2013. *McMaster University and Cisco Collaborate to Further Bioinformatics Research and Build Institutional Research Cloud* [Online]. Available: http://newsroom.cisco.com/press-release-content?type=webcontent&articleId=1181571 [Accessed December 14 2012].

COLLBERG, C. S. & THOMBORSON, C. 2002. Watermarking, tamper-proofing, and obfuscation-tools for software protection. *IEEE Transactions on Software Engineering,* 28**,** 735-746.

CONG, W., KUI, R., WENJING, L. & JIN, L. 2010. Toward publicly auditable secure cloud data storage services. *IEEE Network,* 24**,** 19-24.

COPPEREGG. 2012. *Cloud computing holds key to lower carbon emissions* [Online]. Available: http://copperegg.com/cloud-computing-holds-key-to-lower-carbon-emissions/ [Accessed August 20 2013].

CORPORATION, I. 2009. The benefits of cloud computing: a new ear of responsiveness, effectiveness and efficiency in IT service delivery. New YorkIBM corporation.

DANWEI, C., XIULI, H. & XUNYI, R. 2009. Access Control of Cloud Service Based on UCON. *In:* JAATUN, M., ZHAO, G. & RONG, C. (eds.) *Cloud Computing.* Springer Berlin / Heidelberg.

DEAN, J. & GHEMAWAT, S. 2008. MapReduce: simplified data processing on large clusters. *ACM Communications* 51**,** 107-113.

DECANDIA, G., HASTORUN, D., JAMPANI, M., KAKULAPATI, G., LAKSHMAN, A., PILCHIN, A., SIVASUBRAMANIAN, S., VOSSHALL, P. & VOGELS, W. Dynamo: amazon's highly available key-value store. 21st ACM Symposium on Operating Systems Principles, 2007. 205-220.

DEMCHENKO, Y., CRISTEA, M. & DE LAAT, C. XACML Policy Profile for Multidomain Network Resource Provisioning and Supporting Authorisation Infrastructure. 2009 IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY 2009), 20-22 July 2009. 98-101.

DINH, T. T. A., WENQIANG, W. & DATTA, A. 2012. City on the sky: Extending xacml for flexible, secure data sharing on the cloud. *Journal of Grid Computing,* 10**,** 151-172.

DOLIN, R. H., ALSCHULER, L., BOYER, S. & BEEBE, C. An update on HL7's XML-based document representation standards. American Medical Informatics Association Symposium, 2000. 190-194.

DONOHUE, M. & YPSILANTI, D. 2009. *Cloud Computing and Public Policy* [Online]. Available: http://www.oecd.org/internet/interneteconomy/43933771.pdf [Accessed March 23 2013].

DROPBOX. 2013. *Dropbox* [Online]. Available: https://www.dropbox.com/business?tk=dropbox&ag=home_exp&ad=banner [Accessed September 23 2012].

ERWAY, C., KÜPÇÜ, A., PAPAMANTHOU, C. & TAMASSIA, R. Dynamic provable data possession. 16th ACM conference on Computer and communications security, 2009 Chicago, Illinois, USA. 1653688: ACM, 213-222.

FIAT, A. & SHAMIR, A. How to prove yourself: practical solutions to identification and signature problems. Advances in Cryptology—CRYPTO'86, 1987. Springer, 186-194.

FITZSIMMONS, C. 2013. *IBM cutting up to 1500 jobs in Australia as cloud computing upsets server market* [Online]. Available: http://www.brw.com.au/p/techgadgets/ibm_cutting_upsets_server_jobs_market_pd67SMPbQzIgF4msTKLfSK [Accessed August 26 2013].

FOREMAN, J. 2006. *At risk of exposure* [Online]. Available: http://articles.latimes.com/2006/jun/26/health/he-privacy26 [Accessed July 20 2013].

FOSTER, I., ZHAO, Y., RAICU, I. & LU, S. Cloud Computing and Grid Computing 360-Degree Compared. 2008 Grid Computing Environments Workshop (GCE '08), 2008. 1-10.

GEIER, E. *How to Keep Your PC Safe With Sandboxing* [Online]. Available: http://www.techhive.com/article/247416/how_to_keep_your_pc_safe_with_sandboxing.html [Accessed May 15 2013].

GHEMAWAT, S., GOBIOFF, H. & LEUNG, S.-T. The Google file system. ACM SIGOPS Operating Systems Review, 2003. ACM, 29-43.

GOH, E.-J., SHACHAM, H., MODADUGU, N. & BONEH, D. SiRiUS: Securing remote untrusted storage. The Internet Society (ISOC) Network and Distributed Systems Security (NDSS) Symposium, 2003. Citeseer, 131–145.

GOOGLE. 2013a. *GCM Architectural Overview* [Online]. Google. Available: http://developer.android.com/google/gcm/gcm.html#intro [Accessed November 17 2012].

GOOGLE. 2013b. *Google Cloud Storage* [Online]. Available: https://cloud.google.com/products/cloud-storage [Accessed Feburary 2 2013].

GOOGLE. 2013c. *Google drive* [Online]. Available: https://drive.google.com/?authuser=0#my-drive [Accessed March 23 2013].

GOVERNMENT, A. 2010. *Personally Controlled Electronic Health Records for all Australians* [Online]. Available: http://www.health.gov.au/internet/budget/publishing.nsf/Content/E63FA4D3CB4B1345CA25771E001489BD/$File/hmedia09.pdf [Accessed May 4 2013].

GOYAL, P. & MIKKILINENI, R. Policy-Based Event-Driven Services-Oriented Architecture for Cloud Services Operation Management. 9th IEEE International Conference on Cloud Computing (CLOUD), 21-25 Sept. 2009. 135-138.

GROBAUER, B., WALLOSCHEK, T. & STOCKER, E. 2011. Understanding Cloud Computing Vulnerabilities. *IEEE Security & Privacy,* 9**,** 50-57.

GUILLOU, L. C. & QUISQUATER, J.-J. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. Advances in Cryptology—Eurocrypt'88, 1988. Springer, 123-128.

HAO, J. & CAI, W. Trusted block as a service: Towards sensitive applications on the cloud. 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2011. IEEE, 73-82.

HAQ, I. U., ALNEMR, R., PASCHKE, A., SCHIKUTA, E., BOLEY, H. & MEINEL, C. 2010. Distributed trust management for validating sla choreographies. *Grids and service-oriented architectures for service level agreements.* Springer.

HEALTHVAULT. 2013. *What can you do with HealthVault?* [Online]. Available: https://www.healthvault.com/au/en/overview [Accessed August 18 2013].

HOLFORD, J. W., CAELLI, W. J. & RHODES, A. W. Using self-defending objects to develop security aware applications in Java\&trade. 27th Australasian conference on Computer science, 2004 Dunedin, New Zealand. Australian Computer Society, Inc., 341-349.

HOU, J. & O'BRIEN, D. C. 2006. Vertical handover-decision-making algorithm using fuzzy logic for the integrated Radio-and-OW system. *IEEE Transactions on Wireless Communications,* 5**,** 176-185.

HOWARD, M. 2013. *Threat Modeling* [Online]. Available: http://www.code-magazine.com/articleprint.aspx?quickid=0211091&printmode=true [Accessed October 25 2012].

HUANG, J. & NICOL, D. M. 2013. Trust mechanisms for cloud computing. *Journal of Cloud Computing,* 2**,** 1-14.

HWANG, K., KULKARENI, S. & HU, Y. Cloud security with virtualized defense and reputation-based trust mangement. 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing DASC'09, 2009. IEEE, 717-722.

HYE-KYEONG, K. & SANGKEUN, L. 2010. A Binary String Approach for Updates in Dynamic Ordered XML Data. *IEEE Transactions on Knowledge and Data Engineering,* 22**,** 602-607.

INTERNET-GUIDE. 2011. *Computer Viruses / Virus Guide* [Online]. Available: http://www.internet-guide.co.uk/viruses.html [Accessed September 24 2012].

JIN, J., AHN, G.-J., COVINGTON, M. J. & ZHANG, X. Toward an access control model for sharing composite electronic health record. 4th International Conference on Collaborative Computing, 2008.

JUELS, A. & BURTON S. KALISKI, J. Pors: proofs of retrievability for large files. 14th ACM conference on Computer and communications security, 2007 Alexandria, Virginia, USA. ACM, 584-597.

JUELS, A. & OPREA, A. 2013. New approaches to security and availability for cloud data. *ACM Communications,* 56**,** 64-73.

KALLAHALLA, M., RIEDEL, E., SWAMINATHAN, R., WANG, Q. & FU, K. Plutus: Scalable secure file sharing on untrusted storage. 2nd Conference on File and Storage Technologies (FAST'03), 2003 San Francisco, CA. USENIX, Berkeley, CA, 29–42.

KALLAMBELLA, A. 2006. *Advanced Ant Techniques, Part I* [Online]. Available: http://www.javaranch.com/journal/200603/AntPart1.html [Accessed September 4 2011].

KAMARA, S. & LAUTER, K. 2010. Cryptographic Cloud Storage. *In:* SION, R., CURTMOLA, R., DIETRICH, S., KIAYIAS, A., MIRET, J., SAKO, K. & SEBÉ, F. (eds.) *Financial Cryptography and Data Security.* Springer Berlin Heidelberg.

KO, R. K., JAGADPRAMANA, P. & BU SUNG, L. Flogger: A File-Centric Logger for Monitoring File Access and Transfers within Cloud Computing Environments. 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2011a. 765-771.

KO, R. K. L., LEE, B. S. & PEARSON, S. 2011b. Towards Achieving Accountability, Auditability and Trust in Cloud Computing. *Advances in Computing and Communications.* Springer Berlin Heidelberg.

KOLOVSKI, V., HENDLER, J. & PARSIA, B. Analyzing web access control policies. 16th ACM international conference on World Wide Web, 2007. ACM, 677-686.

KONSTANTINOU, A. V. 2003. *Using RMI over SSL authentication for application-level access control* [Online]. Available: http://www.cs.columbia.edu/~akonstan/rmi-ssl/ [Accessed 13 November 2011].

KOUZNETSOV, P. 2013. *JAD Java Decompiler Download Mirror* [Online]. Available: http://www.varaneckas.com/jad/ [Accessed March 1 2013].

L. KRUTZ, R. & DEAN VINES, R. 2010. *Cloud security: a comprehensive guide to secure cloud computing*, Wiley publishing, Inc.

LEE, B.-D. & WEISSMAN, J. B. 2001. An adaptive service grid architecture using dynamic replica management. *Grid Computing—GRID 2001.* Springer Berlin Heidelberg.

LI, M., YU, S., REN, K. & LOU, W. 2010. Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings. *Security and Privacy in Communication Networks.* Springer.

LI, W. & HOANG, D. A new security scheme for e-health system. International Symposium on Collaborative Technologies and Systems (CTS'09), 2009. IEEE, 361-366.

LI, Z.-R., CHANG, E.-C., HUANG, K.-H. & LAI, F. A secure electronic medical record sharing mechanism in the cloud computing platform. 2011 IEEE 15th International Symposium on Consumer Electronics (ISCE), 2011. IEEE, 98-103.

LIN, D. & SQUICCIARINI, A. Data protection models for service provisioning in the cloud. 15th ACM symposium on access control models and technologies, 2010 Pittsburgh, Pennsylvania, USA. 1809872: ACM, 183-192.

LIU, M., LIN, Z. C., GUO, X. B. & ZHANG, D. K. 2007. Evaluation and improvement of vertical handoff algorithms in heterogeneous wireless networks. *Journal of software,* 18**,** 1652-1659.

LIU, S. Task-role-based access control model and its implementation. 2nd International Conference on Education Technology and Computer (ICETC), 22-24 June 2010. 293-296.

LÖHR, H., SADEGHI, A.-R. & WINANDY, M. Securing the e-health cloud. 1st ACM International Health Informatics Symposium, 2010. ACM, 220-229.

MANIATIS, P., AKHAWE, D., FALL, K., SHI, E., MCCAMANT, S. & SONG, D. Do you know where your data are? Secure data capsules for deployable data protection. 13th Usenix Conference Hot Topics in Operating Systems, 2011.

MARKET, C. C. 2012. *Cloud computing market* [Online]. Available: http://www.cloudcomputingmarket.com/ [Accessed July 23 2012].

MATHER, T., KUMARASWAMY, S. & LATIF, S. 2009. *Cloud security and privacy: An enterprise perspective on risks and compliance*, O'Reilly Media, Inc.

MEDIA, S.-C. *The TPM, Trust and the Cloud: Making Trust Real* [Online]. Available: http://brianberger.sys-con.com/node/1411630/mobile [Accessed May 17 2014].

MELL, P. & GRANCE, T. 2011. The NIST definition of cloud computing. *NIST special publication.*

MIKALSEN, T., TAI, S. & ROUVELLOU, I. 2002. *Transactional attitudes: Reliable composition of autonomous Web services* [Online]. Available: http://xml.coverpages.org/IBM-wstx-WDMS-DSN2002.pdf [Accessed October 10 2012].

MILLER, M. 2008. *Cloud computing: Web-based applications that change the way you work and collaborate online*, Que publishing.

MIRZAEI, N. 2009. *Cloud Computing* [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/ReportNarimanMirzaeiJan09.pdf [Accessed November 13 2012].

MONT, M. C., PEARSON, S. & BRAMHALL, P. Towards accountable management of identity and privacy: Sticky policies and enforceable tracing services. 14th International Workshop on Database and Expert Systems Applications, 2003. IEEE, 377-382.

MORRIS, T. 2011. Trusted Platform Module. *Encyclopedia of Cryptography and Security*, 1332-1335.

MQTT.ORG. 2013. *MQ Telemetry Transport* [Online]. MQTT.org. Available: http://mqtt.org/ [Accessed December 1 2012].

NI, Q., BERTINO, E., LOBO, J., BRODIE, C., KARAT, C.-M., KARAT, J. & TROMBETTA, A. 2010. Privacy-aware role-based access control. *ACM Transactions on Information and System Security (TISSEC),* 13**,** 24.

NTP. 2013. *The Network Time Protocol* [Online]. Available: http://www.ntp.org/ [Accessed November 23 2012].

OPENSTACK. 2011. *Open source software for building private and public clouds* [Online]. Available: http://openstack.org/ [Accessed October 23 2011].

ORACLE. 2011. *The Java Archive (JAR) File Format Using JAR Files: The Basics* [Online]. Available: http://java.sun.com/developer/Books/javaprogramming/JAR/basics/ [Accessed September 12 2011].

OTHMANE, L. B. 2010. *Active Bundles for Protecting Confidentiality of Sensitive Data Throughout Their Lifecycle.* Doctor of Philosophy, Western Michigan University.

OTHMANE, L. B., LILIEN, L., BHARGAVA, B., LINDERMAN, M., RANCHAL, R. & SALIH, R. M. 2010. ABTTP: A TTP-based Prototype for Protecting Confidentiality of Sensitive Data with Active Bundles.

PAHLAVAN, K., KRISHNAMURTHY, P., HATAMI, A., YLIANTTILA, M., MAKELA, J.-P., PICHNA, R. & VALLSTRON, J. 2000. Handoff in hybrid mobile data networks. *IEEE Personal Communications,* 7**,** 34-47.

PARDUCCI, B., LOCKHART, H. & RISSANEN, E. 2010. eXtensible Access Control Markup Language (XACML) Version 3.0.

PAWAR, P., RAJARAJAN, M., NAIR, S. K. & ZISMAN, A. 2012. Trust model for optimized cloud services. *Trust Management VI.* Springer.

PEARSON, S. & CHARLESWORTH, A. 2009. Accountability as a way forward for privacy protection in the cloud. *Cloud computing.* Springer.

PEARSON, S., SHEN, Y. & MOWBRAY, M. 2009. A privacy manager for cloud computing. *Cloud Computing.* Springer.

PETKOVIĆ, M. & JONKER, W. 2007. *Security, privacy and trust in modern data management*, Springer.

PIEPRZYK, J., HARDJONO, T. & SEBERRY, J. 2003. *Fundamentals of Computer Security*, Springer.

PINHEIRO, E., WEBER, W.-D. & BARROSO, L. A. Failure Trends in a Large Disk Drive Population. 5th USENIX Conference on File and Storage Technologies, 2007. 17-23.

PISCOPO, N. 2012. *MaaS (Model as a Service) is the emerging solution to design, map, integrate and publish Open Data* [Online]. Available: http://cloudbestpractices.wordpress.com/2012/10/21/maas/ [Accessed April 08 2013].

POPA, R. A., LORCH, J. R., MOLNAR, D., WANG, H. J. & ZHUANG, L. 2010. Enabling security in cloud storage SLAs with CloudProof. *Microsoft TechReport MSR-TR-2010,* 46**,** 1-12.

POPA, R. A., LORCH, J. R., MOLNAR, D., WANG, H. J. & ZHUANG, L. Enabling security in cloud storage SLAs with CloudProof. 2011 USENIX Annual Technical Conference, 2011 Portland. 355-368.

QU, Y. & XIONG, N. RFH: A Resilient, Fault-Tolerant and High-efficient Replication Algorithm for Distributed Cloud Storage. 41st International Conference on Parallel Processing (ICPP), 2012. IEEE, 520-529.

QUN, N., BERTINO, E., LOBO, J. & CALO, S. B. 2009. Privacy-Aware Role-Based Access Control. *IEEE Security & Privacy,* 7**,** 35-43.

RANCHAL, R., BHARGAVA, B., OTHMANE, L. B., LILIEN, L., KIM, A., KANG, M. & LINDERMAN, M. Protection of Identity Information in Cloud Computing without Trusted Third Party. 29th IEEE Symposium on Reliable Distributed Systems (SRDS), 2010 New Delhi, Punjab, India. IEEE, 368-372.

REISS, C., TUMANOV, A., GANGER, G. R., KATZ, R. H. & KOZUCH, M. A. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. Third ACM Symposium on Cloud Computing, 2012. ACM, 7.

RIES, T., FUSENIG, V., VILBOIS, C. & ENGEL, T. Verification of Data Location in Cloud Networking. 2011 4th IEEE International Conference on Utility and Cloud Computing (UCC), 5-8 December 2011. 439-444.

RISTENPART, T., TROMER, E., SHACHAM, H. & SAVAGE, S. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. 16th ACM conference on Computer and communications security, 2009 Chicago, Illinois, USA. ACM, 199-212.

RUJ, S., NAYAK, A. & STOJMENOVIC, I. Dacc: Distributed access control in clouds. 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2011. IEEE, 91-98.

SAIDANI, O. & NURCAN, S. 2006. FORBAC: A Flexible Organisation and Role-Based Access Control Model for Secure Information Systems. *Advances in Information Systems.* Springer Berlin Heidelberg.

SANDHU, R. S., COYNE, E. J., FEINSTEIN, H. L. & YOUMAN, C. E. 1996. Role-based access control models. *Computer,* 29**,** 38-47.

SANDHU, R. S. & SAMARATI, P. 1994. Access control: principle and practice. *IEEE Communications Magazine,* 32**,** 40-48.

SCHNORR, C. P. 1990. Efficient Identification and Signatures for Smart Cards. *Advances in Cryptology — CRYPTO' 89 Proceedings.* Springer Berlin / Heidelberg.

SCHUBERT, L. & JEFFERY, K. 2012. Advances in clouds. European Union, Tech. Rep.

SHACHAM, H. & WATERS, B. 2008. Compact Proofs of Retrievability. *In:* PIEPRZYK, J. (ed.) *Advances in Cryptology - ASIACRYPT 2008.* Springer Berlin Heidelberg.

SHIREY, R. 2003. *RFC 2828-Internet Security Glossary. 2000* [Online]. Available: http://www.faqs.org/rfcs/rfc2828.html [Accessed March 20 2013].

SHVACHKO, K., KUANG, H., RADIA, S. & CHANSLER, R. The hadoop distributed file system. 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), 2010. IEEE, 1-10.

SONG, D., SHI, E., FISCHER, I. & SHANKAR, U. 2012. Cloud data protection for the masses. *Computer,* 45**,** 39-45.

SQUICCIARINI, A., SUNDARESWARAN, S. & DAN, L. Preventing Information Leakage from Indexing in the Cloud. 3rd International Conference on Cloud Computing (CLOUD), 5-10 July 2010. 188-195.

SQUICCIARINI, A. C., PETRACCA, G. & BERTINO, E. Adaptive data protection in distributed systems. 3rd ACM conference on data and application security and privacy, 2013. ACM, 365-376.

STANOEVSKA-SLABEVA, K., WOZNIAK, T. & RISTOL, S. 2009. *Grid and Cloud Computing: A Business Perspective on Technology and Applications,* Incorporated Springer Publishing Company.

STORER, M. W., GREENAN, K. M., MILLER, E. L. & VORUGANTI, K. POTSHARDS: secure long-term storage without encryption. 2007 USENIX Technical Conference, 2008. USENIX Association, 143-156.

SUNDARESWARAN, S., SQUICCIARINI, A., DAN, L. & SHUO, H. Promoting Distributed Accountability in the Cloud. 2011 IEEE International Conference on Cloud Computing (CLOUD), 4-9 July 2011. 113-120.

SUNDARESWARAN, S., SQUICCIARINI, A. & LIN, D. 2012. Ensuring Distributed Accountability for Data Sharing in the Cloud. *IEEE Transactions on Dependable and Secure Computing,* 9**,** 556-568.

TAKABI, H. & JOSHI, J. B. D. Policy Management as a Service: An Approach to Manage Policy Heterogeneity in Cloud Computing Environment. 45th Hawaii International Conference on System Science (HICSS), 4-7 Jan. 2012. 5500-5508.

TATEBE, O., MORITA, Y., MATSUOKA, S., SODA, N. & SEKIGUCHI, S. Grid datafarm architecture for petascale data intensive computing. 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2002. IEEE, 102-102.

TCG. *Trust Computing Group* [Online]. Available: http://www.trustedcomputinggroup.org/ [Accessed May 16 2014].

VIRVILIS, N., DRITSAS, S. & GRITZALIS, D. 2011a. A Cloud Provider-Agnostic Secure Storage Protocol. *In:* XENAKIS, C. & WOLTHUSEN, S. (eds.) *Critical Information Infrastructures Security.* Springer Berlin Heidelberg.

VIRVILIS, N., DRITSAS, S. & GRITZALIS, D. 2011b. Secure Cloud Storage: Available Infrastructures and Architectures Review and Evaluation. *Trust, Privacy and Security in Digital Business.* Springer Berlin Heidelberg.

WANG, C., WANG, Q., REN, K. & LOU, W. Ensuring data storage security in Cloud Computing. 17th International Workshop on Quality of Service (IWQoS) 13-15 July 2009a. 1-9.

WANG, C., WANG, Q., REN, K. & LOU, W. Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing. 2010 IEEE Proceedings on INFOCOM, 14-19 March 2010. 1-9.

WANG, F., GONG, F., SARGOR, C., GOSEVA-POPSTOJANOVA, K., TRIVEDI, K. & JOU, F. SITAR: a scalable intrusion-tolerant architecture for distributed services. 2003 International Workshop on Information Assurance and Security, 2003. 359-367.

WANG, Q., WANG, C., LI, J., REN, K. & LOU, W. 2009b. Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing. *Computer Security – ESORICS 2009.* Springer Berlin Heidelberg.

WANG, S., AGRAWAL, D. & EL ABBADI, A. A Comprehensive Framework for Secure Query Processing on Relational Data in The Cloud. 8th International conference on Secure data management, 2011 Seattle, WA. 52-69.

WANG, W., LI, Z., OWENS, R. & BHARGAVA, B. Secure and efficient access to outsourced data. 2009 ACM workshop on Cloud computing security, 2009c Chicago, Illinois, USA. ACM, 55-66.

WEBOPEDIA. 2013. *Structured data* [Online]. Webopedia. Available: http://www.webopedia.com/TERM/S/structured_data.html [Accessed December 21 2012].

WEI, L., ZHU, H., CAO, Z., JIA, W. & VASILAKOS, A. V. SecCloud: Bridging Secure Storage and Computation in Cloud. IEEE 30th International Conference on Distributed Computing Systems Workshops (ICDCSW), 21-25 June 2010a. 52-61.

WEI, Q., VEERAVALLI, B., GONG, B., ZENG, L. & FENG, D. CDRM: A cost-effective dynamic replication management scheme for cloud storage cluster. 2010 IEEE International Conference on Cluster Computing (CLUSTER), 2010b. IEEE, 188-196.

WEIL, S. A., BRANDT, S. A., MILLER, E. L., LONG, D. D. E. & MALTZAHN, C. Ceph: a scalable, high-performance distributed file system. 7th symposium on operating systems design and implementation, 2006 Seattle, Washington. USENIX Association, 307-320.

WIKIPEDIA. 2010. *Cloud computing* [Online]. Available: http://en.wikipedia.org/wiki/Cloud_computing [Accessed May 20 2014].

WIKIPEDIA. 2012. *Reverse engineering* [Online]. Available: http://en.wikipedia.org/wiki/Reverse_engineering [Accessed May 7 2014].

WIKIPEDIA. 2013a. *Cloud computing* [Online]. Wikipedia. Available: http://en.wikipedia.org/wiki/Cloud_computing [Accessed June 12 2011].

WIKIPEDIA. 2013b. *Google Drive* [Online]. Available: http://en.wikipedia.org/wiki/Google_Drive [Accessed December 12 2012].

WIKIPEDIA. 2013c. *HTTP Secure* [Online]. Available: http://en.wikipedia.org/wiki/HTTP_Secure [Accessed October 10 2012].

WIKIPEDIA. 2013d. *Mandatory access control* [Online]. Available: http://en.wikipedia.org/wiki/Mandatory_access_control [Accessed November 2012].

WIKIPEDIA. 2013e. *Obfuscation* [Online]. Available: http://en.wikipedia.org/wiki/Obfuscation [Accessed May 1 2013].

WIKIPEDIA. 2013f. *Push technology* [Online]. Wikipedia. Available: http://en.wikipedia.org/wiki/Push_technology [Accessed December 25 2012].

WIKIPEDIA. 2013g. *Steganography* [Online]. Available: http://en.wikipedia.org/wiki/Steganography [Accessed March 23 2013].

WIKIPEDIA. 2013h. *Stress testing* [Online]. Available: http://en.wikipedia.org/wiki/Stress_testing [Accessed May 23 2013].

WIKIPEDIA. 2013i. *XACML* [Online]. Available: http://en.wikipedia.org/wiki/XACML [Accessed September 12 2012].

WIKIPEDIA. 2013j. *XMPP* [Online]. Available: http://en.wikipedia.org/wiki/XMPP [Accessed December 10 2012].

WIKIPEDIA. 2014. *Sandbox (computer security)* [Online]. Available: http://en.wikipedia.org/wiki/Sandbox_(computer_security) [Accessed May 10 2014].

WU, X., LEE, M.-L. & HSU, W. A prime number labeling scheme for dynamic ordered XML trees. 20th International Conference on Data Engineering, 30 March-2 April 2004. 66-78.

XACML. 2011. *OASIS eXtensible Access Control Markup Language (XACML) TC* [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml#CURRENT [Accessed November 13 2011].

XU, Z., XIANLIANG, L., MENGSHU, H. & JIN, W. 2004. A dynamic distributed replica management mechanism based on accessing frequency detecting. *ACM SIGOPS Operating Systems Review,* 38**,** 26-34.

YU, S., WANG, C., REN, K. & LOU, W. Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing. 29th IEEE International Conference on Information Communications (INFOCOM'10), 14-19 March 2010. 534-542

YUAN, E. & TONG, J. Attributed based access control (ABAC) for web services. 2005 IEEE International Conference on Web Services, 2005. IEEE.

YUN, A., SHI, C. & KIM, Y. On protecting integrity and confidentiality of cryptographic file system for outsourced storage. 2009 ACM workshop on Cloud computing security, 2009 Chicago, Illinois, USA. ACM, 67-76.

ZHANG, R. & LIU, L. Security models and requirements for healthcare application clouds. 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD), 2010. IEEE, 268-275.

ZHONG, H., ZHANG, Z. & ZHANG, X. A dynamic replica management strategy based on Data Grid. 9th International Conference on Grid and Cooperative Computing (GCC), 2010. IEEE, 18-23.

ZHOU, W., PIERRE, G. & CHI, C.-H. 2011. CloudTPS: Scalable Transactions for Web Applications in the Cloud. *IEEE Transactions on Services Computing,* 5**,** 525-539