PROTECTION AND EFFICIENT MANAGEMENT OF BIG HEALTH DATA IN CLOUD ENVIRONMENT

Thanh Dat Dang
University of Technology Sydney

A thesis submitted to Faculty of Engineering and Information Technology
University of Technology Sydney
in fulfilment of the requirements for the thesis of
Doctor of Philosophy in Information System

2017

DEDICATION

To my Parents

To my Wife and Kids

To my Parents-in-Law

Thank you for your love and support

CERTIFICATE OF ORIGINAL AUTHORSHIP

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

Signature of Student: _ _____

Date: _____ _____

ACKNOWLEDGMENT

I sincerely express my deepest gratitude to my principal supervisor, Professor Doan B. Hoang, for his supervision and continuous encouragement, enthusiasm, technical comments, and constructive criticism throughout my whole PhD study. His full support, 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 outcomes, 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 the top-up scholarship that was really helpful to support my research and study. I feel so fortunate to have him as my supervisor for the past four years.

I thank Dr. Priyadarsi Nanda for co-supervise me. I am very thankful to him for his valuable feedback into research papers and the thesis. His support enabled me to achieve publications on my research work. I am also very thankful to Dr. Diep Nguyen for his valuable comments and recommendations. His guidance enabled me to establish appropriate investigations for my research.

Special thanks to the Soctrang Teacher Training College, Mekong 1000 project under the Cantho University (CTU) and University of Technology Sydney (UTS) for providing a scholarship and other financial support for my study.

My thanks also go to the staff members research students in the VICS group for their help, suggestions, friendship, and encouragement: special thanks to Eryani Tjondrowalujo, Lingfeng Chen, Fatima Furqan, Noor Faizah Ahmad, Tham Nguyen Thi, Nguyet Pham Thi Minh, Thuy Le Ngoc, Chau Nguyen Thi Minh, and Sarah Farahmandian.

In particular, I give thanks to my mother for her love and everything she has done for me. She is always in my heart and my memory. Without her, this work would not have been possible. I give thanks to my father for your love, encouragement, and patience to help me complete this study. My special thanks go to my brother and sisters, Mau Dang Thanh, Phuong Dang Thi Cam, Han Dang Thi Ngoc, thanks a lot for your inspiration and support. I have been incredibly fortunate to have love and support from such a family. I also thank my parents-in-law, Thoan Nguyen Xuan and Hoa Nguyen Thi, for their support.

I express my gratitude and appreciation to my wife. 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 family, my parents, and parents-in-law for their encouragement, and thank all the people who helped me and contributed to this study.

THE AUTHOR'S PUBLICATIONS

International Conference Publications and Proceedings

HOANG, D. B & DANG, T.D, "Health Data in Cloud Environments", 19th Pacific Asia Conference on Information Systems (PACIS 2015), Singapore, pp. 96-108, 2015.

DANG, T.D, HOANG, D. B & NANDA, P, "Data mobility management model for active data cubes", 2015 IEEE Trustcom/BigDataSE/ISPA, E International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-15), IEEE, Helsinki, Finland, pp. 750-757, 2015.

DANG, T.D & HOANG, D. B, "Data Mobility as a Service", Proceedings of the IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW), IEEE International Conference on Distributed Computing Systems' Second IEEE International Workshop on Security Testing and Monitoring, IEEE, Nara, Japan, pp. 67-71, 2016.

DANG, T.D, HOANG, D. B & NANDA, P, "A novel Hash-Based File Clustering scheme for efficient distributing, storing and retrieving of large scale Health Records", 15th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-16), Tianjin, China, 23 Aug 2016 - 26 Aug 2016., IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom), CPS, IEEE Computer Society, Tianjin, China, pp. 1485-1491, 2016.

DANG, T.D & HOANG, D. B "A data protection model for fog computing", 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC), Valencia, Spain, pp. 32-38, 2017.

DANG, T.D & HOANG, D. B, "FBRC: Optimization of task scheduling in Fog-based Region and Cloud", Proc. of the 16th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-17) Sydney, Australia, pp. 1109-1114, 2017.

CHAU, N, HOANG, D. B & DANG, T.D, "Toward a Programmable Software-Defined IoT

Architecture for Sensor Service Provision On Demand", Proc. of 27th International Conference on Telecommunication Networks and Applications (ITNAC 2017), Melbourne, Australia, November 22-24, 2017.

CHAU, N, HOANG, D. B & DANG, T.D, "A Software-Defined Model for IOT Clusters Enabling Applications on Demand", to appear in the Proc. of the 32nd International Conference on Information Networking (ICOIN 2018), Chiang Mai, Thailand, January 10-12, 2018.

Book chapters

DANG, T.D, HOANG, D. B & NANDA, P, "Data Protection and Mobility Management for Cloud" in Kumar, V., Ko, R. & Chaisiri, S. (eds), Data Security in Cloud Computing, The Institution of Engineering and Technology, 2016

International Journals

DANG, T.D, HOANG, D. B & DIEP, NGUYEN, "A Trust-based Scheduling Model for Big Data Processing with MapReduce", IEEE Transaction on Service Computing, *Under submission*.

ABSTRACT

Healthcare data has become a great concern in the academic world and in industry. The deployment of electronic health records (EHRs) and healthcare-related services on cloud platforms will reduce the cost and complexity of handling and integrating medical records while improving efficiency and accuracy. To make effective use of advanced features such as high availability, reliability, and scalability of Cloud services, EHRs have to be stored in the clouds. By exposing EHRs in an outsourced environment, however, a number of serious issues related to data security and privacy, distribution and processing such as the loss of the controllability, different data formats and sizes, the leakage of sensitive information in processing, sensitive-delay requirements has been naturally raised. Many attempts have been made to address the above concerns, but most of the attempts tackled only some aspects of the problem. Encryption mechanisms can resolve the data security and privacy requirements but introduce intensive computing overheads as well as complexity in key distribution. Data is not guaranteed being protected when it is moved from one cloud to another because clouds may not use equivalent protection schemes. Sensitive data is being processed at only private clouds without sufficient resources. Consequently, Cloud computing has not been widely adopted by healthcare providers and users. Protecting and managing health data efficiently in many aspects is still an open question for current research.

In this dissertation, we investigate data security and efficient management of big health data in cloud environments. Regarding data security, we establish an active data protection framework to protect data; we investigate a new approach for data mobility; we propose trusted evaluation for cloud resources in processing sensitive data. For efficient management, we investigate novel schemes and models in both Cloud computing and Fog computing for data distribution and data processing to handle the rapid growth of data, higher security on demand, and delay requirements.

The novelty of this work lies in the novel data mobility management model for data protection, the efficient distribution scheme for a large-scale of EHRs, and the trust-based scheme in security and processing. The contributions of this thesis can be summarized according to data security and efficient data management.

On data security, we propose a data mobility management model to protect data when it is stored and moved in clouds. We suggest a trust-based scheduling scheme for big data

processing with MapReduce to fulfil both privacy and performance issues in a cloud environment.

- The data mobility management introduces a new location data structure into an active data framework, a Location Registration Database (LRD), protocols for establishing a clone supervisor and a Mobility Service (MS) to handle security and privacy requirements effectively. The model proposes a novel security approach for data mobility and leads to the introduction of a new Data Mobility as a Service (DMaaS) in the Cloud.

- The Trust-based scheduling scheme investigates a novel composite trust metric and a real-time trust evaluation for cloud resources to provide the highest trust execution on sensitive data. The proposed scheme introduces a new approach for big data processing to meet with high security requirements.

On the efficient data management, we propose a novel Hash-Based File Clustering (HBFC) scheme and data replication management model to distribute, store and retrieve EHRs efficiently. We propose a data protection model and a task scheduling scheme which is Region-based for Fog and Cloud to address security and local performance issues.

- The HBFC scheme innovatively utilizes hash functions to cluster files in defined clusters such that data can be stored and retrieved quickly while maintaining the workload balance efficiently. The scheme introduces a new clustering mechanism in managing a large-scale of EHRs to deliver healthcare services effectively in the cloud environment.

- The trust-based scheduling model uses the proposed trust metric for task scheduling with MapReduce. It not only provides maximum trust execution but also increases resource utilization significantly. The model suggests a new trust-oriented scheduling mechanism between tasks and resources with MapReduce.

- We introduce a novel concept “Region” in Fog computing to handle the data security and local performance issues effectively. The proposed model provides a novel Fog-based Region approach to handle security and local performance requirements.

We implement and evaluate our proposed models and schemes intensively based on both real infrastructures and simulators. The outcomes demonstrate the feasibility and the efficiency of our research in this thesis. By proposing innovative concepts, metrics, algorithms, models, and services, the significant contributions of this thesis enable both healthcare providers and users to adopt cloud services widely, and allow significant improvements in providing better healthcare services.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AAC	Active Auditing Control
ABE	Attribute Based Encryption
ADC	Active Data-centric
ADCu	Active Data Cube
AES	Advanced Encryption Standard
BGTSMR	Bipartite Graph Trust-based Scheduling MapReduce
CPRBAC	Cloud-based Privacy-aware Role-Based Access Control
CSCs	Cloud Service Customers
CSPs	Cloud Service Providers
DMM	Data Mobility Management
DMaaS	Data Mobility as a Service
EHRs	Electronic health records
FPRBAC	Fog-based Privacy Role Based Access Control
FBRC	Fog-based Region and Cloud
GCM	Google Cloud Message
HBFC	Hash-Based File Clustering
HIS	Health Information Systems
ILP	Integer Linear Programming
LRD	Location Registration Database

MS	Mobility service
P2P	Peer to Peer
RBTA	Region-Based Trust-Aware
RMD	Recordable Mobility Data
SecLAs	Security level agreements
SLAs	Service Level Agreements
SLOs	Service Level Objectives
TDFS	Triggerable Data File Structure
TSMR	Trust-based Scheduling MapReduce
VLRD	Visitor Location Register Database
VM	Virtual Machine

CHAPTER 1 INTRODUCTION

Cloud computing has become increasingly attractive as an efficient computing infrastructure for managing IT system (Schubert and Jeffery, 2012, Krutz and Vines, 2010) with the rapid migration of both services and applications from users own infrastructures to cloud infrastructure. An increasing number of business customers are shifting their services and applications to Cloud computing since they do not need to invest in their own costly IT infrastructure but can delegate and deploy their services efficiently to Cloud vendors and service providers. Recently, Oracle opened the second local data center in Australia to provide Cloud-based services which meet the increasing demand from customers. As a result, more and more data is generated and stored within IT systems. Electronic health record systems store healthcare information about an individual's lifetime digitally with the purpose of supporting continuity of care, education, and research, and ensuring confidentiality at all times (Iakovidis, 1998). The process of provisioning healthcare involves massive healthcare data which exists in different forms (structured files or unstructured data) on disparate data sources (relational databases, file servers, etc.) and in different formats (text, images, sensor data, XML files, relational database records). A patient often receives medical treatment from different health professionals in various organizations over his/her lifetime, and their health data are stored in different Health Information Systems (HIS) and can be shared with health care providers, insurance practitioners, researchers and family members. It is inevitable that EHRs and healthcare-related services will be deployed on cloud platforms to reduce the cost and complexity of handling medical records while improving efficiency and accuracy. Efficient schemes and models for protecting, storing, distributing and processing EHRs are, thus, extremely important as they allow significant improvements in providing better healthcare services particularly in the big health data distributed in cloud environments (Bamiah et al., 2012, Bahga and Madisetti, 2013).

However, handling a large amount of health data in a cloud environment, where data is stored in distributed storage and resided in various organizations such as a hospital, pharmacy, insurance, becomes the biggest obstacle to widespread adoption of cloud computing technology (Bamiah et al., 2012, Bahga and Madisetti, 2013, Ahmed and Abdullah, 2011, Tancer and Varde, 2012). As data is transferred to the cloud, data owners are concerned about the loss of

control of their data and Cloud Service Providers (CSPs) are concerned about their ability to protect data effectively when it is moved about both within and out of its own environment (Foster et al., 2008, Zhang et al., 2011, Tan et al., 2012). Users may not know where, when, how, and by whom or what data operations were executed on their data at CSPs (Ko et al., 2011a, Benson et al., 2011, Ries et al., 2011, Massonet et al., 2011). Moreover, distributed storage systems such as Hadoop (Tantisiriroj et al., 2011) and Peer to Peer (P2P) systems (DeCandia et al., 2007, Lakshman and Malik, 2010) are not sufficient to process a large number of files with various data sizes and structures (Dong et al., 2012). Furthermore, processing the amount of sensitive data has become an issue for the MapReduce model since the system may incur bottlenecks or longer delay when applying third party authorizing or when dealing with large amounts of data at private clouds. The key to mitigating the users and CSPs' concerns and encouraging broad adoption of cloud computing in heath data management is the establishment of a data mobility management model, a data distribution scheme, a trust-based task scheduling for big data, and data protection in Fog computing for CSPs (Ko et al., 2015, Ko et al., 2011b, Ko et al., 2011c, Chen, 2014).

Many attempts have been carried out to address the above concerns, but many issues are still unresolved:

- Data protection mechanisms such as encryption, policy-base authentications are employed and can resolve the data privacy and security requirements at a dedicated cloud. They are, however, unable to meet these requirements at another cloud due to the implementings of different protection schemes and the computing overheads of the encryption algorithm, and the complexity in key distribution (Foster et al., 2008, Juels and Oprea, 2013).
- As data is allocated in the cloud, the loss of control of their data has become the main concern of both data owners and cloud service providers (Foster et al., 2008, Zhang et al., 2011, Tan et al., 2012). Athought many security and protection mechanisms have been employed to protect cloud data, data is still exposed to potential disclosures and attacks if it is moved and located at another cloud where there is no equivalent security measure at the visited sites.
- Distributed resource models such as Hadoop and parallel databases are not sufficient in managing a large number of small files. These systems suffer from heavy overhead and inefficient data access pattern due to frequent accessing of metadata and frequent searching and hopping. It is less efficient for the retrieval and lookup process when the

access pattern is skewed, or data is spread to various nodes.

- Using only private clouds to process sensitive data is also another approach to preserve data privacy. It is not desirable, however, to schedule all processing tasks for sensitive data at only private clouds while public clouds can provide adequate protection services but they remain idle. The systems may incur bottlenecks or longer delay when the amount of sensitive data is huge, or data operations are heavily performed at private clouds.
- Within the increasing demand on real-time processing for applications, it becomes a problem for cloud computing to support these applications which require cloud servers to meet their delay requirements (Bonomi et al., 2012). As with all logically centralized resource and service provisioning infrastructures, the cloud does not handle well local issues involving a large number of networked elements such as Internet of Things (IoTs), and it is not responsive enough for many applications that require the immediate attention of a local controller.

Consequently, Cloud computing has not been widely adopted by healthcare providers and users. Protecting and managing health data efficiently in many aspects is still an open question for current research.

The aim of this thesis is to seek for mechanisms to protect data more securely and manage data more effieciently. Thus, this enables CSPs to effectively deliver cloud services to healthcare providers and users. The following factors motivate our work.

Firstly, data needs to be protected at cloud storage. When data is stored in the cloud, CSPs have provided security mechanisms to deal with data protection issues and allay users' concerns about the loss of control of their data. The data mobility management model introduces a new location data structure into an active data framework that not only effectively preserves data privacy but also actively notifies data owners if there are any accesses to the data.

Secondly, CPSs do not provide equivalent protection mechanism. When data is moved among clouds, data may be exposed to potential disclosures and attacks as there is no equivalence security measure at visited sites. The data mobility management model provides a LRD, protocols for establishing a clone supervisor and a MS to handle security and privacy requirements efficiently.

Thirdly, the healthcare providers are facing a massive increase of health data. As EHRs are provisioned in different forms on disparate data sources and different formats, it becomes the

biggest obstacle of current big data models to retrieve and lookup data quickly due to heavy overheads and insufficient data access pattern. The HBFC scheme innovatively utilizes hash functions to cluster files in defined clusters such that data can be stored and retrieved quickly.

Fourthly, sensitive data is processed only by private clouds. The mechanisms to preserve data privacy by processing sensitive data at private clouds are not efficient in terms of performance and resource utilization, especially, when sensitive-data is skewed, or data operations are heavily performed. The proposed model provides the real-time trust evaluation for cloud resources at both private clouds and public clouds. It then provides an optimal schedule for tasks and resources for applications with maximum trust execution.

Finally, CSPs are facing the increase of sensitive-latency response, location awareness applications. CPSs can improve their cloud service quality by employing the data protection model for Fog computing and a task scheduling scheme for Region-based Fog and Cloud (FBRC). The model introduces a novel concept "Region" to handle the data security and local performance issues efficiently. It provides trust translation among regions, access control, and mobility management to highly protect data. It also introduces an optimal task scheduling mechanism for FBRC.

This chapter outlines key issues of this research and presents the motivation of our work. In addition, it presents the aims and objectives of this thesis. Moreover, it summarizes the main contributions of this research and discusses research model and methodology. Finally, it provides an overview of this thesis.

This chapter is organized as follows. Section 1.1 introduces the key issues of this research. Section 1.2 presents the motivation of the research. Section 1.3 presents the objectives and scope of the research project. In Section 1.4, the research contributions are summarized. Section 1.5 illustrates the significance of this research. Section 1.6 describes the research methodology involved in this thesis. Finally, we present an overview of the thesis.

1.1 The key issues of this research

Cloud computing has become an alternative IT infrastructure where users, infrastructure providers, and service providers all share and deploy resources for their business processes and applications. In order to deliver cloud services cost-effectively, users' data is stored in a cloud where applications are able to perform requests from clients efficiently. As data is transferred to the cloud, data owners are concerned about the loss of control of their data and cloud service

providers are concerned about their ability to protect data when it is moved about both within and out of its own environment (Foster et al., 2008, Zhang et al., 2011, Tan et al., 2012). Many security and protection mechanisms have been proposed to protect cloud data by employing various policies, encryption techniques, and monitoring and auditing approaches. However, data is still exposed to potential disclosures and attacks if it is moved and located at another cloud where there is no equivalent security measure at visited sites.

In a realistic cloud scenario, the delegated cloud can be trusted to handle data of another sub-provider. However, CSPs do not often deploy the same protection schemes. Movement of user's data is an important issue in the cloud, and it has to be addressed to ensure the data is protected in an integrated manner regardless of its location in the environment. The user is concerned whether its data is stored in locations covered by the Service Level Agreements (SLAs) and data operations are protected from unauthorized users. When user's data is moved to data centres located at locations different from its home, it is necessary to keep track of its locations and data operations.

The key research issues that mitigate users' concern and enhance widespread adoption of Cloud computing in healthcare address the following research questions based on our study.

Can the data be protected when allocating at cloud?

Can the proposed data mobility management model guarantee the data is protected when it moves among clouds?

When the data moves from its original cloud to another cloud, the responsibility to protect the data depends on the SLAs between the data owner and its original cloud provider as well as between the original cloud provider and the visited cloud provider. Thus, the equivalent protection scheme needs to be deployed in the visited cloud to protect users' data. Hence, how can this protection be established between the original cloud and the visited cloud confidentially and accountably?

How can I distribute and retrieve large-scale of EHRs quickly?

Several distributed resource models such as Hadoop and parallel databases have been deployed in healthcare-related services to manage EHRs. However, these models are not effective for managing a large number of small files (Dong et al., 2012) and different data structures. Hence, how can we store, distribute and retrieve these health data efficiently in a cloud environment?

How to assign trust values to cloud resources from CSPs?

To convince users to adopt cloud resources provided by CSPs including private clouds

and public clouds, CSPs must provide trusted resources which are sufficient for confidence and transparency to their subscribed users. Hence, what are the trust metrics that CSPs use to assign trust values to their resources?

How to schedule these trust resources to maximize trust gain according to data-sensitive requirements?

It is assumed that the delegated cloud can be trusted at a given security level when processing data of another cloud. CSPs allow users to consume cloud resources as "Pay-As-You-Go" (Schubert and Jeffery, 2012). It should be feasible to assign a measurable metric to cloud resources so that CSPs allocate appropriate secured resources to client's requests. Hence, how can we schedule these resources to maximize the achieved trust value of a task execution?

How to protect the data at an edge network with the tracking and tracing location requirements?

Fog computing preserves many benefits of cloud computing, and it is also in a good position to address these local and performance issues because its resources and specific services are virtualized and located at the edge of the customer premises. However, fog devices frequently change their locations and may request or provide nearby computing resources for faster responses. The security mechanisms are not provided adequately at these locations. Hence, how can we protect the data in such dynamic fog contexts?

How to systematically structure and schedule fog and cloud resources to process the data efficiently at the edge network?

Fog resources are limited and distributed across regions. Structuring and scheduling these resources to provide low latency responses for clients' requests are presented as the challenge. How can we structure and schedule fog and cloud resources efficiently?

By analyzing and answering these research questions, this thesis proposes a novel data mobility model, a novel HBFC Scheme, a trust-based scheduling scheme for big data processing, and a data protection model in Fog computing to protect, distribute and process the data in a cloud environment. The idea is to manage the data efficiently for protection, distribution, and processing. The detailed descriptions of these models and schemes are presented in Chapter 4, Chapter 5, Chapter 6, and Chapter 7.

1.2 Research motivation

Cloud computing has been gaining acceptance across various domains ranging from business

and government to research (Schubert and Jeffery, 2012). In order to make effective use of cloud services, users' data is stored in a cloud where applications are able to perform more cost-effectively requests from clients. Storing data in the cloud leads to the concern about the loss of control of both data owners and cloud service providers (Foster et al., 2008, Zhang et al., 2011, Tan et al., 2012). Although cloud service providers have provided security mechanisms to deal with data protection issues by employing various policy and encryption approaches, data is still exposed to potential disclosures and attacks if it is moved and located at another cloud where there is no equivalence security measure at visited sites (Foster et al., 2008, Zhang et al., 2011, Tan et al., 2012). Consequently, data mobility issues have to be addressed in any data security models for the cloud.

With rapid migration of both cloud services and applications from users' own infrastructures to cloud infrastructure, more and more data is generated and stored within IT systems. A patient often receives medical treatment from different health professionals in various organizations over his/her lifetime and their health data are stored in different HIS and can be shared with health care providers, insurance practitioners, researchers and family members. However, current big data models are not sufficient in handling a large number of files with various data sizes and structures in terms of storing and retrieving. Thus, efficient schemes for storing and retrieving EHRs are thus extremely important as they allow significant improvements in providing better healthcare services particularly in the big health data distributed cloud computing environments.

In a realistic cloud scenario, a CSP can delegate to another CSP to handle its data for business reasons. It is assumed that the delegated cloud can be trusted at a given security level when processing data of another cloud. It should be feasible to assign a measurable metric to cloud resources so that CSPs allocate appropriate secured resources to client's requests. If these CSPs could be evaluated and ranked by third-party ranking systems (Casola et al., 2015) in a measurable security manner, big data applications have options to select more secured resources in processing the data at different CSPs. Furthermore, within the provided trust resources and required trust tasks, a trust-driven scheduler needs to be designed to deliver the highest trust for the execution of big data applications.

Cloud computing has established itself as an alternative IT infrastructure and service model. However, as with all logically centralized resource and service provisioning infrastructures, the cloud does not handle well local issues involving a large number of networked elements and it is not responsive enough for many applications that require the immediate attention of a local

controller. Fog computing preserves many benefits of cloud computing and it is also in a good position to address these local and performance issues because its resources and specific services are virtualized and located at the edge of the customer premise. However, data security is a critical challenge in Fog computing especially when fog nodes and their data frequently move in their environment.

1.3 Research objectives and scope

The research focuses on the following aspects: 1) data protecting issues when storing, moving and processing in the Cloud; 2) performance issues when distributing and processing the data. When data is allocated in the Cloud, it is facing many unknown attacks as illegal disclosure or malicious violation may disclosure sensitive information. When it is moved to another, there is no equivalent protection scheme at the new cloud. When it is processed, tasks associated with the data may not be allocated appropriate trusted resources. Moreover, the increase of data volume and high latency requirements also present a real challenge. The aims of this research are protection and efficient management of Big health data in the cloud. Thus, we investigate the security mechanisms to protect users' data when it is stored and moved in the clouds, explore data management models and algorithms to efficiently store and retrieve a large-scale of EHRs, investigate new trust metric and scheduling schemes to provide high trust execution for big data application, and explore new concepts and architectures to support sensitive-delay applications.

In this thesis, we investigate the nature of EHR including size, structure and distribution. To this end, we proposed schemes and models to efficiently handle data moving in the cloud, data distribution and data processing. As a consequence, This research focuses mainly on big health data and not general big data.

The objectives of this research are examined and identified to fulfil the research gaps as below

- Investigate security models and protocols to handle data protection and the continuity of protection regardless of whether the data is within its home cloud or located at foreign clouds.
- Propose schemes and models that are scalable, robust and reliable to efficiently store, distribute and retrieve a large-scale of EHR.
- Explore new trust metrics that are based on different trust evaluation mechanisms to apply for cloud resources.

- ❑ Achieve high trust execution for sensitive data and increase resource ultilization.
- ❑ Explore new architectures and security models to enable the effective deployment of applications with low-latency requirements, and address data security issues in Fog computing.
- ❑ Implement and evaluate the proposed models and schemes to verify whether they have properties and operations for data mobility, distribution, and processing in both Cloud and Fog environments.
- ❑ To achieve these objectives, we propose a data mobility management model which can actively protect data either at the home cloud side or visited clouds. We introduce a HBFC Scheme to store and retrieve data efficiently. We propose a trust metric for cloud resources and a trust-based scheduling for MapReduce. We propose a data protection model and a Fog and Cloud scheduling scheme to handle security and performance issues in Fog computing.

There are four major outcomes of this project

- The data mobility management model for Active Data Cube (ADCu) to handle data mobility in a cloud environment that involves data moving within and among the original cloud and visited clouds.
- A data distribution scheme that enables storing, distributing and retrieving a large-scale of health data efficiently.
- A novel trust metric that is assigned to cloud resources. A trust-based scheduling scheme for big data processing that allows users' sensitive-data to be executed on high trusted resources.
- A data protection model in Fog computing to provide region-based trust establishment, MS, and Fog-based Privacy Role Based Access Control (FPRBAC). The model enables fog devices in different regions to share and access resources in a secured manner. In fact, the MS enables clients to keep track of changes of data location among regions periodically.

1.4 The significance of the research

Cloud computing today is being leveraged in various aspects of life. Once applications have been deployed in cloud servers, users' data or resources have to be allocated at cloud side in

order to enhance the effectiveness of cloud services. This results in raising users' concerns due to losing controllability of data and lacking appropriate security schemes to prevent unauthorized disclosure of sensitive information. In addition, achieving the close connection between users and data also proposes the increase in quality of cloud services. In fact, despite the trust among clouds, data is still exposed to potential attacks when moving to another cloud. Active data framework not only provides efficient security measures for the data even at visited cloud site but also reduces users' concern about data control and leads to various applications.

- The cost for the deployment of EHR will be reduced. As a result, everyone can use EHR for the management of personal health information.
- Healthcare providers are more assure and see values in adopting Cloud technology.
- A new data mobility model and service is introduced in the cloud for data protection and data mobility.
- A new data distribution scheme is proposed to store and extract big health data efficiently.
- For the first time, a novel trust metric is introduced to assign for cloud resources, which can be used in different systems.
- A trust-based scheduling scheme is introduced based on this trust metric. It establishes models for secure processing in any data processing systems.
- A new data protection model is proposed in Fog computing for data protection and location management.

1.5 Research contribution

The unique contribution is a new approach on data security and data management particularly for EHR. A novel data mobility management model enables EHR data moving among cloud securely.

The research focuses on data arrest, data mobility, data distribution and data in processing for EHRs in a cloud environment. Cloud data can be classified as structured or unstructured data in terms of management type. 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 EHRs are considered unstructured data. Most previous studies, however, have been based on mechanisms for structured data in terms of security and management. Unstructured data strongly relies on third-party security mechanisms or encryption. Once third-party services are compromised, unstructured data would be vulnerable to violation and tampering. It has become clear that more and more we have to deal with bigger and bigger amount of data as well as its mobility within cloud environments. In this work, we concentrate on protection mechanisms for unstructured data. In the following, the major contributions of this study are listed as below:

The comprehensive study related to issues of EHR security, privacy, and management in a cloud environment are investigated. The research provides novel mechanisms and techniques for protecting cloud data within and around Inter-cloud environments, and big health data so that efficient storing, accessing and processing can be performed. A data management mobility model for ADCu, a HBFC scheme for large-scale of EHRs, a trust-based scheduling for big data processing, a data protection in Fog computing for healthcare are proposed.

A novel data mobility management model provides active data protection and the continuity of protecting regardless of whether the data is within its home cloud or is located at foreign clouds. The model relies on an active protection framework and introduces a MS and a new location database service to handle data moving operations. The model allows the establishment of a proxy supervisor to safeguard its associated data in the new environment and introduces a new component of the active data structure to record its own location.

A new DMaaS is proposed to address concerns relating to data control, data protection and data mobility: 1) users may lose control of their resource; 2) data protection schemes are not adequate when data is moved to a new cloud; 3) tracking and tracing changes of data location as well as accountability of data operations are not well supported. The novel cloud service for data mobility operates from two aspects: data mobility and data protection.

A novel HBFC Scheme that efficiently distributes, stores and retrieves EHR efficiently in cloud environments is proposed. The scheme utilizes hashing to distribute files into clusters in a controlled way and it utilizes a P2P structure for data management. EHR files are clustered in a controlled manner into a defined number of nodes to minimize the number of steps in searching for a requested file, resulting in the decrease of data lookup latency among peers. Furthermore, a data replication management based on the HBFC scheme is proposed to address challenges of data availability, workload balance and distributions among clusters are maintained with the number of new replications inserted into the system.

A novel composite trust metric that accommodates a new mechanism, to perform a real-time trust evaluation for cloud resources for big data processing has been designed.

A trust-based task scheduling for big data processing with MapReduce is first investigated. The proposed scheme achieves the highest trust value in scheduling tasks with different required trust levels to trusted resources. Consequently, the scheme provides both CSPs and users with higher gains in terms of revenues.

A novel data protection model in Fog computing is proposed to address the data protection and the performance issues. This study investigates a new concept of "Region" to accommodate location requirements of users, fog nodes, and fog devices. This introduces a thorough protocol to establish trust agreements among regions, a Fog-based Privacy-aware Role Based Access Control for access control at fog nodes, and a mobility management service to handle changes of users and fog devices' locations.

An optimal task scheduling in FBRC, systematically structures, manages and schedules fog and cloud resources for application which are latency-sensitive at the edge network. The scheme provides scheduling scenarios where computation can be processed either at regions and in cloud servers. By obtaining an optimal task schedule at both regions and cloud servers, the computation and transmission latency of all requests can be minimized.

1.6 Research model and methodology

We adopt the research model in this thesis based on a comprehensive software engineering approaches (Adrion, 1993, Moher and Schneider, 1982). The research strategy is comprised of three main stages: problem definition, developing and implementing new approaches/prototypes, and verification and evaluation. The research model is as shown in Figure 1.1.

In term of the research question, we investigate the basic research problem and analyze current approaches. By studying the related works, we identify the requirement and develop new approaches. We next implement the proposed models and schemes to achieve the research results. These results are then validated and evaluated which may lead to final results or a review to refine the design approaches.

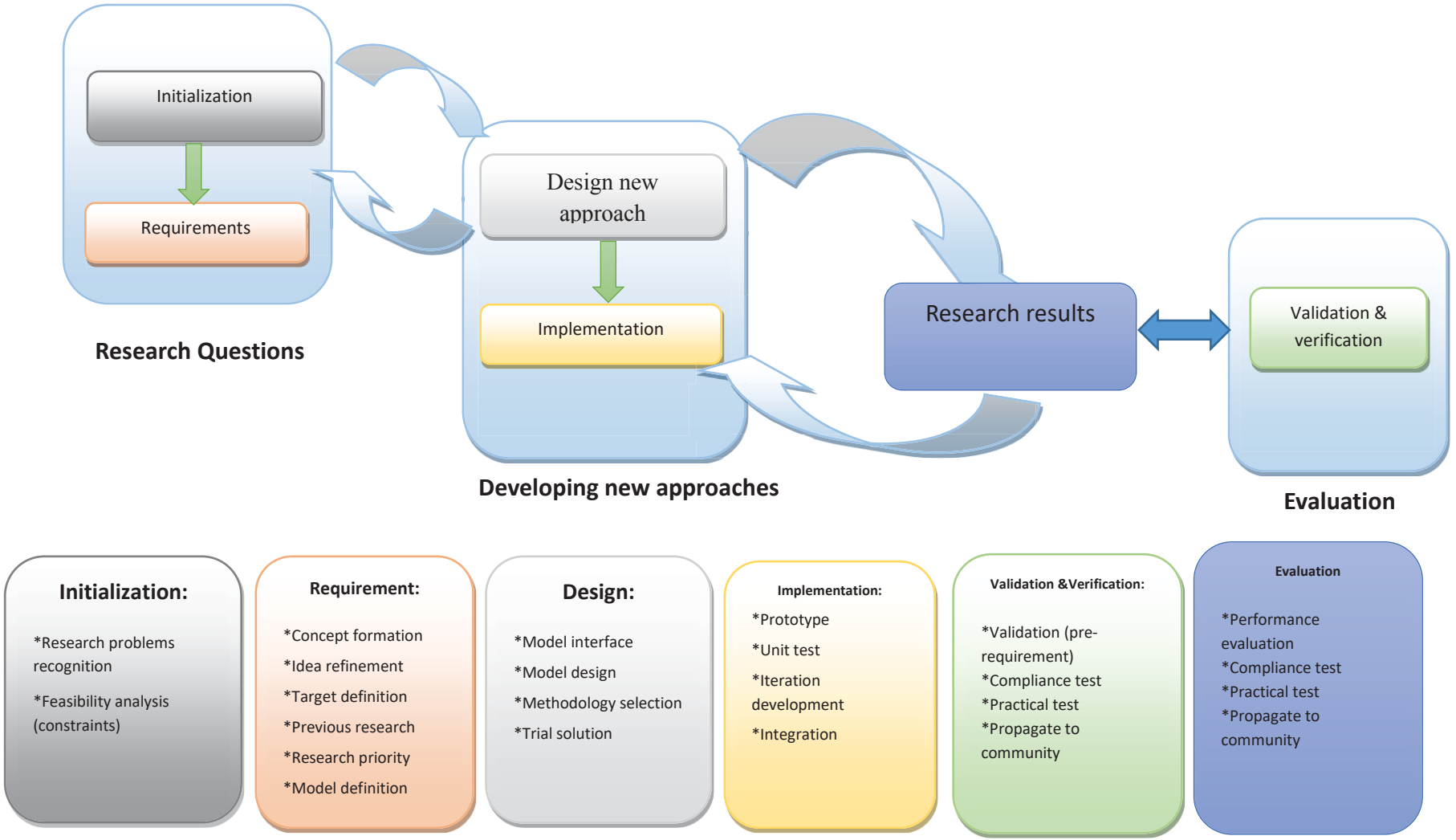


Figure 1.1 Research Model

1.7 Structure of thesis

This research has produced six international conference papers, one book chapter, and a journal paper under review. Figure 1.1 illustrates the structure and organization of this thesis into chapters as follows

Chapter 1 - Introduction

Chapter 1 presents an overview of this research. It provides cloud computing and EHRs definitions, the research questions, research motivation, research aims and objectives, research contribution and significance, and research model and methodology.

Chapter 2 - Literature Review and related work

This chapter presents the literature review on EHR, data security, and big data management. We first introduce EHR definition, structure and then discuss data security, privacy, and mobility issues. In fact, big health data management is also discussed in relation to distribution and processing. We also introduce Fog computing and then discuss data protection and resource scheduling issues in relation to security and performance. Subsequently, we identify and study previous mechanisms to deal with data security, distribution, and processing in a cloud environment. Research challenges and comparisons between our work and the previous studies are also discussed in this chapter.

Chapter 3 – A Trust-oriented data protection framework and management

Chapter 3 provides an overview of our group's work. The chapter introduces briefly our contributions.

Chapter 4 – Data mobility management model for Cloud

Chapter 4 investigates the data mobility model for ADCu. The previous work on ADCu is described. In detail, the assumption of data mobility scenario, the establishment of a supervisor at visited clouds, the proposed data mobility model including components, workflows, implementation, and evaluation are also described in this chapter.

Chapter 5 – Data distribution model and data replication management model

This chapter presents in detail my proposed data distribution model and describes mechanisms applied to store and distribute large-scale of EHRs. The HBFC scheme is one of the main parts of the model to address the problem of data distribution. The data replication based on the

HBFC scheme is also investigated in this chapter.

Chapter 6 – A Trust-based resource scheduling scheme for big data processing

This chapter describes the trust-based scheduling scheme for big data processing. In this chapter, we propose a trust metric to assign to cloud resources. Moreover, we also propose a trust-based scheduling scheme for the task. The details of the proposed scheme, including the problem formulation, the proposed heuristic algorithm, are described. In addition, the simulation results and the significance of trust gain are also discussed in this chapter.

Chapter 7 – A Data protection model and task scheduling scheme for Fog computing

In this chapter, we describe the proof-of-concept prototype of the data protection model to demonstrate the features of fog-based region protection. Furthermore, we propose a task scheduling scheme in FBRC. First, we design a Region-Based Trust-Aware (RBTA) model for trust translation among fog nodes of regions, a Fog-based Privacy-aware Role Based Access Control for access control at fog nodes, and we develop a mobility management service to handle the data protection and the performance issues. Next, we design an efficient task scheduling mechanism to minimize the completion time of tasks by proposing the fog-based region to provide nearby computing resources and a task scheduling scheme in FBRC.

Chapter 7 - Conclusion and Future Works

This chapter summarizes the research findings of the study. The contributions of research expressed in this thesis are summarized in there. Finally, we discuss the future research directions based on the results of this study.

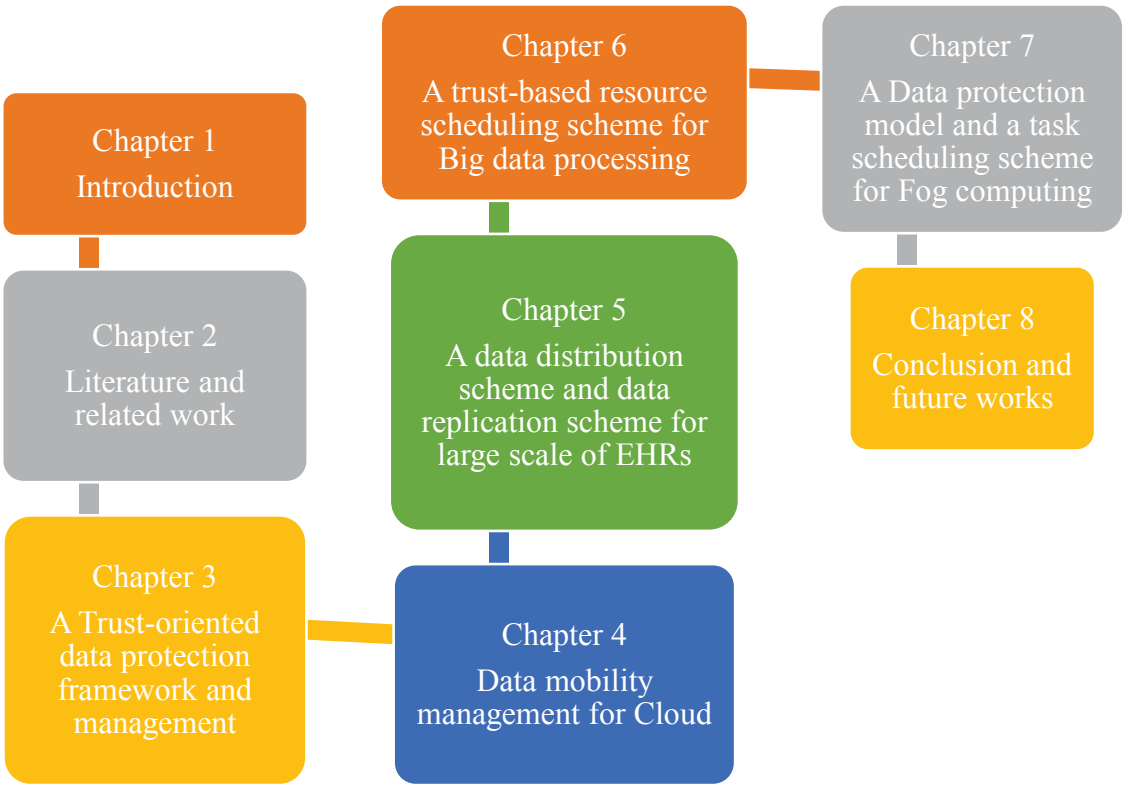


Figure 1.2 Thesis structure

1.8 Summary

This chapter has provided the context and research issues for this thesis. We have identified the key issues of this research and research questions, research motivation. After that, research objective and scope, the significance, the research contribution, research methodology and the contents of each chapter are presented. Based on the investigation about the research issues, Chapter 2 presents a literature survey and works related to the field of the study.

CHAPTER 2 LITERATURE REVIEW AND RELATED WORK

Researchers have proposed various mechanisms and technologies relating to data arrest, data mobility and data processing in cloud computing. This chapter presents a review of the literature directly relevant to the issues being investigated in the thesis. It provides the concepts necessary to understand EHR including definitions, components, and structures. Following that, the concept of cloud computing and its features are presented. Next, the chapter reviews the literature on mechanisms to protect data security and privacy, including policy-based authorization approaches, cryptographic approaches, and data protection approaches. Studies related to big data management models are also presented in the literature, including Hadoop, MapReduce, P2P and distributed databases. Last but not least, we point out the gaps in the existing approaches that are filled in this thesis.

The structure of this chapter is as follows: Section 2.1 to 2.5 introduces the literature of EHR, cloud computing, data security and privacy mechanisms, data mobility in the cloud, and EHR data security and privacy in the cloud. Section 2.6 present big data management models. Section 2.7 focuses on existing mechanisms in data processing. Section 2.8 presents an overview of Fog computing and major data security and privacy challenges in Fog computing. Section 2.8 draws the conclusion of this chapter.

2.1 Electronic Health Record

An electronic health record generally implies information related to medical conditions of a person, but this is not useful for all practical purposes as it is vague and does not provide a complete picture of a person's health conditions and wellness. It is not specific enough to a specific medical condition or treatment. This next section establishes a more comprehensive definition of EHR in terms of its contents and with examples.

Iakovidis (Iakovidis, 1998) defines EHR in a broader term encompassing personal data, care,

and treatments, education and security as “digitally stored health care information about an individual's lifetime with the purpose of supporting continuity of care, education and research, and ensuring confidentiality at all times”.

The Health Information and Management Systems Society (HIMSS) (HIMSS, 2017) defines an electronic health record as, “a longitudinal electronic record of patient health information generated by one or more encounters in any care delivery setting. Included in this information are patient demographics, progress notes, problems, medications, vital signs, past medical history, immunizations, laboratory data, and radiology reports”. The Institute of Medicine (IoM), USA, states that an essential technology for health care is the Computer-based Patient Record (CPR). According to the IoM, “A computer-based patient record (CPR) is an electronic patient record that resides in a system specifically designed to support users by providing accessibility to complete and accurate data, alerts, reminders, clinical decision support systems, links to medical knowledge, and other aids.” (Roberts, 1999). The International Organization for Standardization (ISO) defines EHR as, “a repository of patient data in digital form, stored and exchanged securely, and accessible by multiple authorized users. It contains retrospective, concurrent, and prospective information and its primary purposes are to support continuing, efficient and quality integrated healthcare (Häyrinen et al., 2008).

2.1.1 EHR contents

Explicitly, a useful EHR is composed of five essential components: personal information, medical history reports, health examination, nursing data, and administrative medical and health service records.

- ***Personal information*** presents the patients’ general demographic information such as Patient name, Previously registered name/maiden name, Individual Identifier/medical record number, Universal patient health number, Gender, Race, Address, Telephone number, Date of birth, Organization, Admission Date, Discharge Date, Legal authenticator, Authentication date, Transcriptionist/data enterer and Transcription date.
- ***Medical history reports*** present a whole picture about a patient’s medical history including past medical history, medications, social history, procedure history and allergies.
- ***Health examination*** includes physical examinations, diagnostic findings, assessment

and plan, and Operative Report.

- ***Nursing data*** contains daily charting, physical assessments, and admission nursing notes. Daily charting includes patients' daily functional activities such as vital signs, food, elimination, mobility and patient teaching. Physical assessment comprises all kinds of status assessments. Admission nursing note contains information on allergies, health behavior, physical assessment, discharge planning and initial care plan.
- ***Administrative medical and health service records*** include medical administration information such as admissions records, referrals records, consultation records and bill payer records.

Figure 2.1 shows an EHR sample which is extracted from EHR data (Dolin et al., 2000). The EHR complies with HL7 standard. The purpose of element Header allows the EHR to be exchanged among healthcare providers. The Header presents Personal information such as patient name, individual identifier/medical record number, practitioner ID and practitioner name. The Body is comprised of nursing data and health examination.

```
<?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 2.1. An example of raw EHR

2.1.2 EHR Structures

EHR structures vary greatly since data is recorded in HIS by different groups of health care professionals together with the different legislation, standards and health practices of different countries, states, and health care facilities. Xu et al. (2009) used XML to presents an EHR data structure which is a hierarchical tree structure. The EHR data represented in the tree structure can include patient's basic information, hospital entering records, records of the first disease course, surgical records and records of discharge from hospital (Figure 2.2). The record starts with the root node which could have one or more sub-nodes, and each sub-node serves as one case that presents for a component. The hierarchical tree structure is desirable and flexible in defining structures since it can be described by XML stored by XML database, and allows fast retrieval. (Lj et al. (2000)) also uses XML to present EHR structure at two levels as shown in Figure 2.3. The GEHR Object Model (or GOM) defines the data structures representing the electronic health record.

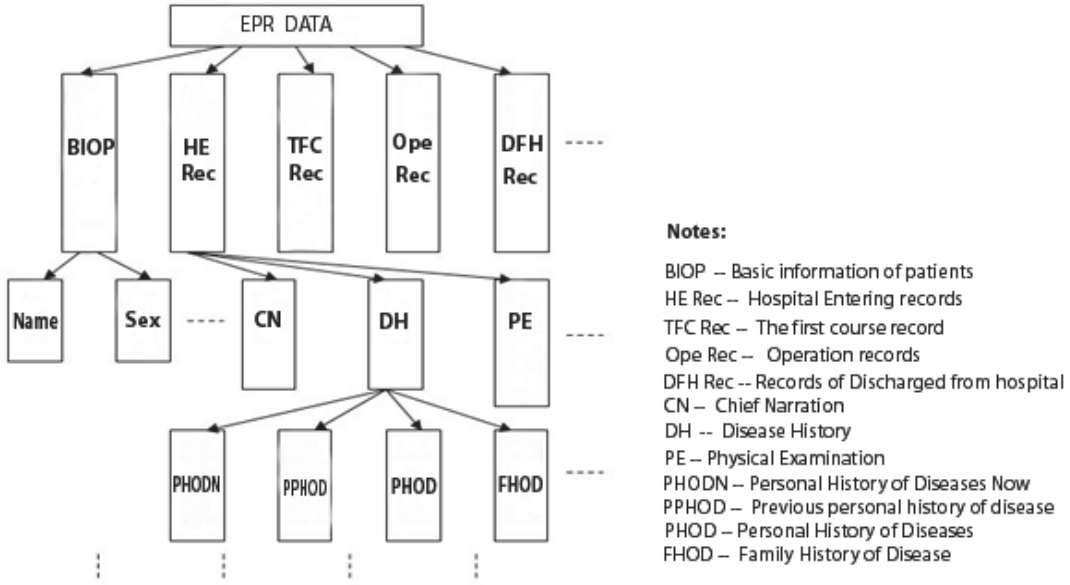


Figure 2.2 Data Structure of Electronic Patient Record (EPR) (Xu et al., 2009)

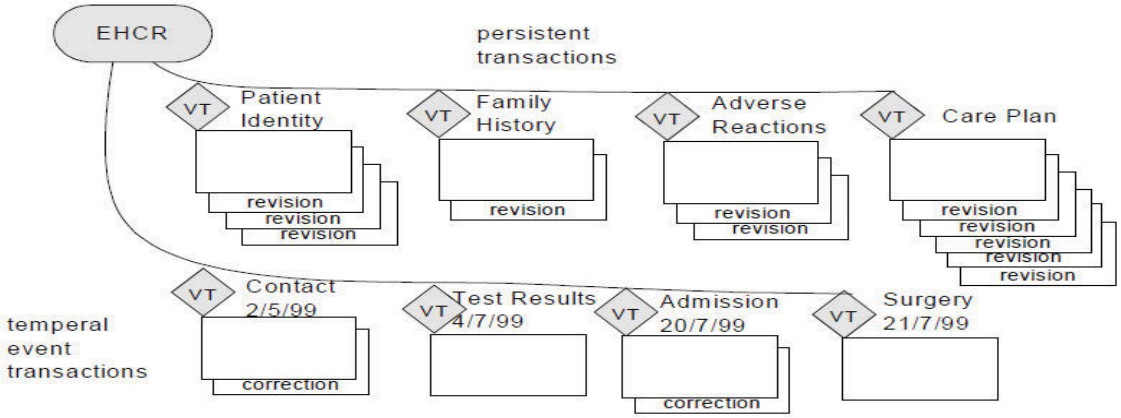


Figure 2.3 Versioned transaction view of the EHR (Lj et al., 2000)

At the root, EHR is a container to store a collection of transactions. Persistent transactions contain information which remains valid in the long term, such as family history, chronic conditions. Event transactions are used for information where the validity is relatively short-lived, such as the test results, a contact with a health care professional or a hospital admission.

2.1.3 EHR formats

Currently, EHR information is stored in various proprietary formats. Typical formats include structured document-based storage in various formats such as database tables and structured XML files, and unstructured document storage in the text, image, or video formats and digitized hardcopies maintained in a typical classical document management system. All possible formats of EHR, however, can be categorized in unstructured data and structured data as shown in Figure 2.4.

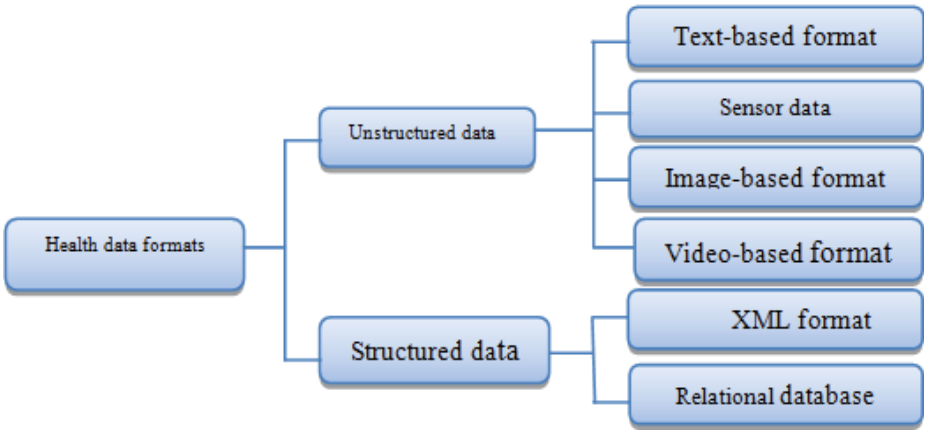


Figure 2.4 Categories of health data formats

2.1.3.1 Unstructured data

Unstructured EHR is an electronic health record which does not follow any patterns and structures. It allows users to store health-related information, such as case histories, health examination results, daily records in unstructured forms such as text, images, sensor data. Using Unstructured EHRs offers users some benefits. Firstly, it supports a variety of file formats which make it possible to store different types of health data such as text, images, and video. Secondly, it enables doctors who are collecting information while interacting with patients. By using special notes, they can write the notes and detailed records when they are diagnosing patients. Lei (Lei et al., 2012) introduced a new de-identification method especially for publishing free text Chinese electronic medical records (EMR) without leaking patients' privacy. A typical EMR template from a Hospital in Beijing is presented as an EMR which is in free text form, but the EMR lists patient information in a relatively formatted pattern.

2.1.3.2 Text-based electronic health record

Unstructured data is commonly used for storing data collected from sensor networks such as text, log files. Moreover, there are many unstructured health data such as radiology reports, discharge summaries which are still stored in a free-text format without any formal convention. Clinical narratives, such as radiology and pathology reports, are a growing electronically available source of information. Clinical texts are commonly dictated and transcribed by a person or speech recognition software or are directly entered in text form by physicians.

a. Image-based electronic health record

Image-based EHR systems can deal with a wide array of medical data types including multi-dimensional medical images personal health records (PHRs) based on the web such as PET-CT, MRI, Xray, medical videos, text-based data and metadata, and supplementary notes. An image-based electronic health record could store a collection of CT or MRT images which is useful for radiologists and other specialists such as surgeons. An imaging report that is widely needed and broad accessibility is vital for achieving optimal patient care. However, each image format defines its own unique metadata format. Existing image metadata fields are limited and not extensible to individual needs. Benjamin Jung (Jung, 2005) proposed the DICOM-X as a framework to prepare medical images for seamless integration into the EHR, providing another step towards the Semantic Health Records (SHR). Michael de Ridder (de Ridder et al., 2013) proposed a novel web-based Personal Health Records (PHRs) visualisation system, called the medical graphical avatar (MGA). The system can deal with a wide array of medical data types

including multi-dimensional medical image PHRs based on the web such as PET-CT, MRI, medical videos, text-based data and metadata, and supplementary notes.

b. Video-based electronic health record

A Dietary Data Recording System (DDRS) which consists of an electronic data collection device and the software and protocols necessary to support data capture and calculation of nutrient intake was proposed in (Junqing et al., 2011). A video camera and a laser-generated grid of distances to food surfaces are employed for calculating the food intake. DDRS includes three components namely the mobile application, the web service, and the volume measurement and uses the client-server model where data is collected by smartphone and then sent to a server via a wireless network.

c. Unstructured Sensor Data

EHR data gathered from different sensors can be stored in files such as text files, CSV files. This creates unstructured sensor files including a very large amount of health records captured from patients such as heart rate and blood pressure. EHR based on sensor data enables collecting patient health information continuously due to the use of sensor devices. Consequently, patients can achieve continuity of care because each health record is analyzed and alerts may be sent to physicians and healthcare providers if there are indications of an emergency.

2.1.3.3 Structured data

Structured EHR is an electronic health record in which health-related information is stored in structured forms. In this category, medical information is collected using designated forms and interfaces which make it possible to index and search information quickly. It can be easily converted to different formats allowing the exchanging of information.

a. XML file format

Extensible Markup Language (XML) is a simple, flexible and a standard meta-language recommended by the World Wide Web Consortium (W3C) as a mechanism for structuring data for large-scale electronic publishing. It is a system independent and programming language independent description language that is especially suitable to describe any type of data of known or unpredicted data structures. It uses hierarchical, object-oriented structure methods of description, which is well suited to describe the complex contents of medical records. Camous

(Camous et al., 2008) presented the HealthSense prototype for harvesting and integration of physical data from wearable sensors. An example in (Khan et al., 2012) showed that sensors monitoring Alzheimer's patient activities such as motion sensors, video based sensors, wearable sensor based, and location based sensors create data which are stored in a raw sensory data repository or XML repository.

b. Relational database

Relational database is known as the most popular structure to store health records in healthcare systems. Accessing information in the database is through the simple and almost universal structured query language (SQL). Several standard Relational Database Management Systems (RDBMS) using any SQL-based software such as Oracle, Sybase or MySQL are used as databases in HIS. In RDBMS, EHR contents are stored in tables whereas EHR elements are mapped into corresponding fields and data types. In addition, relevant relationships among tables are also created by linking identified fields as primary keys in order to perform queries. Vucetic et al. (2011) designed and developed a health information system using a central database. The scheme describes crucial tables such as Patient, Health_id_card, doctor, patient_alergy, patient_surgery, check. A relational database uses key-based storage which enables fast retrieval by using queries and updating tools. However, using relational database method to store EHR has two major drawbacks: a) relational database cannot cope with frequent changes in the Healthcare Information System as it relies on static designs that do not allow data type extensions and b) the size limit of database tables prevents storage of big data.

2.2 Cloud computing

2.2.1 Definitions and models

The term cloud computing is popularly understood as IT services provided over the internet. Cloud computing has been defined as "a style of computing where users consume scalable IT-related capabilities as a service" by using Internet technology (Mirzaei, 2009). From the service point of view, cloud computing is a deployment model for applications that provide services for users by CSPs. Customers use the web service interfaces to access cloud services and configure service capabilities subscribed to CSPs. A number of researchers and practitioners have attempted to define Clouds in various ways. The most popular definition is probably the one provided by the National Institute of Standards and Technology (NIST) (Mell and Grance,

2011). According to this definition

“CLOUD computing is a model for enabling ubiquitous, convenient, on-demand, network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This CLOUD model is composed of five essential characteristics, three service models (Software/ Platform/ Infrastructure as a Service), and four deployment models, whereas the five characteristics are: on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service. The deployment models include private, community, public and hybrid CLOUD”.

Figure 2.5 presents the general architecture of Cloud computing (Lenk et al., 2009). The service models in cloud computing can be categorized as follow:

- *Infrastructure as a Service (IaaS)*: this provides computing resources such as servers, network capabilites and, storage (e.g. Amazon EC2 and Microsoft Azure).
- *Platform as a Service (PaaS)*: This offer a higher-level environment for developers to customize applications (e.g. Google AppEngine).
- *Software as a Service (SaaS)*: This provides services such as software or applications for customers over Internet (e.g. SalesForce, Google Docs).

More recently, research works introduced new cloud services such as Data as a Service, Data mobility as a Service (Dang and Hoang, 2016), and Security as a Service (Varadharajan and Tupakula, 2014).

Cloud infrastructures can be classified into public, private, or hybrid. Public clouds are deployed to serve the general public or a large company. A Cloud Service Provider owns these clouds and sells cloud services to customers(e.g. Amazon, Google, Microsoft). Private clouds are usually owned and managed by one organization. They provide services that allow enterprise applications running on private, secure off-site data. Hybrid clouds combine both private clouds and public clouds to fulfil issues with the current cloud technologies. Beside services provided, the cloud also offers management tools and facilities such as monitoring, billing, user management. These functions enable users to manage large cloud systems with user-friendly interfaces.

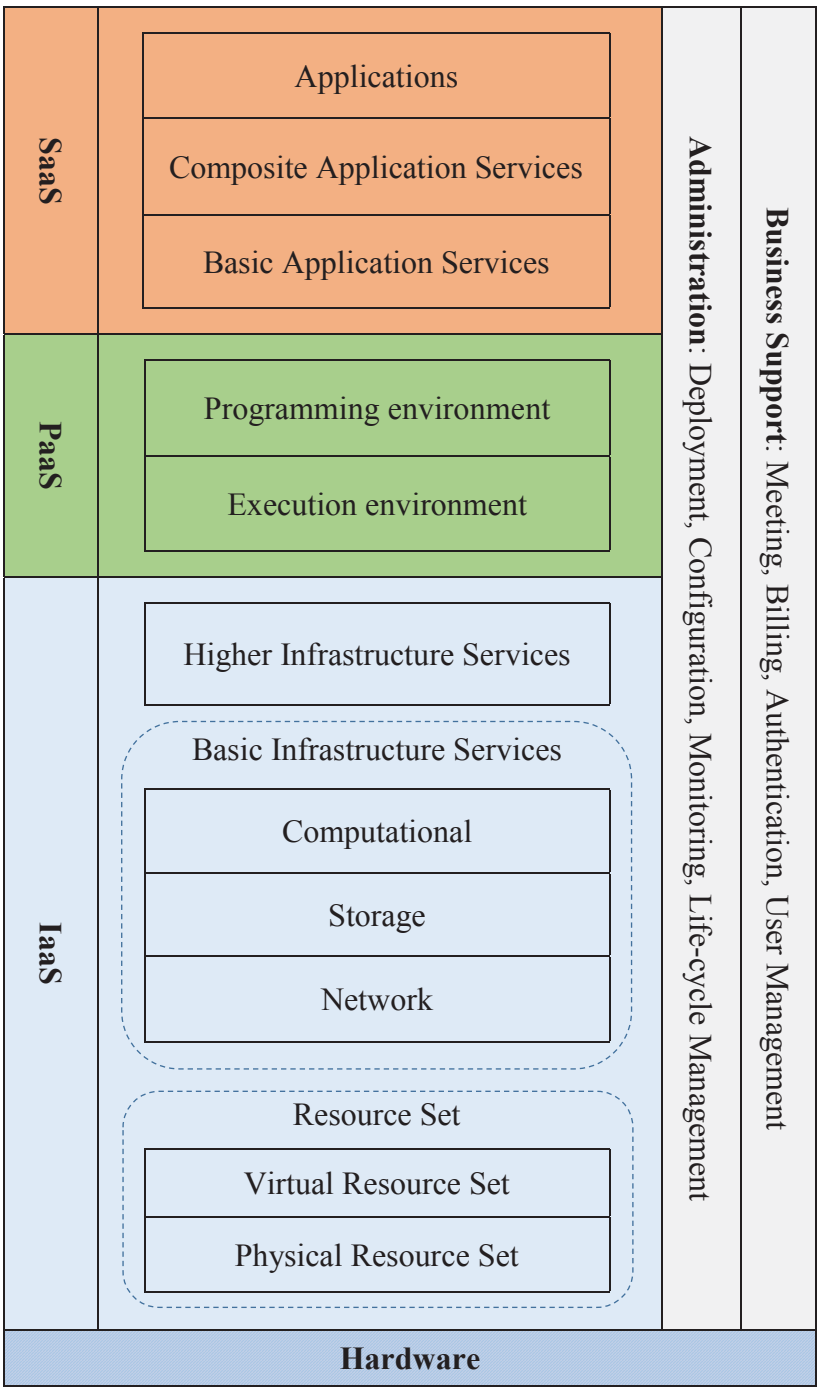


Figure 2.5 Architecture of Cloud computing (Lenk et al., 2009)

2.2.2 Healthcare system in Cloud environments

The Healthcare ecosystem in cloud environment was presented in (Bahga and Madisetti, 2013). The Cloud enables healthcare service providers to house their systems such as EHR systems, HIS, laboratory information system. It also allows patients to share their EHR to other hospitals for admission and care purposes. Figure 2.6 presents a healthcare ecosystem in cloud

environments.

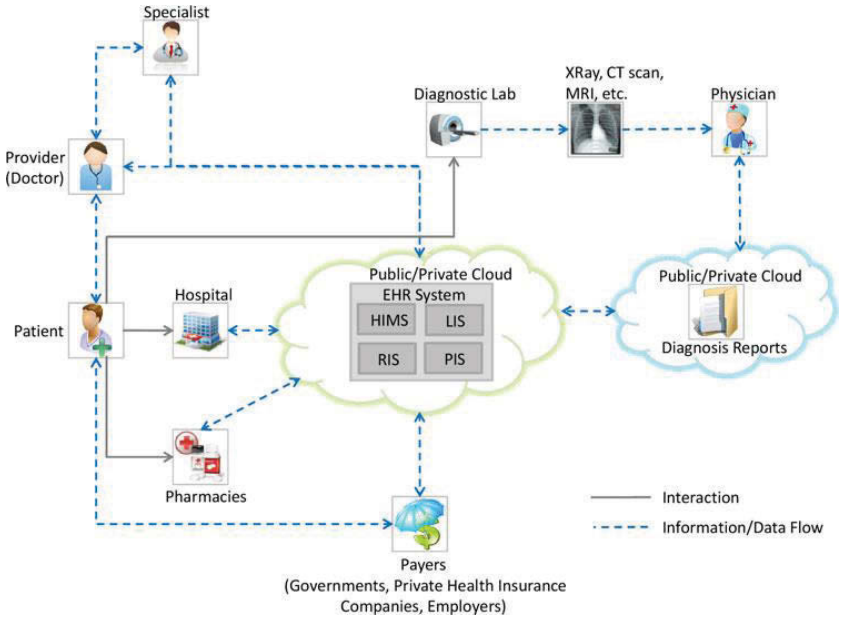


Figure 2.6 Application of cloud computing environments to the healthcare ecosystem (Bahga and Madisetti, 2013)

Bahga (Bahga and Madisetti, 2013) proposed an EHR system—Cloud Health Information Systems Technology Architecture (CHISTAR) to achieve interoperability by using a generic design approach for data which allows a wide variety of data to be stored in generic data structures. CHISTAR adopted HDFS distribute storage to store to store flat files. These files result from converting EHR data which is in different sources. It then used MapReduce framework to input flat files into HBase for querying. EHR data in HBase is retrieved by Cloud-based applications either Java APIs or SQL-like language. Figure 2.7 depicts the CHISTAR data integration approach for Cloud-based EHR storage.

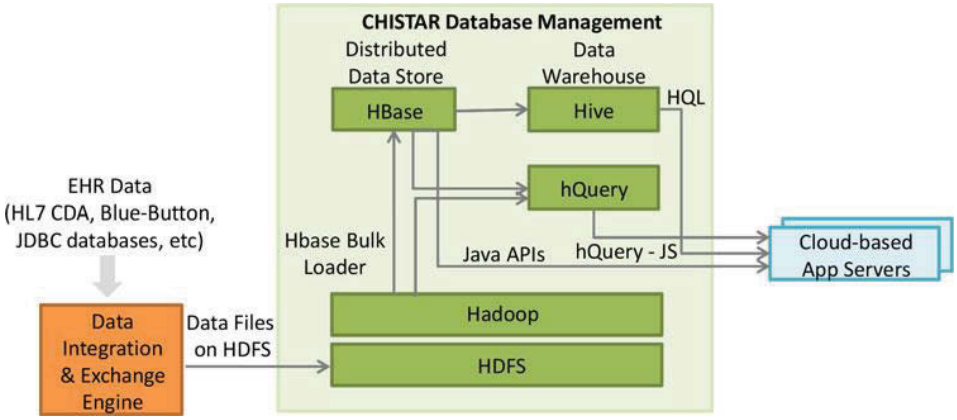


Figure 2.7 Cloud-based EHR data storage architecture of CHISTAR (Bahga and Madisetti, 2013)

(Liu and Park, 2014) presented the issues of applying current big data technologies to e-health in cloud environment. The work proposed BDeHS (Big Data e-Health Service) to support Big Data applications in the e-health area and address issues related to big data management, security, interoperability. The work also reviewed benefits and drawbacks of current big data technologies such as STORM, Hadoop, MapReduce. In addition, issues associated with data federation, data management and security need to be addressed were also outlined in the paper. Finally, BDeHS architecture was proposed as a solution which addresses big data flows, security, and data operational management. Figure 2.8 presents a cloud environment supporting e-health application. Healthcare related services are connected through the national health information networks whereas EHRs are allocated to cloud storages. Thus, entities such as doctors, hospitals, insurance providers and government agencies can access EHRs from these networks.

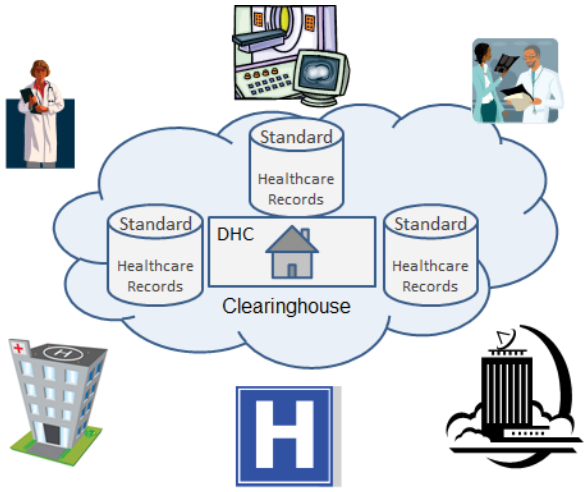


Figure 2.8 e-Health Big Data Service Environments

2.2.3 Issues related to the deployment of Cloud computing

Cloud computing dramatically reduces the expense and complexity of managing IT systems (Schubert and Jeffery, 2012). An increasing number of business customers are shifting their services and applications to Cloud computing since they do not need to invest in their own costly IT infrastructure, but can delegate and deploy their services effectively to Cloud vendors and service providers. Within cloud storage services, users can share their data with counterparts, and access data faster at anywhere and anytime with low-cost due to high availability of virtual servers rather than physical servers. It is inevitable that applications and services such as healthcare, banking will be deployed on cloud platforms to reduce the cost and

complexity of processing data while improving efficiency and accuracy. However, this deployment naturally raises a number of issues related to data security and processing as follows.

- Users may lose the controllability of the data as they may not know who, where, what, when and why there are accesses on their data.
- Cloud service providers have provided security mechanisms to deal with data protection issues by employing various policy and encryption approaches. However, data is still exposed to potential disclosures and attacks if it is moved and located at another cloud where there is no equivalent security measure at visited sites.
- A large amount of data is generated in the cloud storages, and each trunk of data is unstructured and small, but current big data models are unable to retrieve critical information quickly as data is spread into many clusters.
- Sensitive data needs to be processed at reliable or trusted resources as these data may be exposed to the public by criminal attacks or accidental information breaches.
- The rapid growth of applications with sensitive-delay requirements leads to local performance issues as requests need to be processed using local computing resources to provide real-time responses.

As a consequence, the widespread adopting of cloud computing presents a real challenge for both cloud service providers and users perspectives. In the next section, we will present challenges in deploying health data in the Cloud.

2.3 Challenges to be addressed in future research on Health Clouds

Users and health service providers are reluctant to adopt Cloud computing because of two main concerns: protection of sensitive data and efficient access of health data. Latency in retrieving relevant information from hundreds of millions of health records over a large number of geographically distributed data servers could very well be the difference between life and death for a patient (Tancer and Varde, 2012). These present serious barriers to wide adoption of cloud computing in the E-Health regardless of the reduction in costs.

2.3.1 Protection health data in Health Cloud Environments

Health records contain sensitive information that must be protected. This means that the data should only be accessed by authorized handlers AND that every action on the data has to be accounted for. The users are concerned about their EHRs not being fully protected as they lose the control of the data once it is transferred to the Cloud.

In the Cloud Service Providers' environments, the main concern is that patients might lose control of their own EHRs. They might not know who can gain access to their sensitive data, how the EHRs are processed, what details are disclosed to others, and whether the security procedure and privacy protection conform to defined SLAs. Existing work lacks appropriate schemes to protect users' sensitive EHRs from illegal disclosure or malicious violation by employees within 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 (Virvilis et al., 2011a).

Traditional security and protection mechanisms are not sufficient to protect data: when user data is passed onto a Cloud, the user no longer has control over the data and relies on the Cloud provider to assure them that their data is in safe hands. Traditional security and protection mechanisms used by cloud providers such as encryption, authentication protocols, and digital signature mechanisms are not sufficient to address the above challenges and concerns, especially in big data context with increasingly complex health data formats. For example, Virvilis et al. (2011a) introduced a cloud provider agnostic protocol combining hybrid encryption and message authentication code technologies to preserve data confidentiality and integrity for outsourcing both static and dynamic data to third parties. Shucheng et al. (2010) proposed a scheme that achieves fine-grained access control, data confidentiality, and scalability by key policy attribute-based encryption and proxy re-encryption and lazy re-encryption techniques. These traditional concepts and techniques do not really protect data as they cannot answer when the data is accessed, what has been altered, where it is moved to, by whom. Furthermore, they inevitably bring in complexity and performance bottleneck due to key distribution, sophisticated encryption, and decryption.

Existing work lacks the scheme to prevent intrusion, data leakage: existing work lacks appropriate schemes to efficiently prevent intrusion attack, data leakage, and deliver controllability and transparency to data owners. The related works focusing on policy-driven frameworks (Dan Lin and Squicciarini, 2010, Pankaj Goyal and Mikkilineni, 2009, Takabi and Joshi, 2012) play a role of adjudication in deciding the legality of access to the external layer

of data storage; however, cloud data is still in danger if adversaries have sufficient hacking skill or can leverage elevation of privilege to bypass access control service. Trust computing related technologies (Firdhous et al., 2011, Abawajy, 2011, Cong et al., 2010) administrate cloud data security through trusted third-party management systems. These schemes can efficiently increase the transparency of data usage and ensure that data is not being compromised or leaked by CSPs. However, these frameworks still disclose vulnerability once the hosted third party security services are compromised. Often, 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. These CSPs do not always employ the same protection schemes and standards (Foster et al., 2008) and data may lose its protection on the new cloud hosts. Therefore, ensuring healthcare cloud data security, mitigating users' concerns, and encouraging broader adoption of cloud computing in the healthcare sector requires alternative methodologies and technologies.

2.3.2 Efficient management of big data in Health Clouds

Currently, EHRs are stored in various proprietary formats through a multitude of medical information systems available on the market. There is an increase in the requirements of accessing information anywhere and anytime and exchange of health information and this drives the need to address the efficient management of health data. Efficient health data management must not only reduce the costs but also address protection, access speed, availability, and mobility of data.

Along with science data, financial data, social network data, health data has also entered the big data era. Ultra-high "volumes" of health data collected from patients, hospitals, and research organizations not only improve healthcare quality and aid clinicians and researchers but also help transform the way the hospital is managed with cloud storages (Lewis, 2013), resulting in a reduction of complicated procedures, efficient use of resources, and reduced operational costs. Big data analytics have huge potential to better match health care provision with need in e-Health (Tancer and Varde, 2012). When health data reaches big data level its management presents a higher level of complexity and difficulty. Various types of EHRs may require different security mechanisms that will inevitably cause additional overhead. Improvements and reductions in costs would not be very useful if these made healthcare professionals worse at their jobs by not being able to access relevant data timely. Latency in retrieving relevant

information from hundreds of millions of health records over a large number of geographically distributed data servers could very well be the difference between life and death for a patient (Tancer and Varde, 2012). Managing data usage in a big data context becomes a huge challenge and presents another barrier to wide adoption of cloud computing in the healthcare industry regardless of the reduction in costs.

Though much work has been invested into distributed databases, management of big data has not been addressed. Particularly, the challenges for health data management and related performance issues, new means of distribution and processing of data are needed. New data distribution models and programming models must address aspects of distribution, parallelism, and replication. MapReduce (Dean and Ghemawat, 2008) is an approach developed by Google for performing large-scale data analysis over its Google File System. It has received much attention because it provides a simple model through which users can express relatively sophisticated distributed programs. MapReduce programs consist only of two functions, called Map and Reduce, to process key/value data pairs. It permits data to be in any arbitrary format and supports flexible data distribution. Dynamo (Giuseppe DeCandia et al., 2007) is another highly available key/value storage system that has been used by Amazon's core e-commerce platforms. Data is partitioned and replicated using consistent hashing. Dynamo can be characterized as a special form of distributed hash table (DHT) peer-to-peer system, where each node maintains enough routing information locally to route a request to the appropriate node that holds the requested data directly. Addressing the above issues and challenges is essential to realizing the potential of healthcare cloud.

2.4 Data security and privacy mechanisms in cloud

Various approaches have been proposed to protect users' sensitive data in the cloud environment, which can be categorized as cryptography-based approaches, trust-based approaches.

2.4.1 Cryptography-based approaches

The Cryptographic approaches employ encryption schemes such as Public Key Encryption, Symmetric Key Encryption, and Attribute-Based Encryption (ABE) and its variants to encrypt data. The original data is encrypted to be unreadable by unauthorized users. This mechanism

relies on the high complexity of encryption algorithms that requires attackers more efforts and employ large computation resources to decrypt the data.

Attributed-Based Encryption (ABE) has been widely used for data access control in a cloud environment (Wang et al., 2015b, Yang et al., 2015, Sun et al., 2016, Liu et al., 2016, Pei et al., 2016, Li and Du, 2013). The concept of Attributed-Based Encryption (ABE) was proposed by Sahai and Waters (2005). In this scheme, user keys and ciphertexts are labeled by using the sets of descriptive attributes. Goyal (Goyal et al., 2006) categorized ABE into types namely Ciphertext-policy Attribute-Based Encryption (CP-ABE) and Key-Policy Attribute-Based Encryption (KP-ABE). Lewko (Lewko and Waters, 2011) proposed a decentralizing MA-ABE scheme that allows any party to be an authority. In the scheme, users' private keys associated with their attributes and a public key can be generated by a party. There is, thus, no needs for a central authority.

Kamara and Lauter (Kamara and Lauter, 2010a) introduced a secure cloud storage service which relies on different cryptographic techniques such as symmetric encryption scheme (ASE), searchable encryption, ABE, and proof of storage. However, it failed to address the scalability and auditability as cryptographic schemes rely on complex algorithms that result in performance overheads and bottlenecks.

Nesrine (Kaaniche et al., 2014) proposed CloudSec framework that uses public key cryptography to protect data in public clouds. The framework provides several public cryptographic tools to encrypt the data. Fully Homomorphic Encryption (FHE) (Beunardeau et al., 2016, AlBelooshi et al., 2016) is a strong encryption approach for data in the cloud as it provides computation over encrypted data. The computation is executed by the combination of additions and multiplications. Although cloud can achieve strong security guarantees, FHE has not been widely adopted due to its costly computation overheads.

2.4.2 Policy-based approaches

The policy-based approaches authorize access requests based on users' permissions. These approaches often employ access control models such as role-based access control, time-bound access, task-based access control or even a combination of these policies to define and assign roles for users. Role-based access control is a standard and well-known model in performing large-scale authorization management (Ferraiolo et al., 2001). However, in a cloud context, access models such as mandatory access control, discretionary access control, and role-based

access control (RBAC) lack context information to meet with privacy protection requirements. Ni(Ni et al., 2009) proposed a privacy-aware role-based access control model (P-RBAC) to protect customer privacy by enforcing policies on data accessing. The model extends the classical RBAC model by adding additional information relevant for controlled access such as Purpose, Conditions, and Obligations. Similarly, Chen (Chen and Hoang, 2011a) proposed a novel framework with Cloud-based Privacy-aware Role Based Access Control model for controllability, traceability of data and authorized access to system resources in a healthcare environment. The author enriched policies to meet with complicated cloud healthcare contexts for protecting data privacy. Beside basic elements such as Subject, Object, and Role, it introduced four new components including Organizations Purposes, Conditions, and Obligations. The proposed scheme demonstrated that it can achieve fine-grained data protection in authorizing access to cloud resources.

2.4.3 Protecting cloud data using trusted third-party technologies

Trusted third-party technology enables establishing trustworthy service relationships. (Cong et al., 2010, Abawajy, 2011, Firdhous et al., 2011) introduced trust computing concepts for cloud data security through trusted third- party management systems. These schemes can efficiently increase the transparency of data usage and ensure their data is not being compromised or leaked by CSPs. However, merely relying on Third Party Auditors (TPAs) may still disclose vulnerability once the adversary obtains the privilege to access data directly in the cloud storage. These proposed solutions lack certain protective ability on data per se. To cast off the constraints, data has to be smart to self-defend without the assistance of TPAs.

2.5 Data mobility in cloud computing

Movement of user's data is an important issue in Cloud, and it has to be addressed to ensure the data is protected in an integrated manner regardless of its location in the environment. Researchers have proposed various mechanisms to enhance data mobility and protect sensitive data in distributed cloud environments, which include geographic location-based mechanisms, data mobility based policy and encryption mechanisms, binding user and data location, protecting cloud data using trusted third-party technologies, and data mobility based on the LRD. In this section, we will present a review of these solutions for data mobility and data protection.

2.5.1 Geographic location-based mechanisms

The data's geographic location in the cloud has raised user's concern about the confidentiality and privacy as the host cloud often stores user's data at some cloud storage which is owned by another cloud. Albeshri (Albeshri et al., 2012) proposed a new approach, GeoProof as a protocol, for geographic location assurance via a proof of storage protocol and a distance-bounding protocol. It allows data owners to verify cloud locations where data is stored without relying on the word of the cloud provider. Benson (Benson et al., 2011) introduced a theoretical framework for verifying the 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. (Peterson et al., 2011) introduced a new technique for sovereignty with a data possession protocol that binds network geolocation query responses with proof of data possession using a MAC-based PDP (MAC-PDP) as the interrogated server. Ries et al. (Ries et al., 2011) focused on verifying geographic data location in a cloud environment, detecting data movements and proving data possession for data moving among CSPs. These efforts mainly relied on the linear relationship assumption network latency and distance among clouds. However, in realistic network environments this relationship is not valid. Furthermore, verifying the integrity and confidentiality of the data relies on third parties to retrieve the information from the origin of the data and this raises an additional source of concern.

2.5.2 Data mobility based policy and encryption mechanisms

In policy-based approaches, cloud policies are generally translated from the natural language policies which define the usage constraints of the corresponding data. For the cryptographic-based scheme, data is encrypted to ensure the confidentiality and integrity of data in the outsourced environment. Betge-Brezetz (Betge-Brezetz et al., 2013) proposed a privacy control model at the Virtual Machines (VM) level to protect sensitive files stored, processed, and moved in an Infrastructure as a Service (IaaS) cloud. Noman (Noman and Adams, 2012) introduced the Data Location Assurance (DLAS) solution for cloud computing environments by applying two cryptographic primitives: zero-knowledge sets protocol and ciphertext-policy attribute based encryption scheme. (Androulaki et al., 2014) proposed a new framework focusing on location- and time-based access control to cloud-stored data. In this framework, cloud provider stores the data reliably while localization infrastructure is deployed for accessing user location. More recently, (Squicciarini et al., 2010, Sundareswaran et al., 2011,

Sundareswaran et al., 2012) focused on designing a cloud information accountability framework to keep the data usage transparent and traceable through embedding the enforceable access control policies and logging policies. One of the main features of their work was that the data owner could audit the copies of its data.

The cryptographic-based solutions may include encryption (Kamara and Lauter, 2010b, Virvilis et al., 2011b, Shucheng et al., 2010), verification of integrity (Juels and Burton S. Kaliski, 2007, Bowers et al., 2009, Shacham and Waters, 2008), secure file or storage system (Kallahalla et al., 2003, Goh et al., 2003) and dispersal storage schemes (Wang et al., 2011b, Storer et al., 2007). Policy-based mechanisms mainly focus on the provisioning of access privilege for the requests. If the adversary has sufficient hacking skill or leverages elevation of privilege to bypass the access control service, the data is still in danger. Encryption-based mechanisms impose some constraints. For example, the complexity of key distribution and data encryption and decryption may be introduced, especially while protecting the highly dynamic data. These approaches have mainly focused on protecting data by using encryption schemes but there have been few efforts on data mobility.

2.5.3 Binding user and data location

Data-policy binding mechanism is a novel data protection scheme which encapsulates data, policy, and functional entities within its encapsulating object. The integrity of the data structure is enhanced. Any accesses to the data will enforceably activate the bundled security procedure. Chen (Chen and Hoang, 2013b) proposed an active data binding framework in addressing data mobility and user mobility. This research concentrated on notifying data owners when their data is moved from one location to another, tracking data geographic location using Google services and integrating an active data-centric framework to collect evidence of data movement. However, the lack of secured components or equivalence protection schemes at the new location result in data violation and illegal disclosure as there are no monitoring services to track the data and prevent illegal requests. Consequently, data cannot survive independently in the heterogeneous cloud environment. The works (Othmane et al., Ranchal et al., 2010, Angin et al., 2010) proposed the active bundle scheme to protect disseminated sensitive data. Each bundle packages data and metadata involved in data access and dissemination policies. Through a relevance VM with each bundle, the data can be protected by enforcement of the privacy policies in the VM. Popa (Popa et al., 2013) proposed a data traceability service for mobile

computing application. This approach only considers the case that the data is accessed by the data owner and there is no data movement among clouds. In particular, a request to send data is simply performed by creating a copy of the data. In other words, the data sent by the user will not be traced by the service.

2.5.4 Data mobility based on LRD

LRD enables subscribing and storing data locations when data is allocated at cloud storage. (Noman and Adams, 2012) used database for data registration and user' verifications to provide DLAS at the cloud. Nevertheless, data is held in the cloud to inform users that it is protected in a secure environment and there is no data moving operation among clouds. In fact, research on data mobility and LRD in the cloud has not been well-established to support tracking and tracing data location records.

2.5.5 EHR Security and privacy mechanism in cloud

As EHR contains sensitive health data of a person, it must be protected from unauthorized accesses. Security and privacy are thus crucial in any HIS. Numerous approaches have been proposed and used for preserving the privacy and security of EHR. However, there is no clear classification of the security approaches. For simplicity, we classify these approaches into three categories: policy-based authorization, cryptographic, and policy-based authorization and cryptographic. The policy-based authorization allows data to be accessed with the role-based access control. The cryptographic approaches employ encryption schemes and cryptographic primitives. Policy-based authorization and cryptographic approaches combine the policy-based and cryptographic approaches.

2.5.5.1 Policy-based authorization approaches

Policy-based authorization approaches ensure that EHR can only be accessed by authorized users who received permission rights to access patient data in HIS. These approaches employ different policy mechanisms such as role-based access control, time-bound access, task-based access control or even a combination of these policies to define and assign roles for users. As a result, with both valid system identity credential and access credential, users can legally obtain corresponding patient data from EHR databases without disclosure-sensitive information. Narayanan and Giine (2011) introduced Task Role-Based Access Control in which roles are

used to support passive access control and tasks are used to support active access control. Policy-based authorization approaches allow users to access their EHR if they and their roles meet the systems' policies. However, health data are not encrypted and hence these approaches cannot protect sensitive data against malicious modification or intrusive attacks.

Policy-based authorization approaches ensure that EHR can only be accessed by authorized users who hold permission rights to access patient data in HIS. These approaches employ different policy mechanisms such as role-based access control, time-bound access, task-based access control or even a combination of these policies to define and assign roles for users. As a result, with both valid system identity credential and access credential, users can legally obtain corresponding patient data from EHR databases without disclosure-sensitive information. For example, physicians who are responsible for patient diagnosis and care can obtain the access rights such as reading patient's EHR contents for treatment purpose only if the current time is during 9 AM to 6 PM.

A novel framework with Cloud-based Privacy-aware Role Based Access Control (CPRBAC) model for controllability, traceability of data and authorized access to system resources in a healthcare environment is proposed in (Chen and Hoang, 2011a). Task Role-Based Access Control is presented in (Narayanan and Giine, 2011) where roles are used to support passive access control and tasks are used to support active access control. Both workflow based and non-workflow based tasks are supported by this model. These approaches focus on identifying users with their own roles in accessing EHR but data are not protected in encrypted forms.

Policy-based authorization approaches can solve the issues for users accessing EHR with the systems' policies to receive health records corresponding with their roles. However, one of the critical issues is that health data are not encrypted. Consequently, these approaches lack an efficient surveillance mechanism to protect sensitive data against malicious modification or intrusive attacks due to its centralized nature.

2.5.5.2 Cryptographic approaches

The Cryptographic approaches used to secure EHR are based on encryption schemes such as Public Key Encryption, Symmetric Key Encryption, and Attribute-Based Encryption (ABE) and its variants such as Ciphertext-policy Attribute-Based Encryption (CP-ABE), Key-Policy Attribute-Based Encryption (KP-ABE), Multi-Attribute-Based Encryption (MA-ABE). In these approaches, EHR contents which are mainly objects for encryption schemes are stored in

encrypted forms to provide data confidentiality in HIS. The decryption process is the reverse process at the receiver end when users retrieve the data.

Jie (Jie et al., 2012) employed KP-ABE and CP-ABE to share EHR stored in the cloud. These schemes enable owners to achieve full control over their EHR files and to select users who would like to share their EHR and which parts of EHR data a user can read.

Broadcast Ciphertext-policy Attributed-Based Encryption Attributed-Based Encryption (bABE) was used to create a secure and privacy-preserving EHR system by Nararyan (Narayan et al., 2010). This approach addresses the key management issues since it employs users' attributes for data encryption. Every user is then provided with one private key base on their attribute set for decryption. In addition, a Secure Channel Free Public-Key Encryption with Keyword Search (PEKS) was combined with bABE to provide keyword search within a health record.

Li (Li et al., 2010) also proposed another cryptographic framework by using an extension of ABE. The framework employs ABE as the encryption primitive by classifying Users/data according to their attributes, such as professional roles/data types. Moreover, users are divided into the whole Personal Health Record system into multiple Security Domains (SDs) namely Public Domains (PUDs) and Personal Domains (PSDs). A large number of professional users and Multiple Public Attribute Authorities (PAA) can be included in a PUD which can be the health care, education, government or insurance sector. An encrypted EHR can be accessed by authorized users from both PSD and PUDs. These domains retrieve credentials from the corresponding public authorities, without the need to interact with any PHR owner. This can avoid the key management overhead of owners and users.

Public Key Encryption (PKE) (Wohlmacher and Pharow, 2000) introduced the use of public-key certificates in the health sector and the private keys corresponding to the certified public keys are bound to a carrier - a secure token as a smart card namely health professional card (HPC). (Doukas et al., 2012) introduced a system based on Gateways namely Internet of Things (IoT) Gateways which address health sensor data and resolve security issues through digital certificates and PKE data encryption.

Symmetric Key Encryption (SKE) (Chen and Hoang, 2011b) used symmetric data encryption to encrypt data blocks in the data protection framework. Only the leaf nodes in the tree structure of EHR data blocks are encrypted to avoid computation overhead. (Yu and Chekhanovskiy, 2007) introduced a patent-centric protection system by using a smart card and SKE based on

Advanced Encryption Standard (AES) to protect EHR. Patient, healthcare providers, and support personnel will be provided with a smart card. Patient's smart card is used to encrypt his or her EHR and to identify role-based access control while doctor's smart card allows him or her to sign the transaction in accessing the EHR.

Cryptographic approaches prove that EHR can be secured at a high level by using encryption schemes. Firstly, EHR will be stored in encrypted forms which avoid the disclosure of sensitive health information. In addition, schemes such as RSA and AES proved that it is very hard to break these cryptographic systems while empowering HIS to share patient information efficiently. Nevertheless, computing overhead and key management are some of the challenges which need to be addressed when applying these solutions. In fact, the lack of defined roles to access data in several schemes makes it impossible to deploy the schemes in a multi-users environment where users can own one or more roles in the system.

2.5.5.3 Policy-based authorization and cryptographic approaches

The combinations of policy-based authorization and cryptographic approaches ensure data confidentiality by defining roles for clients to access EHR and by storing EHR in encrypted forms. These result in appropriate security frameworks as approaches embedded in HIS in order to serve for security requirements. In order to access EHR, users need to obtain corresponding roles which describe permissions such as read, write to EHR. These roles have to be authorized by HIS or third parties before users receive keys for decryption processes. (Kathryn Garson and Adams, 2008) proposed Policy Based Encryption which builds on the idea of encrypting under an arbitrary string. In order to access the document, users need to be authenticated by the system and receive a decryption key associated with their role.

These approaches provide high-security level and meet with requirements when healthcare moves to new infrastructures such as cloud computing. However, it needs to be considered that the approaches are complex and require systems implementing on the same framework or adaptable framework while the majority of current HIS are developed based on different sources and providers.

The combinations of policy-based authorization and cryptographic approaches ensure data confidentiality by defining roles for clients to access EHR and storing EHR in encrypted forms. These result in appropriate security frameworks which are approaches embedded in HIS in order to serve security requirements. In order to access EHR, users need to obtain corresponding

roles which describe permissions such as read, write to EHR. These roles have to be authorized by HIS or third parties before users receive keys for decryption processes.

As presented in (Chen and Hoang, 2011b), the data protection framework employs both CPRBAC and Symmetric Data Encryption. (Garson and Adams, 2008) proposed Policy Based Encryption which builds on the idea of encrypting under an arbitrary string. Besides these approaches above, there are some effective frameworks which assist in protecting health data. The National EHealth Transition Authority (NEHTA) proposed a framework namely National EHealth Security and Access Framework (NESAF). NESAF supports organisations engaged in national EHealth to adopt a consistent approach and application of health information security standards and provides better practice guidance for the Australian health sector. The NESAF has been developed in collaboration with almost 200 unique stakeholders, including health practitioners, health business owners, health information managers, consumers, privacy advocates, information security experts and health technology specialists. The NESAF starts by focusing and understanding the risk profile of a given organisation, and where it will be located in the overall e-Health environment.

These approaches provide a high-security level and meet with requirements when healthcare moves to new infrastructures such as cloud computing. However, it needs to be considered that the approaches are complex and require systems implementing on the same framework or adaptable framework while the majority of current HIS are developed based on different sources and providers.

2.6 Big data management models

The growing popularity of “cloud computing” results in a large number of internet applications together with a very large anount of data to cloud. Consequently, providing data management services in the cloud enables both users and CSPs to achieve higher levels of efficiency in terms of scalability, reliability, and security. This is because the traditional databases cannot scale to the thousands of nodes when they are deployed in the cloud context. In addition, it is difficult to support multiple, distributed updaters to the same data set and replicate huge data sets for availability, due to capacity (storage, network bandwidth, …). To deal with exponentially growing data volumes, Amazon and Google introduced a new storage model known as NoSQL databases such as Amazon's Dynamo (DeCandia et al., 2007) and Google's BigTable (Ghemawat et al., 2003). No-SQL systems adopt schemes such as a key-value pair or document

collections to store data in distributed systems rather than structured tables in relational databases. P2P is another data management mechanism to deal with the growing data in information systems. P2P can be incorporated with traditional databases or by itself to be an efficient model for big data management. Figure 2.9 presents current big data models in the cloud environments. The next sections introduce data models that address challenges relating to high availability and scalability in a cloud environment.

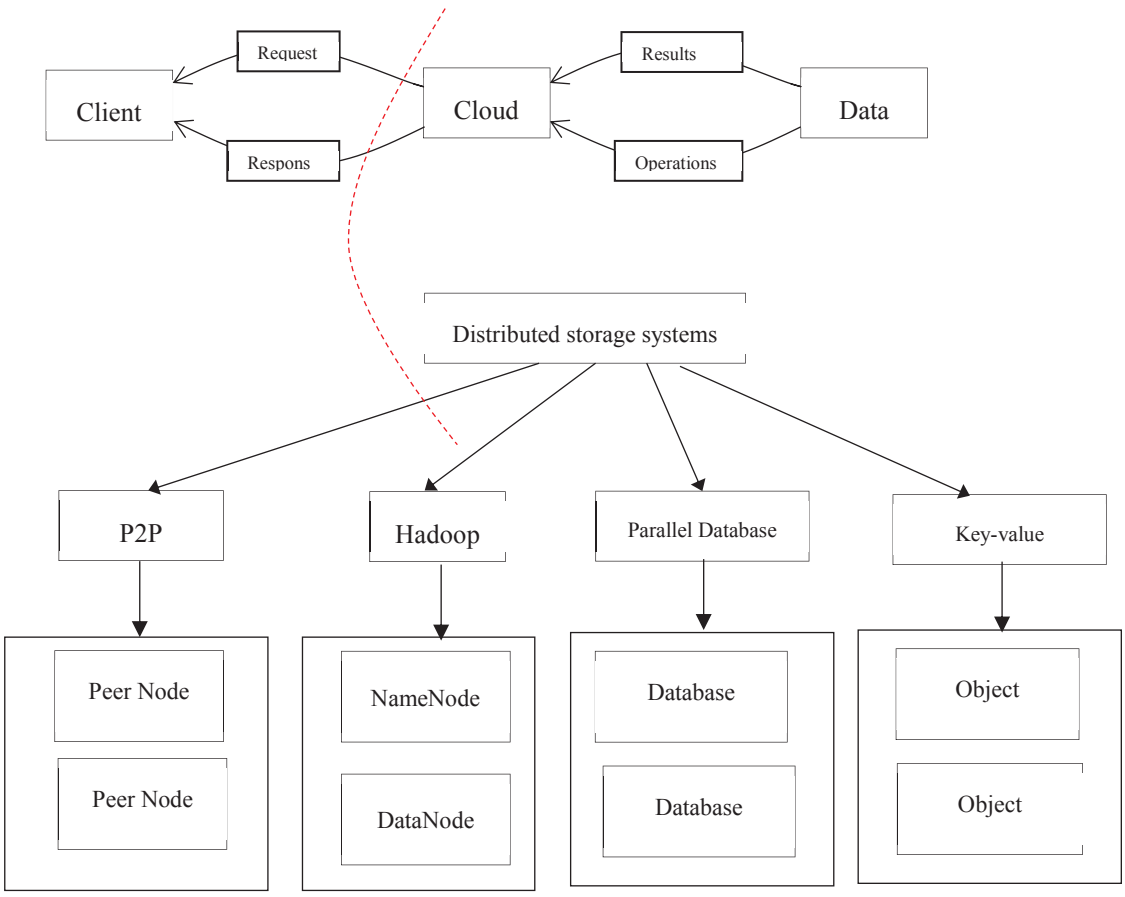


Figure 2.9 Current big data models

2.6.1 Peer-to-peer network

P2P network has been adopted widely in distributed systems and in many successful P2P applications such as distributed data storage, application-layer multicast, and event notification services (Risson and Moors, 2006, Le et al., 2006, Min and Yuanyuan, 2010). It supports large-scale resources discovery and sharing file systems by relying on scalable, dynamic and self-organized infrastructure. In addition, it is also a decentralized infrastructure, enabling access to provided services at each node in the whole network. P2P is a logical overlay network laying

on top of a physical network. In P2P network, each peer is associated with a node and plays the same role in the system. Peers are connected via logical links and accessed peers' address. One link is assigned to a physical path in the physical network to route paths with appropriated requests.

In a normal structured P2P system, the Distributed Hash Table (DHT) is built on top of an overlay network. Data lookup and insertion process are efficiently supported by the hash table. Given the key of the file, the corresponding value of the file can be inserted and searched by hashing the key to a value with an appropriate hash function. The hash value is the index of the file and all the hash values are from the ID space. In DHT, peers are delivered in the ID space. Each peer is responsible for one partition of the ID space. Peers are connected by an overlay network through which the requests for data insertion and lookup are delivered. Basically, there is only one hashing round using the data key such as filenames or file content to obtain the value corresponding to the file. Files might be distributed relatively evenly based on hash functions in the original P2P systems. However, for a system with a very large number of files relative to the number of storage nodes and if the access pattern is skewed, the load files will be distributed to their nearest hashed ID nodes which may not be evenly with some nodes heavily loaded and others extremely lightly loaded. In other words, there may be some nodes storing large amount of files in the system while others have few files or no files. Consequently, load balancing is not achieved and data retrieval is not efficient. Firstly, peers still perform the lookup process in the same ID space but the destinations nodes tend to access several nodes in the systems. In addition, storage space of peers is not used efficiently causing not only unbalanced distribution but also lots of unused storage in peers. Figure 2.10 presents the lookup process in original Chord in a P2P system.

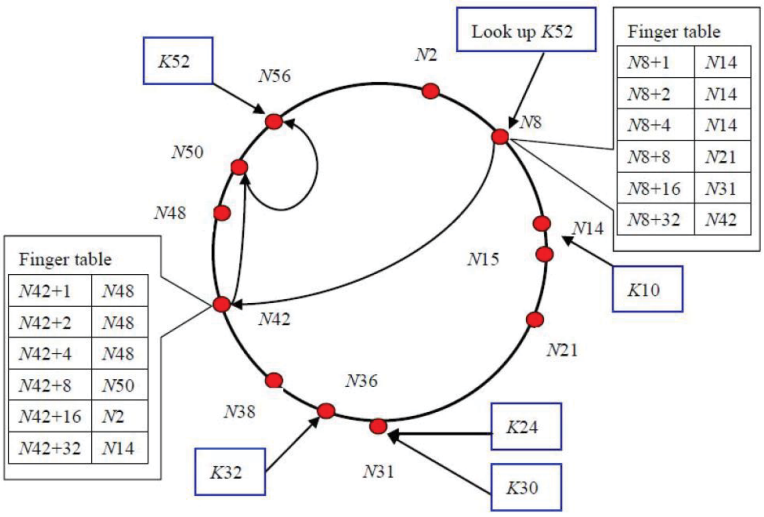


Figure 2.10 Illustration of lookup process in original Chord (Wang et al., 2012)

2.6.2 Hadoop and MapReduce

Hadoop (Apache, 2017) is an open-source implementation of MapReduce which has been widely adopted for handling large scale data in distributed systems. It has emerged as the leading computing platform for both academia and industry used by many companies such as Google, Facebook, New York Times. The Hadoop distributed file system (HDFS) is a distributed file system that stores data commodity nodes, providing a high degree of fault-tolerance and high aggregate bandwidth by using parallel processing in clusters (Wiki, 2017). A HDFS cluster consists of a single NameNode as the master node and number of DataNodes as the slaves. HDFS implements Google File System (GFS) (Ghemawat et al., 2003) by providing file system name space and storing users' data in files. It partitions each file into one or more blocks and stores these blocks at Datanodes. The Namenode manages the file system namespace and client accesses while DataNodes provide block storage and serve I/O requested from clients such as creating, replicating, and deleting blocks.

MapReduce (Dean and Ghemawat, 2008) is introduced by Google as a framework for processing large-scale data on clusters of computers in a massively parallel manner. MapReduce has been adopted broadly in various applications such as business data analysis, machine learning, scientific research, multimedia processing, and healthcare (Wiki, 2017, Neshatpour et al., 2015, Bahga and Madisetti, 2013). The MapReduce programming model consists of two phases: the map phase and the reduce phase. In the map phase, each map task is allocated to a map slot on a machine and processes a portion of the input data. In the reduce phase, the outputs of the map phase with the same key are processed by a reduce task allocated to a reduce slot. The Map function processes key/value records from an input file and then outputs a set of intermediate new key/value records. These output records are partitioned into disjoint buckets by applying a (hash) function to the key of each output record and are written to the server's local disk. In fact, multiple instances of the Map function run on different servers of the cluster and each map instance is assigned a distinct portion of the input file by the scheduler to process. The second phase of a MapReduce program executes instances of the Reduce program. The input for each reduce instance is output files from the map server local disks. Each reduce processes the records assigned to it and then writes records to an output file in the distributed file system as part of the computation's final output.

Figure 2.11 shows the execution of the MapReduce program. MapReduce adopts a master node which manages workers as slave nodes. Heartbeat messages are used to provide communications between the master node and workers, including job status and workers' status. Jobs are scheduled by a scheduler which follows defined policies to assign tasks to slots of workers.

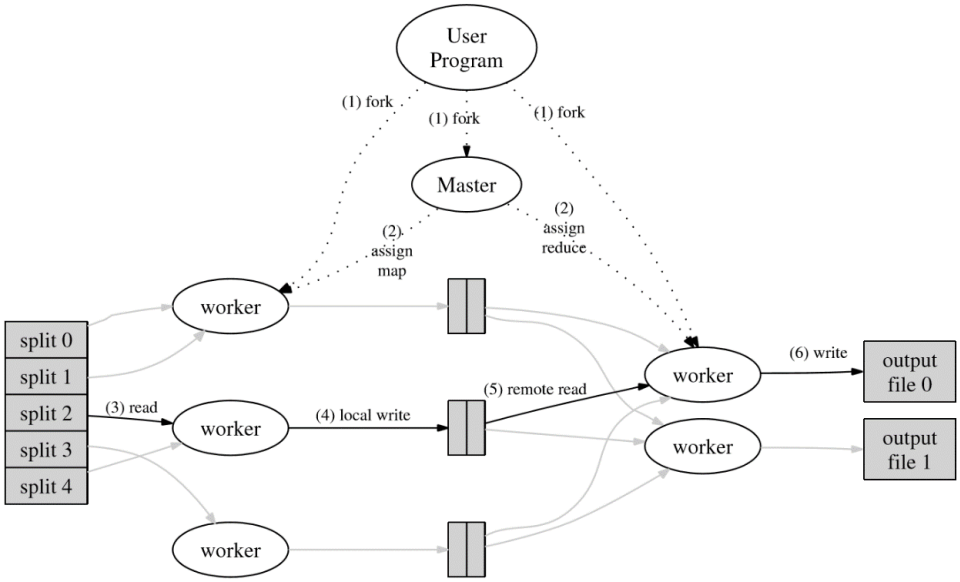


Figure 2.11 Overall flow of a Map-Reduce operation (Song et al., 2011)

"Map" step: The master node takes the input, chops it up into smaller sub-problems, and distributes those to worker nodes. A worker node may do this again in turn, leading to a

multi-level tree structure. The worker node processes that smaller problem, and passes the answer back to its master node.

"Reduce" step: The master node then takes the answers to all the sub-problems and combines them in a way to get the output - the answer to the problem it was originally trying to solve.

In the MapReduce program implemented on Hadoop, the master node consists of two processes: NameNode and JobTracker while workers comprise two processes: DataNode and TaskTracker. A file is partitioned into blocks of 64 MB and stored at the worker. Each block is then replicated three times. JobTracker is responsible for initiating job and assigning map or reduce tasks of jobs to TaskTracker. TaskTracker executes map or reduce tasks that are assigned by JobTracker.

Hadoop employs different schedulers for task scheduling namely FIFO, capacity and fair

schedulers (Apache, 2017).

- FIFO simply schedules jobs in a first-in first-out manner.
- The Capacity scheduler enables sharing resources in the large cluster among multiple organizations based on computing needed. An organization is allowed to access any excess capacity not being used by others.
- The FAIR scheduler shares the cluster's resources equally among multiple users. Each user is provided with a separate pool which is divided into the same capacity across the cluster so each user is granted an equal share.

2.6.3 Key-value data model

Key-value data model (Cattell, 2011) is the simplest data model using a single key-value to index for all data. In this model, values (data) are stored based on programmer-defined keys which are indexed to serve for searching. Systems based on this model normally provide clients interfaces to insert, delete and lookup. This model is efficient for services that only need primary-key access to a data store. For example, Dynamo (Amazon) (DeCandia et al., 2007) provides a simple primary-key access only interface to meet with application requirements. It is the underlying storage technology for a number of the core services in Amazon's e-commerce platform and uses Dynamo instances for each service. Objects associated with a key are stored through a simple interface. Two operations namely get() and put() allocate objects' location for storing and retrieving. The get(key) operation searches for the object replicas associated with the key in the storage system and returns a single object or a list of objects with conflicting versions along with a context. The put(key, context, object) operation determines where the replicas of the object should be placed based on the associated key, and writes the replicas to disk. When there is an increase in the data set size or request rates, the scheme will be scaled out to support for service requests.

2.6.4 Current Approaches for P2P in handling data distribution

Various distributed storage systems such as Hadoop (Tantisiriroj et al., 2011), P2P systems (DeCandia et al., 2007, Lakshman and Malik, 2010) have been widely deployed in the cloud environment. However, storing and retrieving a large number of files with various data sizes and structures pose a big challenge for current big data models. The work in (Dong et al., 2012) proved that Hadoop is not efficient in managing a large number of small files. Hadoop is

designed to process large scale data in data mining and machine learning. Despite the fact that P2P systems and mechanisms are generally efficient for data lookup with low latency, they do not perform well in processing large-scale data analytic jobs compared to Hadoop (Chen et al., 2012). Alternative infrastructures and access mechanisms are needed to handle this type of data and applications. In this section, we present related work that reflect to P2P systems in handling data distribution.

Chen (Chen et al., 2012) presented BestPeer++, a system which delivers elastic data sharing services for corporate network applications in the cloud based on BestPeer – a P2P system based data management platform. This model was deployed as a service in cloud environment including bootstrap peer as monitors, normal peer as database engine and access control. In fact, an adaptive query is also proposed to switch between P2P engines for small scale data to leverage fast performance on local database engine and Map Reduce engine for large-scale data in order to exploit the benefits of the Hadoop model. The work in (Min and Yuanyuan, 2010) proposed a hybrid P2P system as a combination of structured P2P called *t-network* and unstructured P2P called *s-network*. The *t-network* plays the role of a core transit network while *s-network* stores data in the system and each *s-network* is attached to a peer in t-network. The p_s parameter is defined to adjust the number of peers in each of the two networks. Altmann (Altmann and Bedane, 2009) presented a P2P file sharing topology based on social network enabling users to share resources for their friends or family. There is a limited number of peers accessing the resources based on their established relationship. The work in (Yuqi et al., 2009) presented a data replication approach to avoid overload of some objects as hotspots by using multiple hash functions. These hotspots are replicated to different nodes dynamically resulting in the improvement in access latencies and load balancing. (Carra et al., 2008) proposed a model to distribute the content to users in the overlay network as a Constrained Stochastic Graph Process (CSGP). The CSGP is modelled by different architectures based on trees and meshes. The file distribution is only considered to deliver the content in the shortest possible time to users in the P2P systems while the searching task is not the main process. In our work, we combine either clustering or searching files for the P2P system. Files are controlled to store in defined clusters and the same mechanism for storing is applied for searching.

However, these efforts focus mainly on structuring P2P systems to deal with dynamically joining and leaving nodes; the data distribution still relies on the original P2P system. Besides that, searching requested files is performed the same way as in the original system.

Our proposed scheme focuses on distributing and retrieving data from P2P based on two hashing rounds in order to enhance the performance of P2P systems. Through the first hashing round, data is distributed to defined clusters enabling efficient retrieval of data.

2.7 Trust-based scheduling for big data processing with MapReduce

Various mechanisms to protect sensitive data for MapReduce have been proposed over the last few years. To that end, task scheduling mechanisms have been investigated in different aspects. We summarize the related work (Figure 2.12) from the following perspectives: hybrid clouds for processing sensitive data, security mechanisms for MapReduce, trust for MapReduce, and Task scheduling for MapReduce.

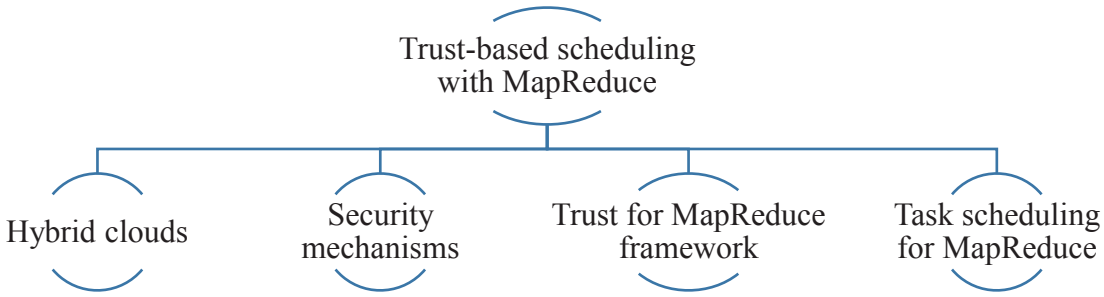


Figure 2.12 Mechanism for Trust-based scheduling with MapReduce

2.7.1 MapReduce framework based on hybrid clouds to process sensitive data

Hybrid clouds have addressed the privacy and integrity of users' sensitive data. Xiang (Xu and Zhao, 2015) proposed a framework using tagging mechanisms (Figure 2.13). The sensitive data is tagged and retained at private clouds only. The framework employs tagging mechanisms similar to tagged-MapReduce but providing different types of file tags; level sensitive tags, line level sensitive tags, temporal and spatial sensitive tags. Specifically, MapReduce was extended to protect sensitive data by tagging the sensitive data and retaining them within private clouds only. Non-sensitive data is allocated to public clouds. Before allocating MapReduce tasks, data is pre-processed into a sensitive data stack, and a non-sensitive data stack in order to apply

different sensitive tags to files and lines based on the input configuration. These MapReduce tasks are then executed on both private and public clouds. Similarly, Zhang (Zhang et al., 2014) introduced a framework to secure mix-sensitive data on hybrid cloud based on tagged MapReduce (Figure 2.13). It separates non-sensitive data, which can be outsourced to public clouds, while sensitive data is processed at private clouds. The prototype of tagged-MR on Hadoop demonstrated the feasibility of the system and the effectiveness in scheduling tasks. However, these efforts rely on the hybrid clouds which do not utilize cloud resources efficiently for big data applications. They performed a fixed scheduling strategy to process sensitive data at private clouds.

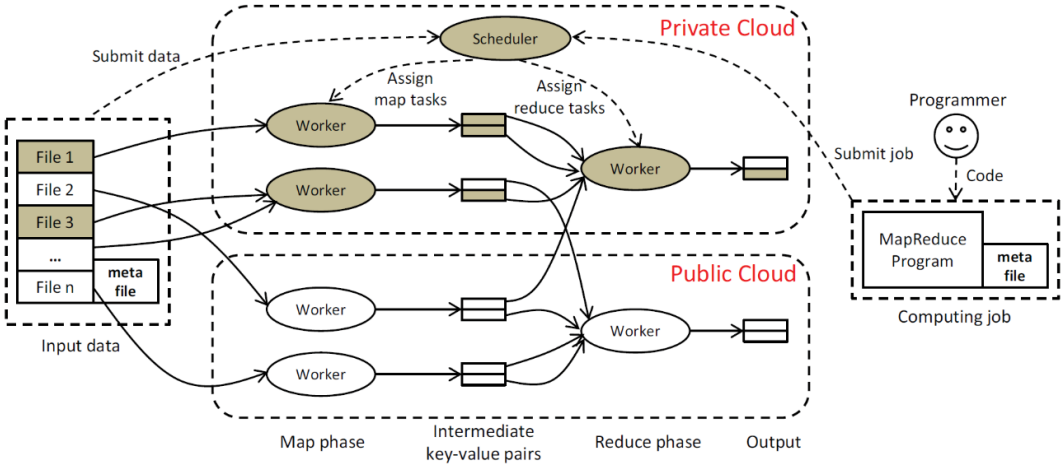


Figure 2.13 A framework using private clouds to process sensitive data with MapReduce (Zhang et al., 2014)

2.7.2 Security mechanisms for MapReduce framework

(Wang et al., 2011a, Ateniese et al., 2008, Liu et al., 2014) proposed third party auditing (TPA) schemes which carry out data verification on a regular basis for users' sensitive data. (Liu et al., 2014) proposed an authorized public auditing scheme based on Boneh-Lynn-Shacham (BLS). BLS signature and Merkle hash tree can support fine-grained update requests. However, these TPA schemes are inefficient since the more frequently data updates are performed, the more computational overheads are added.

Research works focused on trusted third-party systems were introduced in (Abawajy, 2011, Casola et al., 2015, Modic et al., 2016). They enable establishing trustworthy service relationships among CSPs. However, merely relying on the third-party system cannot address fully users and CSPs requirements as CSPs may trust one another differently and a CSP may

rank (trustwise) its own resources differently according to their types.

Encryption techniques provide high-security level for the data. Yang (Yang et al., 2015) introduced an access control scheme based on Attribute-based encryption (ABE) with dynamic policy updating for big data. It allows data owners to send policy updating requests to a cloud server to update policies of encrypted data. Lei Xu (Xu et al., 2015) proposed a model that covers both data and program security of big data processing. The model employs operation steganography and Fully Homomorphic Encryption (FHE). However, encryption mechanisms encounter more computational overheads when deploying in big data applications and heterogeneous architecture of cloud computing.

2.7.3 Trust for MapReduce framework

Many trust mechanisms have been proposed for MapReduce (Ulusoy et al., 2015, Wei et al., 2009, Huang et al., 2012, Ruan and Martin, 2012, Gao et al., 2015). They provided integrity verification and attack detections at node and task levels. (Ulusoy et al., 2015) proposed a replication-based verification scheme for detecting attacks with very high detection rate in MapReduce program. Map or reduce tasks are decomposed into small tasks which are replicated selectively and executed at other nodes in the clusters to identify compromised nodes and corrupt results. Huang (Huang et al., 2012) presented the two result verification schemes for MapReduce framework within text applications. It uses watermark injection to inject watermarks into input data and the computed results are verified the consistency and inconsistency. Ruan (Ruan and Martin, 2012) proposed a Trusted MapReduce framework by combining Trusted Computing Group (TCG) with MapReduce system. No replication is used in this framework such that the overheads are reduced significantly. Gao (Gao et al., 2015) introduced an integrity protection approach for big data processing with dynamic redundancy computation. A trust rating system was employed to monitor duplicate computations and identify suspicious nodes. Nevertheless, these schemes are not deployed adequately at CSPs, resulting in interoperability problems when employing resources from different CSPs for big data processing.

2.7.4 Task scheduling for MapReduce

Scheduling tasks for MapReduce have been investigated in different aspects (Chen et al., 2015, Wei et al., 2015, Kuo et al., 2014, Chang et al., 2011, Zhang et al., 2015, Mashayekhy et al.,

2015). The bipartite graph was used to model the scheduling problem in (Chen et al., 2015, Wei et al., 2015, Kuo et al., 2014). (Chen et al., 2015) proposed a MapReduce scheduler for a deadline-constrained problem. A minimum weighted bipartite graph was formulated to match between tasks within deadline-constrained and slots of nodes in clusters. The bipartite graph was also applied to solve Virtual Machine Placement problem in MapReduce framework (Wei et al., 2015, Kuo et al., 2014). Chang (Chang et al., 2011) formulated MapReduce scheduling problem as a linear program that minimizes the job completion times to solve the problem. Two 3-approximate algorithms were proposed for the offline case and the online version to minimize the completion time of all jobs. Zhang (Zhang et al., 2015) introduced a fine-grained resource-aware MapReduce scheduler which performs allocating resources to tasks at phase-level. Mashayekhy (Mashayekhy et al., 2015) formulated the problem of scheduling as an integer program. Two heuristics algorithms were proposed to find the assignment of tasks to slots that minimizes energy consumption of MapReduce framework in data centers.

In this thesis, we focus on defining trust metric for cloud resources and scheduling these resources to tasks, which is important because these trusted resources provide a secured execution for users' sensitive-data in different CSPs no matter in private clouds or public clouds. Tasks executing on either sensitive data or non-sensitive data are allocated naturally to appropriate trusted resources that protects the data leakage. This also, moreover, utilizes cloud resources efficiently so that there is increasing performance of big data processing. The trusted based scheduling scheme provides high level trust assignment of slots to tasks. This means selected slots which not only satisfy trust requirements but also contribute highest trust values in the total trust. Consequently, CSPs can achieve highest gains in term of revenue when performing tasks within this strategy.

2.8 Fog computing

2.8.1 Definitions and features

Fog computing enables more processing tasks to be performed at the network edge before being moved to the core network or centralized clouds. The decisions shall be made by Edge devices rather than being submitted and received from clouds. This leads to more efficient process and the ability to react more quickly to events (Mouradian et al., 2017, Stojmenovic and Wen, 2014). With the potential for billions of IoT devices creating data, data management becomes

an issue at the edge network since it fails in providing adequate bandwidths for all of the data to be transferred through the network. Architectures for fog are, however, in the early stage of being defined with open issues for the current research (Consortium et al., 2016). In fact, current definitions are also proposed in different perspectives (Yi et al., 2015a).

As defined in (Yi et al., 2015b), "*Fog computing is proposed to enable computing directly at the edge of the network, which can deliver new applications and services especially for the future of Internet*". According to (Bonomi et al., 2012, Vaquero and Rodero-Merino, 2014), Fog computing is a non-trivial extension of cloud computing and lays from the core network to edge network. It resources is structured from the large scale of from nodes and distributed geographically at edge networks. Fog devices form a resource pool at the edge network to provide computation, storage, and communication with low latency response to serve for a large scale of clients. Fog devices are heterogeneous ranging from high-end servers to end devices such as mobile devices, wearable devices.

Fog node is defined as heterogeneous semi-virtualized components deployed in a variety of environments where computing and storage are structured from several fog devices with a weak performance at edge network. It can provide computation, network resources, and storages near to applications in real-time. Fog nodes can be deployed in the core, edge, access networks, and endpoints.

Fog computing provides low latency response and location awareness services for users at edge network. Thus, it may be potentially widely adopted in practice to provide resources and services to applications that are latency sensitive. Some of the common key features can be presented as below:

- *Heterogeneity*: Fog computing is a virtual-based platform that provides computational, networking and storage services.
- Geographic distribution: resources of Fog computing are distributed geographically and deployed nearby users to deliver low latency computing services.
- *Edge location*: The concept of Fog computing is structuring and providing resources and services locally at the edge of the network.
- *Real-time response*: various applications, such as smart home, healthcare monitoring, and smart grid, rely on fog's computational resources to achieve low latency response.
- *Support for mobility*: fog devices with high mobility demand change locations frequently. Hence, mobility becomes an essential feature to provide both resources and

communications for applications.

- *Scalability*: this is applicable when monitoring demands are integrated applications such as healthcare, smart traffic, and smart grid. These applications are deployed in distributed systems that require distributed computing and storage resources.

2.8.2 An architecture of Fog computing

With the potential for billions of IoTs devices creating data, data management becomes an issue at the edge network since it fails in providing adequate bandwidths for all of the data to be transferred through the network. Architectures for fog are, however, in the early stage of being defined with open issues for the current research (Consortium et al., 2016). Thus, different Fog computing architectures have been proposed in the literature (Osanaiye et al., 2017, Yi et al., 2015b, Bonomi et al., 2014, Bonomi et al., 2012). The architecture of Fog computing is implemented in the context of Fog and cloud computing. Figure 2.14 shows the implementation of Fog computing. Fog computing is deployed in between cloud and smart devices. In this architecture, each fog node is structured from several fog devices to provide computation and storage for applications. Data generated from fog devices are collected and send to fog nodes first. These data are filtered and processed locally for real-time requests. The rest of data are sent to the cloud for analytics and reporting purposes.

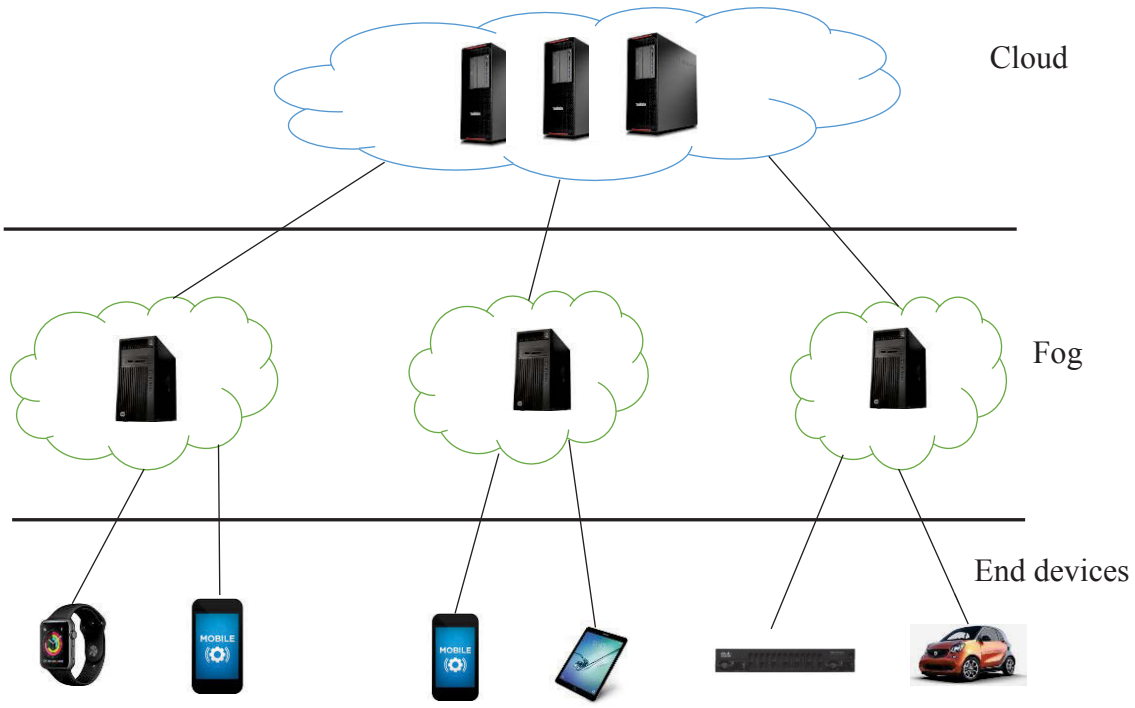


Figure 2.14 Architecture for Fog computing

2.8.3 Fog computing security and privacy challenges

Fog computing has become a computing model for providing real-time computing services and storage. However, few research efforts have focused on security issues which existed intrinsically in highly dynamic computing services. Clinton (Dsouza et al., 2014) proposed a policy-driven security management framework for Fog computing which secures collaboration and enables interoperability of user resources. Yi (Yi et al., 2015b) identified authentication, access control, intrusion attack and privacy issues that Fog computing may potentially encounter in designing and deploying. Similarly, Stojmenovic (Stojmenovic and Wen, 2014) presented authentication at different levels of gateways and smart meters as the main security issue in the fog environment. CPRBAC was presented in (Chen and Hoang, 2012). The model, however, fails to provide low sensitive-latency verification as it is at cloud-based authorizing. Studies related to potential security and privacy problems in Fog computing (Lee et al., 2015, Wang et al., 2015a) have been investigated to identify types of attack on users' data. Authors have raised some potential attacks such as man-in-the-middle attack, intrusion detection, malicious detection, and malicious Fog. Nevertheless, the lack of security approaches in Fog computing prevents users from adopting Fog computing since the users' sensitive data may be exploited and compromised leadding to the vulnerability by the adversaries.

Our work focuses on designing an important class of fog security framework and building trust among regions enabling fog's mobility. In particular, a MS with location registration is designed and implemented to store information about fog's devices registration, users, operations, location, and timestamp. In fact, FPRBAC verifies requests to allow users accessing computing resources from fog nodes based on granting permissions based on assigned roles.

2.8.4 Resource scheduling challenge in Fog computing

Research on scheduling tasks for fog and cloud resources have been mainly on costs and energy consumption.

Lingjun (Pu et al., 2016) proposed Device-to-Device fogging framework that schedules mobile tasks to available resources of mobile devices. A task scheduling approach in a cloud-fog computing system was presented in (Xuan-Qui and Eui-Nam, 2016). It uses VMs at cloud as extended resources when fog nodes do not have sufficient resource to fulfill clients' requests. The optimization approach focuses on cost for cloud services instead of latency-sensitive

responses to clients. Deze (Zeng et al., 2016) proposed a joint optimization of task scheduling and image placement in a Fog computing supported software-defined network embedded system. Computational resources are provided from two sources; embedded clients and fog nodes represented by computation servers. Storage servers can be shared by both clients and computation servers. Jessica (White, 2012) proposed a low complexity small cell clusters establishment and resources management customizable algorithm to address the load balancing in Fog computing. The proposed mechanism allows small cell (SCC) to minimize the computation resource and power consumption while still satisfying users' requests. Yuhua Lin (Wang and Shi, 2014) proposed CloudFog which is a lightweight system and allocated nearby users to provide computation resources. Fog nodes play as supernodes to render video games and stream them. Cloud servers serve for heavy computation tasks and update computation results to supernodes. Our work employs computing resources from both regions and clouds to handle requests with sensitive latency demands. In fact, these resources are scheduled optimally to allocate to each request hence providing a better trade-off compared to fog resources or cloud resources alone.

2.9 Summary

In this chapter, we have reviewed the works that are related to EHR, cloud computing, data at rest, data mobility, data in processing and associated scenarios. From EHR and cloud computing, we moved to data security in cloud and then specify security for EHR in a cloud environment. By investigating data at rest, we examined and identified security and privacy in data mobility where relatively little research has been done. Data is potentially compromised when moving among clouds where the new cloud does not have the equivalent protection schemes as the original cloud. Furthermore, we presented an overview of current big data models and related work associated with the models. Hence, we pointed out the differences of our proposed schemes. We reviewed trust on cloud resources in processing for big data. So far, most of the study relies on hybrid clouds mechanisms and strategies mainly focus on processing sensitive data at private clouds. Thus, only very limited attention has been paid to cloud resources at public clouds or other private clouds that are trusted. Finally, we present a literature of Fog computing and its issues related to security and scheduling resources. In the next chapter, we will describe our previous work on data at rest and the data protection framework that is undertaken in this study.

CHAPTER 3 A TRUST-ORIENTED DATA PROTECTION FRAMEWORK AND MANAGEMENT

The data is passive when it is created and stored. In order to make effective use of cloud services, users' data is stored in a cloud where applications are able to perform more cost-effectively requests from clients. As data is transferred to the cloud, data owners rely on CSPs' security mechanisms to protect their data. These protection mechanisms are called passive data security protections. They cover the actions and mechanisms taken to monitor and audit data manipulation and status based on third party security services (Foster et al., 2008, Zhang et al., 2011, Tan et al., 2012). The trust-oriented data protection framework proposed an active data protection-centric which adds various properties of protection. The active data is considered as a secure data container that manipulates and verifies the data without the involvement of a third-party service. Consequently, the data is empowered with capabilities of self-defend and self-protect against intrusions or violations.

The research in this thesis is based on our trust-oriented data protection framework to develop a data mobility management model, a data distribution model, and a data processing model. In this chapter, Section 3.1 introduces the structure of active data. Section 3.2 introduces the supervisor. Section 3.3 presents the features of the Trust-Oriented Data Protection framework proposed by our research group. Section 3.4 introduces briefly our new proposed schemes and models. Section 3.5 describes the proposed deployment of Trust-Oriented Data Protection Framework for EHR protection in cloud environment. Section 3.6 presents the summary of the chapter.

3.1 Active data cube model for EHR protection

This section introduces our active data cube model for encapsulating an EHR for housing and

protecting data in a Cloud environment.

As introduced in Section 2.1, EHRs can be classified into *unstructured data* and *structured data*. In the context of cloud computing, *structured data* management typically interfaces with data by using secure connection interfaces. *Unstructured data* strongly relies on third-party security mechanisms or encryption. Once third-party services are compromised, *unstructured data* would be vulnerable to violation and tampering. In our work, we present protection mechanisms for *unstructured data.*

Instead of paying attention to attacks or violations, we focus on the target data and equip it with self-describing and self-defending capability. We introduce an ADCu structure which encapsulates both sensitive data and verification mechanisms to monitor the inner data and the derivatives. Any violations of policy or tampering with data would be compulsorily recorded and reported to data owners via the notification mechanisms within ADCu. ADCu triggers the log collection process to record every access to the specific data elements. Thus, it provides transparency and accountability of data usage by disseminating the log information to data owners.

As shown in Figure 3.1, our proposed ADCu consists of a *shell* and a *core*. The *shell* acts as a starting point to access the *core* when the ADCu is triggered. It is structured with an active form and tamper-proofing codes . The ADCu is associated with a runtime environment.

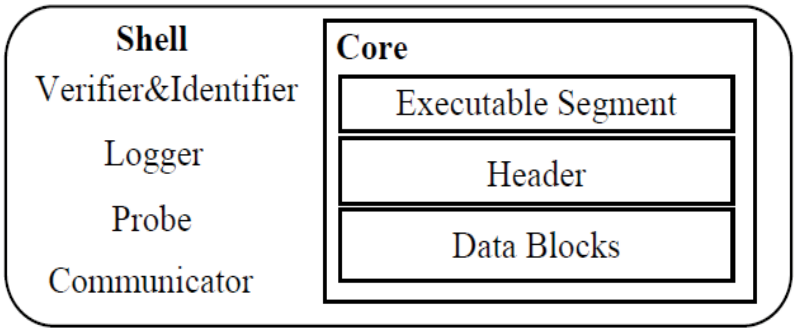


Figure 3.1 Structure of an ADCu (Chen and Hoang, 2012)

The *shell* is the entry point for all requests to access an ADCu. It then activates the *verifier and identifier* to verify request parameters. A permitted request issues a certification which is authorized by the CPRBAC service. The *logger* module in *shell* records main checkpoints when transactions are executed. The *logger* is required to record significant intermediate information. 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 ADCu'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 to take into account the fact that cloud resources may be distributed across different time zones. Each ADCu's log information is transparent to its data owner. When the log records are stored in the cloud, the RSA encryption is employed for these logs to prevent possible information leakage. These records can only be retrieved by a user who has the corresponding key. These logs are sent to cloud servers to reduce the cost of storage.

Each ADCu has a corresponding *supervisor* deployed in the same domain, which acts as a monitor for external data operations (such as move, and copy). This is because these operation cannot be detected by the internal probe inside the ADCu. If the ADCu cannot establish a proper network connection or cannot contact its supervisor, it would switch to the termination state to avoid the 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 the zero-knowledge proof procedure.

Once the verification and identification procedure succeeds, the *shell* delegates control to the data *core*. The *core* of ADCu is wrapped by an Executable Segment (ES), a header, and data blocks. The *shell* triggers the *core* and executes data operations via the ES. The *header* is presented by a manifest including the basic information of supplementary data residing at the beginning of *data blocks*.

3.2 Supervisor

The *supervisor* is designed to monitor OS-level data manipulation that cannot be detected by the ADCu. It can actively detect OS-level operations such as copy, move, and remove on the ADCu via a verification monitor. Thus, it can activate the ADCu to execute a runtime environment analysis and inspect the validity of data operations. The supervisor is responsible for protecting its associated data, data locations, and the communication among clouds. Figure 3.2 presents the design of a supervisor which monitors multiple ADCu in the same working domain.

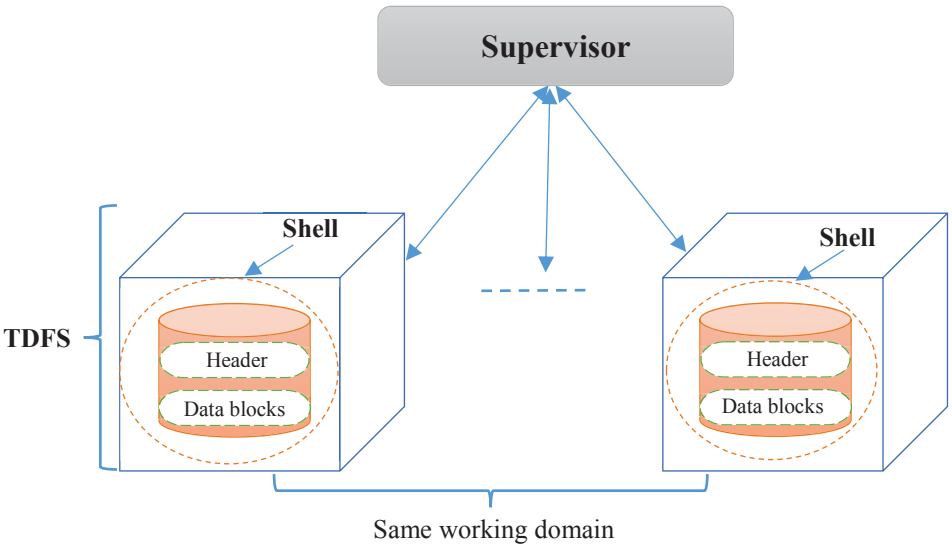


Figure 3.2 The design of a supervisor (Chen, 2014)

3.3 Trust-oriented data protection framework

In an earlier work, we proposed a Trust-Oriented Data Protection Framework for data protection in cloud environments (Chen and Hoang, 2012). Figure 3.3 illustrates the proposed trust-oriented cloud data protection framework. The framework is structured into three blocks: the data security control block provides secure access control and data auditability functions, the data core protection block provides techniques and procedures for implementing active data protection, and the data operation and management block handles mobility and data replication management. This framework can be considered as a secure data container that manipulates and verifies the data without the involvement of a third-party service. This structure presents a security and trust abstraction at a higher level than conventional data protection models that rely on the peripheral security environment and third-party protection mechanisms. Our core goal is to empower data with the capabilities of self-defense and self-protection against intrusions or violations. Data misbehavior and violation can be actively detected by the data itself, reducing the risk of use by adversaries. The data core protection block employs an active security approach whereby the data is made active against any invocation, whereas the data security control block and data operation and management block support this active approach by providing data auditing and other measures including secure access control, data replication and mobility.

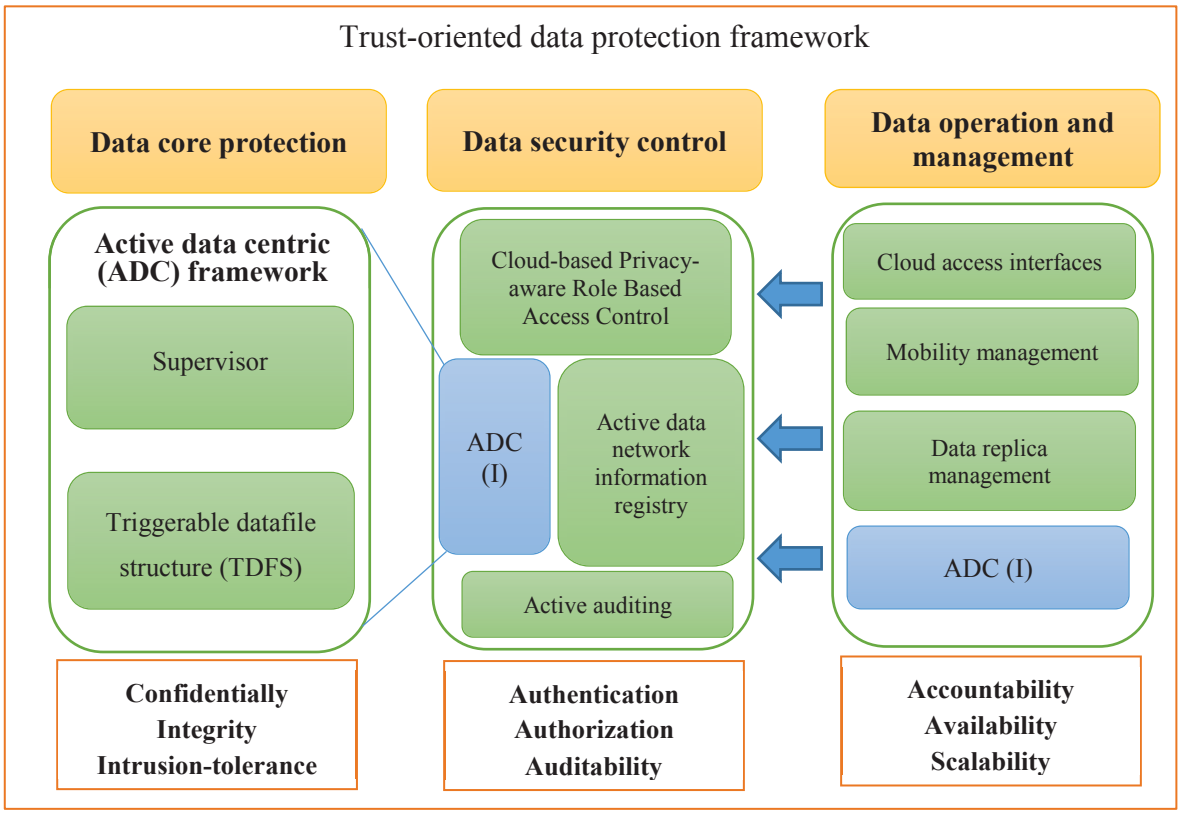


Figure 3.3 Trust-oriented data protection framework (Chen, 2014)

The management and coordination of cloud data for each tenant are processed by the supervisor, an active service instance that is activated when the corresponding tenant subscribes to cloud storage services.

The Data core protection: A unit of raw data is transformed into a novel data structure called a Triggerable Data File Structure (TDFS). A TDFS (Figure 3.4) is also referred to as an ADCu. The core of the TDFS comprises an executable segment, a header, and data blocks. The runnable scripts in the executable segment allow basic data operations, data loading, and data analysis functions. The header refers to a manifest specifying the basic information related to the data such as security header, data descriptor, and timestamp. Raw data blocks are encrypted.

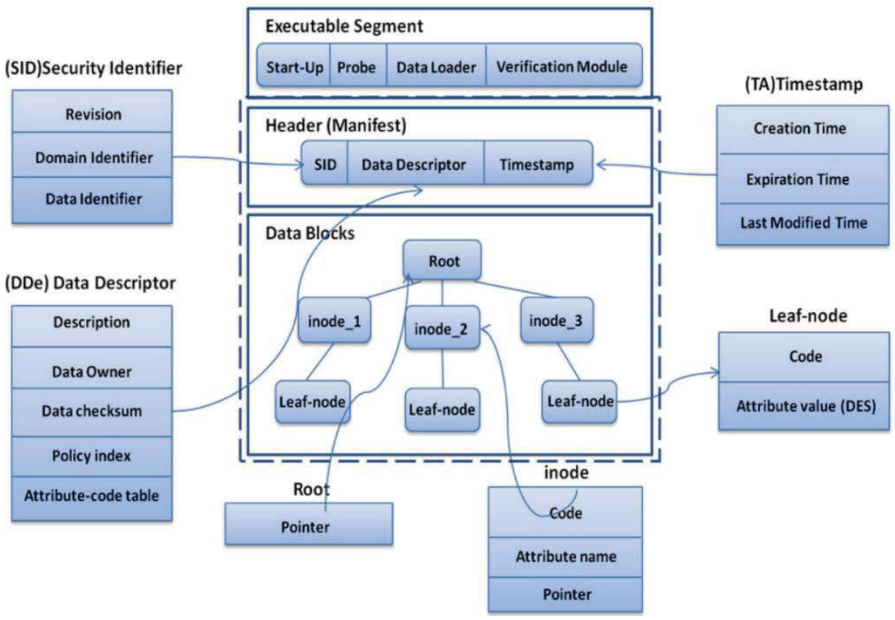


Figure 3.4 The TDFS structure (Chen, 2014)

The Data security control: this component is for executing trusted security management. The CPRBAC service (Chen and Hoang, 2012) is proposed to define and describe the security boundary on data operations in distributed clouds. Access resource requests that are not specified in the policy database will be rejected. The fine-grained policy structure of the CPRBAC allows users to configure and define specific and secure protection requirements on their data. Authentication and authorization will be offered by the service. The Active Auditing Control (AAC) (Chen and Hoang, 2011b) 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. Through recording audit data created as the attestations, the CSPs can report the evidence of data violations to their users. The users are more inclined to adopt the cloud solution for their businesses as they can establish more acceptable SLA with their subscribed CSPs in a firmed trustworthy relationship. The auditability can be achieved by the AAC.

Data operation and management: Cloud access interfaces provide data service interfaces to access active data in cloud storage systems. It forwards requests with parameters to security management component to verify access permission.

The work on the Trust-oriented data protection framework has been established and carried out by our research group. In this thesis, we extend the Trust-oriented data protection framework intensively and establish new schemes and modes for data handling. As a result, the data structure will be changed to a suite with new design models. A brief description of the new

models and schemes will be presented in the next section.

3.4 Enhancing Trust-oriented data protection framework for data management

Figure 3.5 presents our work in this thesis based on a Trust-oriented data protection framework. Even though we use the framework, we extend it intensively to address the identified research issues. In particular, we propose the data mobility management model, data distribution scheme, data replication scheme, data processing model to address requirements related to data security and privacy, data distribution, and data processing issues in a cloud environment.

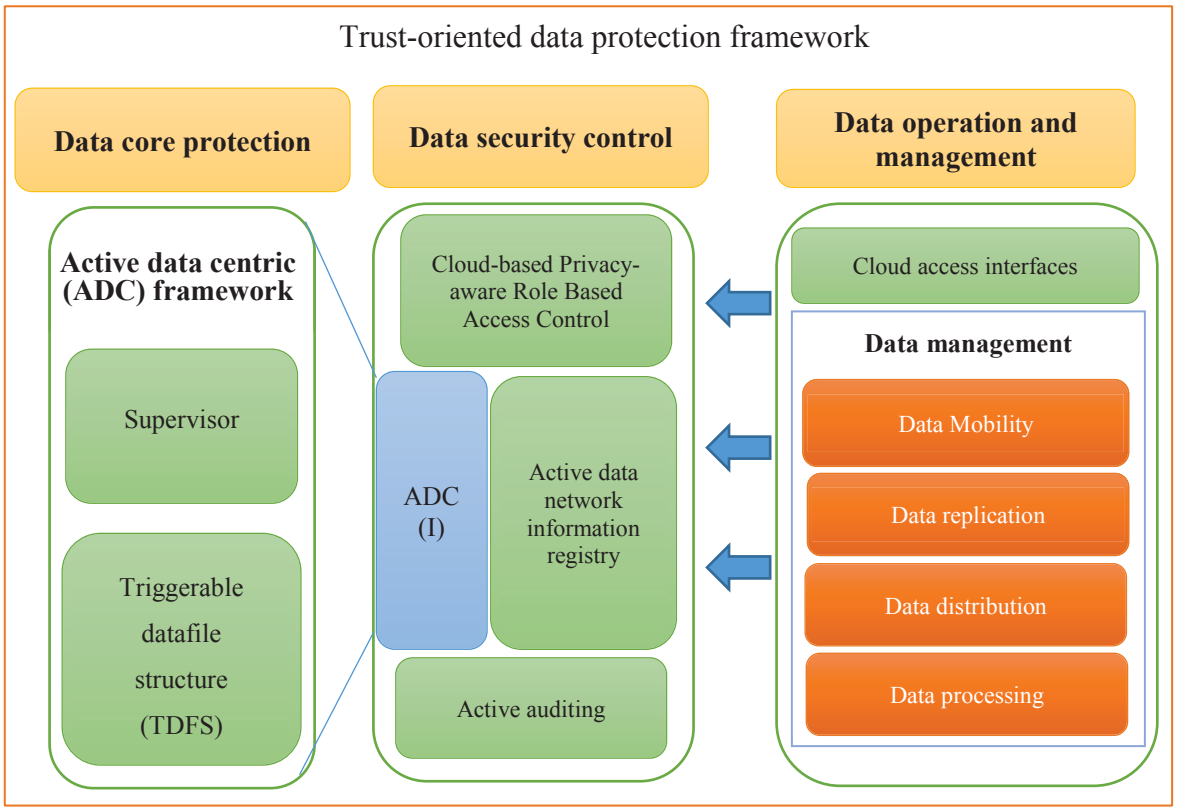


Figure 3.5 New proposed schemes and models based on the Trust-oriented data protection framework

The data mobility management model addresses security and privacy issues related to the movements of user's data among clouds. The model ensures users' data is protected in an integrated manner regardless of its location in a cloud environment. When user's data is moved to data centers located at locations different from its home, it is necessary to keep track of it

and take care of it by a user provider's SLA. The reporting service can be achieved through an agreement that enables a monitoring service from the original cloud. When a violation against the established SLA occurs, the monitoring component will be able to detect it through the corresponding CSP and to trigger protection services on the original cloud which can immediately analyze and audit the data. Moreover, data locations need to be maintained at the home cloud and encoded within the data itself in case it loses the connection with its monitoring service.

In this model, we extend our active data framework with a new location data structure and a Location Registration Database to deal with mobility; we investigate protocols between clouds for data movement; we investigate procedures for establishing a proxy supervisor at a visited cloud for monitoring purposes; and we propose a novel MS to handle requests for moving the data among clouds. The proposed data mobility model focuses on two aspects: Data Mobility Management (DMM) and active protection service. The DMM deals with physical location changes of user's data in various cloud environments and ensures these locations are registered with LRD and recorded within the data itself. We also present a comprehensive analysis of data mobility scenarios in various Cloud environments. The design and the implementation of the proposed data mobility management model are carried out together with the evaluation of the implemented model. Details of these components will be presented in Chapter 4.

The data distribution scheme provides significant improvements in providing better healthcare services particularly in the big health data distributed cloud computing environments. The process of provisioning healthcare involves massive healthcare data which exists in different forms (structured files or unstructured data) on disparate data sources (such as relational databases, file servers) and in different formats (text, images, sensor data, XML files, relational database records). Current big data processing models do not handle well these types of data in term of sizes and numeric. P2P search is efficient and has been used in many real systems. However, for a system with a very large number of files relative to the number of storage nodes and if the access pattern is skewed, the data is spread to various nodes resulting in less efficient retrieval and lookup processes. This data distribution scheme addresses the two major concerns from a novel perspective: clustering EHR files in a controlled manner into a defined number of nodes to minimize the number of steps in searching for a requested file and reducing the data lookup latency among peers. Our work proposes a novel HBFC scheme to distribute, store and retrieve EHR efficiently in cloud environments. The HBFC possesses two distinctive features: it utilizes hashing to distribute files into clusters in a controlled way and it utilizes P2P structure

for data management. We are not aware of any existing methods that use "collision" to control the formation of clusters as designed in our scheme. The proposed scheme will be presented in Chapter 5.

The data replication scheme provides high availability and reliability of data storage service in case of nodes failure and network reconfigurations. In particular, when the number of replications distributed at nodes directly affects the workload performance of a distributed system, the scheme supports a dynamic adjustment mechanism to balance the distribution of data. The data replication scheme is designed based on HBFC scheme from a novel perspective. It utilizes HBFC to establish data distribution balance in P2P systems by using two hashing rounds, one for efficient data clustering and the other for leveraging the P2P search mechanism. The proposed scheme can achieve better workload balance and storage efficiency than random replication (refers to replicate data replication randomly at nodes) and status replication (refers to the fixed and static replicate data replication at nodes) which might result in hotspots. Details of the scheme will be presented in Chapter 5.

The data processing introduces two processing models namely a trust-based scheduling model for big data processing with MapReduce and Fog-based region model. The former addresses scheduling cloud resources issues to process users' sensitive data for big data applications. The later deals with low latency response requirements by proving nearby computing resources for applications.

The trust-based scheduling model introduces a composite trust metric and the method to determine the required trust of tasks and the trust value of given cloud resources. A hybrid cloud was used in (Xu and Zhao, 2015, Zhang et al., 2014) to preserve data privacy by processing sensitive data at private clouds only while non-sensitive data was processed at public clouds. It is not desirable, however, to schedule all processing tasks for sensitive data at only private clouds while public clouds can provide adequate protection services but they remain idle. The systems may incur bottlenecks or longer delays when the amount of sensitive data is huge or data operations are heavily performed at private clouds. The proposed scheduling scheme for MapReduce matches tasks with adequately trusted cloud resources. The optimization strategy is carried out to gain the highest trust value when performing tasks. The trust-based scheduling framework for realisation is also proposed. A prototype of the trust-based scheduling model is simulated based on the MapReduce simulation tool to evaluate the feasibility of the system and the effectiveness of the proposed scheduling strategies. The proposed model with security

constraints for protecting data privacy improves system performance significantly and efficiently utilizes computation resources from both private clouds and public clouds while ensuring users' data is executed at highly trusted resources. Details of the scheme will be presented in Chapter 6.

The Fog based Region model introduces a local computing concept called "Region" to deal with these issues. The deployment of computing resources and data at regions shall provide users better experiences based on nearby resources. To this end, the novel data protection model for Fog computing provides region-based trust establishment, MS, and FPRBAC. It enables fog devices in different regions to share and access resources in a secured manner. In fact, the MS enables clients to keep track of changes of data location among regions periodically by using a LRD as well as an enhance verification procedure at FPRBAC. As a result, it tightens security constraints while ensuring more flexibility in mobility management. The Region concept is also applied to address requirements of application's latency-sensitive in Fog computing and in combination with cloud computing to provision computation resources on demand. Hence, tasks are performed at not just a single region but multiple regions and/or possibly cloud servers in order to minimize its completion time. The resources of a region are, however, limited and their availability varies. Some tasks may be processed in one region for the faster response but others may have to be distributed and executed over multiple regions or even at remote cloud servers as they have more computational resources. Although more computation resources result in shorter processing time, data transmission between them and their users leads to higher latency. The FBRC introduces an efficient schedule mechanism for tasks at both regions and cloud servers to minimize the computation and transmission latency of all requests. Details of the proposed model and scheme will be presented in Chapter 7.

3.5 Securing EHRs in the cloud with the Trust-oriented Data Protection framework

It is anticipated that EHRs and healthcare-related services will be deployed on cloud platforms to reduce the cost and complexity of handling medical records while improving efficiency and accuracy. However, both users and healthcare service providers are slowly to adopt the integration of EHRs in cloud environment due to users' concerns relating to the data control. Thus, it is essential to adopt a secure deployment framework for EHRs in the outsourced environment to protect sensitive information contents of health records.

Figure 3.6 briefly describes how our proposed Trust-Oriented Data Protection framework can be deployed for specific EHR protection. Originally, EHRs were stored in HIS which provided the access for users via portals. In the past, HIS were deployed in traditional factions where EHRs were housed by physical servers in healthcare organizations. Hence, security mechanisms are deployed in these infrastructures. Although transferring these EHRs into clouds allows users high availability and scalability, users had no other choice but to rely on CSPs' security approaches in protecting their data. With our proposed trust-oriented data protection framework, an EHR is not only protected by CPRBAC and AAC but also transformed in an ADCu before uploading it to cloud storage. ADCu offers self-defend and self-protect features to incorporate with both CPRBAC and AAC services for protecting users' data.

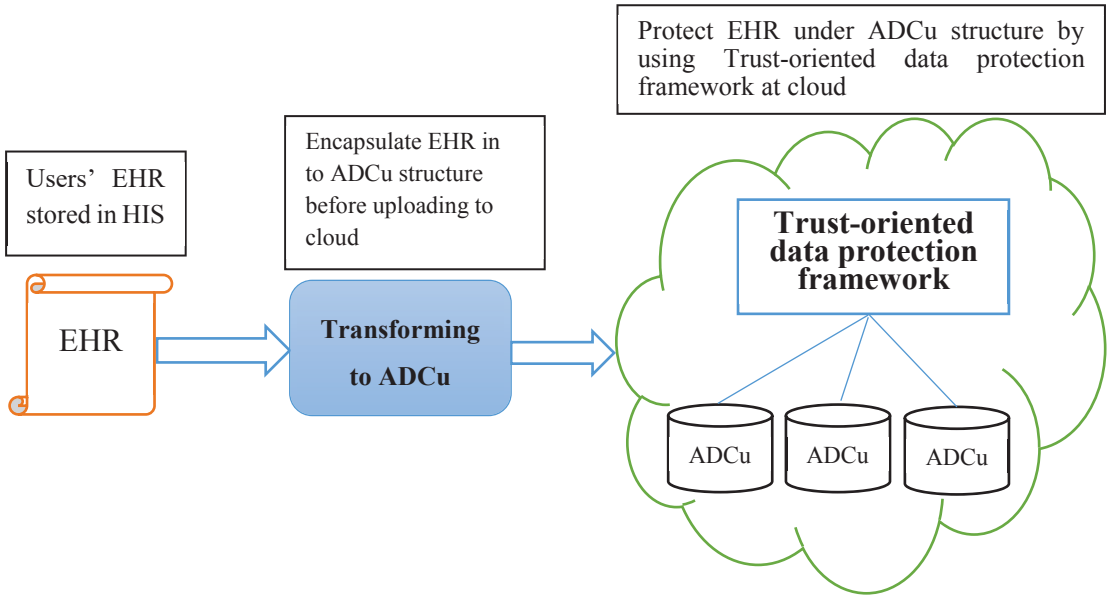


Figure 3.6 Workflow for securing EHR with Trust-oriented data protection framework at cloud

3.6 Summary

This chapter has given an overview of the Trust-oriented data protection framework in our previous work, and our new proposed models and schemes. The proposed models are based on the trust model but the underline of the work is security. In fact, we addressed security issues intensively. On data at rest, EHR is encapsulated in ADCu, which is empowered with capabilities of self-defend and self-protect against intrusions or violations. The Active data

framework employs Role-based access control for authorizations, whereas EHR is encrypted to prevent illegal disclosures or attacks to sensitive information. On data mobility, data mobility management model ensures data moving within and among the original cloud and visited clouds securely by observing confidentiality and availability. On processing, the trust-based scheduling scheme for big data processing provides a mechanism to ensure users' sensitive-data to be executed on high trusted resources.

We first introduced the structure and features of active data, enabling the capabilities of self-defense and self-protection against attacks. Next, we introduced the supervisor. Then, we presented the Trust-oriented data protection framework including three main layers: the data core protection layer, data security control layer, and data operation and management layer. Following that, we introduced our new proposed models and scheme for data handing. The work is based on the Trust-oriented data protection framework which was carried out by our group previously. Finally, we described the deployment of EHRs in the cloud with trust-oriented data protection framework in a secure manner. In the next chapter, we will present the novel data mobility management model that is undertaken in this study.

CHAPTER 4 DATA MOBILITY MANAGEMENT FOR CLOUD

Protecting data in the outsourced cloud requires more than just encryption (Juels and Oprea, 2013) which merely provides data confidentiality, anti-tampering, and anti-disclosure. The key to mitigate users' concern and promote a broader adoption of cloud computing is the establishment of a trustworthy relationship between CSPs and users. For users to trust the CSPs, users' data firstly should be protected with confidentiality maintained and no one should be able to disclose data, which is sensitive information, except the authorized users. Secondly, any actions or behaviors on the data should be enforced and recorded as the attestation to avoid false accusation of data violation. Once a breach against the SLAs subscribed between CSPs and users occurs, the attestation can be used as a proof that the CSPs have violated the agreed-upon service level and consequently, appropriate compensation may be offered to the users. Finally, the data should be able to preserve itself independently in a heterogeneous cloud environment with diverse protection frameworks.

This chapter discusses the above concerns from a novel perspective, offering a data mobility management model with enhanced transparency and security. The data mobility model focuses on two aspects: Data Mobility Management and active protection. The DMM deals with physical location changes of user's data in various cloud environments and ensures these locations are registered with the LRD and recorded within the data structure itself. The structure of this chapter is as follows: Section 4.1 presents the motivation of data mobility in cloud. Section 4.2 presents components for a data mobility model in cloud. Section 4.3 discusses data mobility scenarios. Section 4.4 presents the design of the data mobility management model. Section 4.5 presents the implementation of the mobility management model. Section 4.6 discusses the evaluation of the mobility management model. Section 4.7 provides the discussion about the improvement for our proposed model in the future. Section 4.8 summaries the chapter.

4.1 Data mobility

Even with the Trusted-Oriented Data Protection Framework (Chen and Hoang, 2012), the data is still exposed to potential violations when it is moved to new cloud hosts where there are no equivalent security measures to protect it. Data mobility is still a challenge for exchanging information among CSPs due to the lack of models to ensure data protection and data auditing.

Clearly, mobility management is one of the most important challenges in mobile IP (Jae-Woo, 2007). When a mobile is roaming away from its home network, it registers its current location to its home agent on the home network. Similarly, when a data unit moves from its original cloud, similar mechanisms should be provided for the data to inform its original cloud of its current locations and/or the data owner of its status if necessary. The data itself, however, cannot execute these actions. For cloud data mobility, we leverage the ideas from mobile network about location register for a mobile by using a LRD located at original cloud for updating or retrieving data locations and a recordable data structure for recording cloud host location in the data itself when there is a data request operation. Moving data to a new cloud environment, however, has to involve both the data and its supervisor (in cellular networks only the mobile phone is involved). In this model, a clone supervisor is established for monitoring the moved data and data operations. A verification procedure will process data requests at the original cloud for access permissions; however, the mobility management model may also delegate the verification and access control to the visited cloud. With the deployment of the supervisor, data protection can be achieved despite the fact that data is located at visited cloud side.

4.2 Components of a data mobility model

We define functions of parties involved in the mobility management model in Table 4.1.

Table 4.1. Terms and description of components in the data mobility model

Terms	Descriptions
Original cloud	Providing cloud resources and verification procedure
Old cloud	An original cloud or a visited cloud offering cloud resources
New cloud	Creating a request to move the data with appropriate parameters

Visited cloud	A new cloud or an old cloud which stores, forwards data verification requests for permissions
Supervisor	Protecting its associated data, data locations and the communication among clouds

- *Data verification:* The supervisor monitors data operations either at the original cloud or at the visited cloud. Data only accepts instructions from its supervisor. For monitoring function, the supervisor focuses mainly on detecting user's operations at the operating system level such as move, copy and delete. When the supervisor detects such a request, it would activate the TDFS through instructions to execute the runtime environment analysis and inspect the validity of data operations. In the proposed active data model, data operations will be verified at the shell which wraps the data in an executable status.
- *Data location management:* The supervisor also reports timely data locations to the original cloud as well as updating the LRD since data itself cannot send location information to original cloud but it is able to record current location for tracing operations. It analyses networking location in order to obtain the current network address and send back to original cloud which is set as default source address in the report message.

4.3 Data mobility scenarios

It is assumed that the new cloud and the original cloud agree to establish a new supervisor at the new site. Since the verification is processed at the original side, the new clone supervisor is used for dealing with data moving cases among clouds. In this chapter, a TDFS is used as data objects move among clouds. Data mobility in a cloud environment involves several cases. In the first case, data is moved from an original cloud to a new cloud (the request may be originated from the original cloud or from the new cloud). In the second case, data at an old cloud is moved to a new cloud and original cloud will process request verifications. In the third case, data at an old cloud is moved back to the original cloud; and in the last case, data at a cloud is moved to local storage (offline) with or without permissions.

4.3.1 Moving from home cloud to a new cloud

In the first case, data is moved from an original cloud to a new cloud (the request may be originated from the original cloud or from the new cloud). In both scenarios, a general procedure (Figure 4.1) has to be executed between the two clouds before moving the data. The procedure is as follows:

- After analyzing the request, the original cloud sends a request to the new cloud for establishing a new supervisor at the new cloud. This request is then verified by the visited cloud for the permission to install a new service. At this time, the new cloud also verifies the request to evaluate wherever it can or cannot create the service. If there is an agreement between two clouds to create the new supervisor, a confirmed message will be sent from the new cloud to the original cloud.
- Following the first step, information pertaining to the supervisor and the TDFS including the type of services, the template of the supervisor, and the original location will be sent to the new cloud for creating a new instance of the supervisor.
- After the supervisor is created, the TDFS will be moved to new cloud.

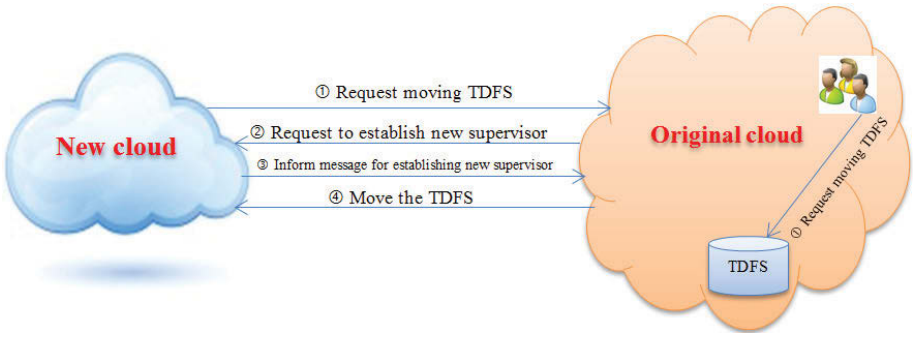


Figure 4.1 General establishing supervisor procedure

The new supervisor will be responsible for monitoring the TDFS at the new cloud as well as communicating with the original cloud since the new cloud does not provide the same data protection services.

If the request originated from an entity in the original cloud, the destination address of the new cloud where data is moved to has to be provided with the request. When the destination address is identified, the original cloud can communicate with the new cloud. In both scenarios, security procedures are performed by the access control component, the CPRBAC, and the

auditing component, the AAC together with the associated supervisor. Having satisfied the conditions for moving the requested data, the original cloud sends a request in order to establish the clone supervisor at the new cloud. By doing this, the link between the supervisor and its data is still kept when data is stored at another location. Details of the procedure (Figure 4.2) are as follows:

- The service request from the new cloud or from within the original cloud is sent to the original cloud. This request is authorized by the access control and the auditing control components of the original cloud. If it is a valid request having the correct required parameters, the supervisor is triggered to analyse data operations. Since it is a "move" request, the supervisor has to communicate with mobility component for establishing a clone supervisor at the new cloud. Invalid requests against predefined policies will be triggered by the ADCu and captured by the auditing component. Violation triggers in the auditing may also be triggered to assist the security administrator to execute some security prevention operations.
- The mobility component at the original cloud sends a request to the visited cloud for the permission to install a new service. The new cloud also verifies and evaluates the request to see if it can create the service. If an agreement is in place to create a new supervisor between two clouds, a confirmed message will be sent from the new cloud to the original cloud.
- At this step, necessary information for creating the clone supervisor will be supplied to the new cloud.
- Once the new supervisor is created, the mobility component invokes instructions to move the data. Hence, the new location of TDFS is also updated at LRD.

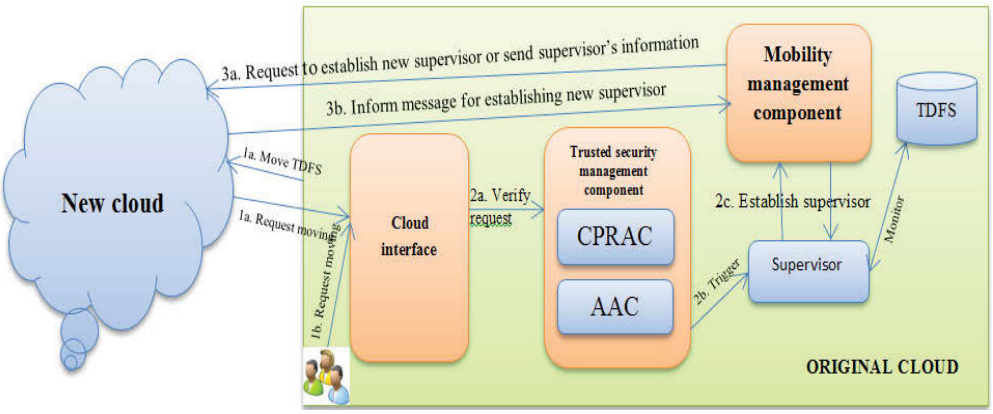


Figure 4.2 Details of establishing supervisor procedure

4.3.2 Moving from an old cloud to a new cloud

In the second case, data at an old cloud is moved to a new cloud and original cloud will process request verifications. The procedure is similar but the verification request needs to be forwarded from the old cloud to the original cloud for authorization. After analysing the request, the original cloud sends a request to the new cloud for establishing a new supervisor at the new cloud. Once the new service is established, the original cloud will inform the old cloud to move TDFS. Otherwise, original cloud will invoke a message for old cloud to terminate the request. Figure 4.3 depicts procedures of the second data moving case. The procedure for this case is as follows:

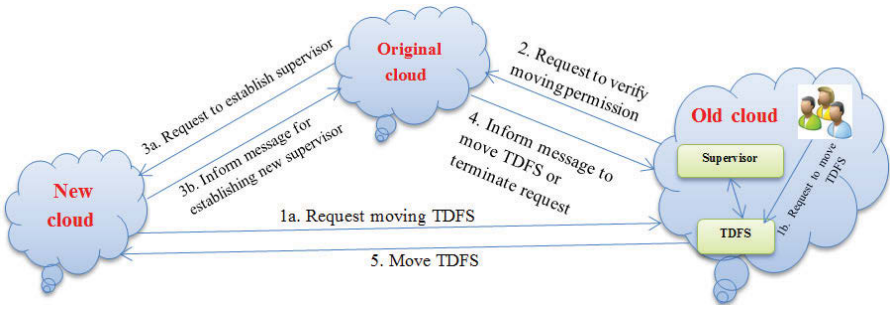


Figure 4.3 General procedure in establishing supervisor from old cloud to new cloud

- The request from the new cloud or from within the old cloud is sent to the old cloud to move TDFS. After analysing this request, the old cloud forwards it to the home cloud for authorization.
- Following the first step, the home cloud sends a request to the new cloud for establishing new supervisor at the new cloud. This request aims to ask new cloud for the permission to install a new service. At this time, new cloud also verifies the request to evaluate wherever it can or cannot create the service. If there is the agreement to create a new supervisor between two clouds, then a confirmed message will be sent from new cloud to home cloud. Then, information about supervisor and TDFS including the type of services, the size of supervisor, original location will be sent to the new cloud to serve for creating a clone supervisor.
- Once the new supervisor is created, the home cloud invokes a message to old cloud for moving TDFS. Hence, the new location of TDFS is also updated at LRD.

The general procedure has been described for this case. In order to provide more details about the procedure, we establish detailed steps (Figure 4.4) as follows:

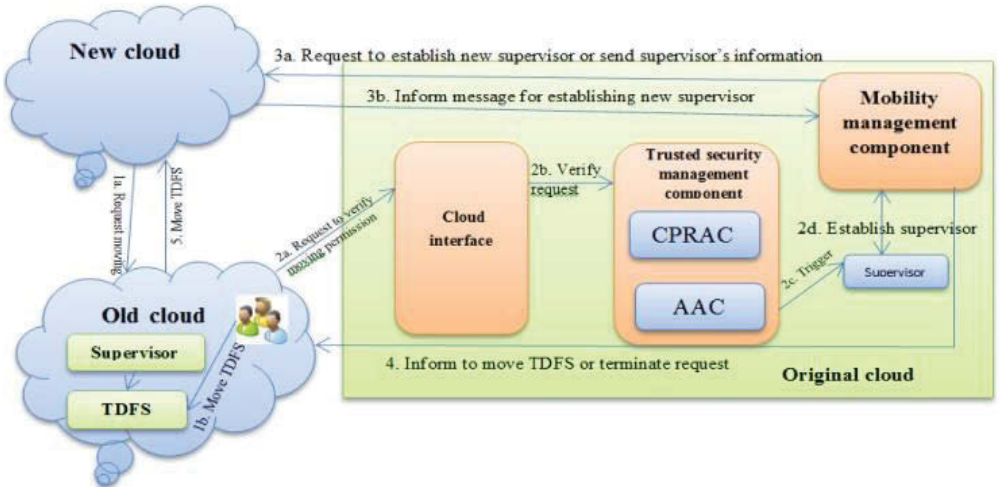


Figure 4.4 Details of establishing supervisor procedure from old cloud to new cloud

- The service request from the new cloud or from within the old cloud is sent to the current cloud to move TDFS. This request is then forwarded and authorized at the original cloud. At this stage, the same procedure as the first case will be performed in order to establish a clone supervisor at the new cloud.
- After supervisor is created at the new cloud, the original cloud invokes a message to the old cloud for moving TDFS. Hence, LRD is also updated on new location of TDFS.

4.3.3 Moving back to the original cloud

In the third case, data at an old cloud is moved back to the original cloud; and in the last case, data at a cloud is moved to local storage (offline) with or without permissions. Therefore, the procedure for establishing supervisor is not performed at this stage, but the moving request has to be authorized at the original cloud in order to approve for the permission. The procedure for this case is as follows:

- The service request from the original cloud or from within the old cloud is sent to the old cloud to move TDFS. At this time, the original cloud also verifies the request to evaluate wherever it can or cannot allocate the data at its storage. A confirmed message allowing or rejecting will be sent from the original cloud to the old cloud.
- If the request access is permitted, the new location for TDFS will be updated at LRD.

For the last case, the regular verification procedure is still executed at the original cloud to obtain the move permission. However, an exception will be raised when the original cloud detects a data move without a request or it cannot communicate with destination address

provided in the request to establish supervisor. In this case, the current supervisor has to report the current TDFS location to the original cloud for updating the LRD as well as triggering the data to update this address in its core component before moving. From this point, the TDFS has to record data access locations in its location list whenever users access the data even in the offline mode. The moving operation occurs when TDFS is stored at the original cloud or at the visited cloud. Figure 4.5 depicts procedures of the last moving case.

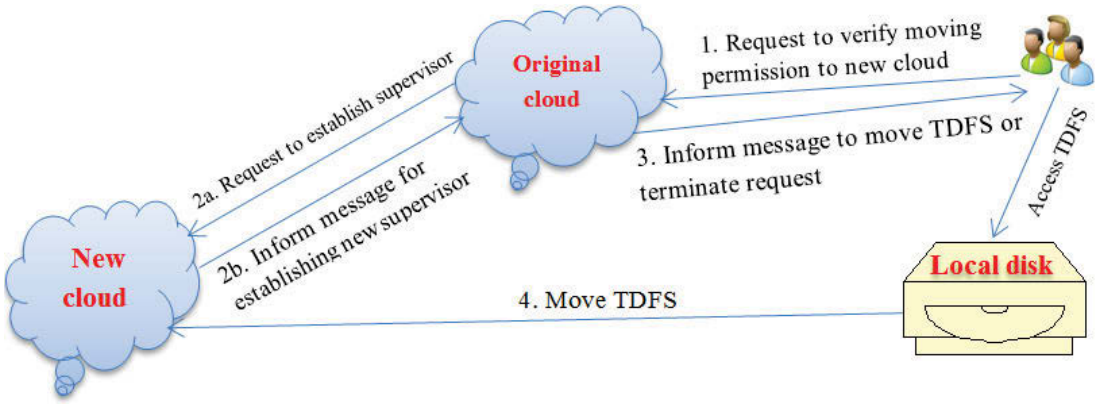


Figure 4.5 Details of processing moving request from old cloud to original cloud

4.4 Mobility management model

In this section, we present the design of our data mobility management model. The mobility management model enhances the mobility of ADCus and their protection when deploying in a cloud environment. It supports the management of data and data locations as well as ensures protection data at different clouds. The model is designed to achieve the following objectives:

Verification, authentication and authorization: Requests are verified with defined policies to access the resources. The fine-grained policy structure allows users to configure and define more specific and secure protection requirements on their data. Furthermore, these requests are also audited and recorded as audit data in which evidence of data violation will be reported to the data owner.

Mobility service: In order to deal with the changes of data location, the MS is introduced to execute users' requests for updating, tracking data locations at LRD and informing data owners with data operations and new locations. This addresses users' concerns when allocating their data within cloud services.

Location register: When data is moved to a new cloud environment, it has to nominate its presence at the new cloud to obtain cloud services within SLAs established at its original cloud. In addition, with the introduction of the LRD, locations of data are also collected and maintained in order to serve for tracking, retrieving and monitoring data status.

Data monitoring: Data operations on the data are monitored by a supervisor at an original cloud or a clone supervisor at a visited cloud to prevent data violations against the SLAs subscribed between CSPs and users. The supervisor detects these operations and triggers its TDFS into an active state ready for executing the self-protection process.

Data protection: The data is structured within an active protection structure which enables it to be monitored, which triggers smart analysis, self-protection, and self-defense capability directly upon external requests. The active structure also allows the recording of data operations with itself for tracing purposes.

We focus on providing a data mobility solution for cloud data moving among clouds while ensuring data protection, auditing relevant data locations and accessed data operations. The new features of our proposed model include: an active data framework with the appropriate data structure and the LRD to deal with mobility; protocols between clouds for data movement; procedures for establishing a clone supervisor at the visited cloud for data protection purposes. In particular, there will be a MS agent responsible for updating and retrieving data location from the LRD. Figure 4.6 depicts the model and its four core components: 1) the data core protection component, 2) the mobility management component, 3) the trusted security management component, and 4) the cloud interface.

The data core protection component: this component is designed to enable active surveillance, smart analysis, self-protection, and self-defense capabilities directly onto the data itself. The data is transformed and encapsulated in a TDFS that supports confidentiality, integrity, and intrusion-tolerance (Chen and Hoang, 2012). To support mobility, a special encrypted element namely Recordable Mobility Data (RMD) is designed to record user's operations when they access the data. Only the data owner who has the decryption key can trace previous data operations. The management and coordination of cloud data for each tenant is processed by the supervisor, whose active service instance is activated when the corresponding tenant subscribes to a cloud storage service. Several supervisor service instances can be deployed to deal with a large number of requests from diverse virtual servers or machines (VMs). An atomic active data protection unit (APDu) contains an active data cube (the TDFS)

and its supervisor.

The mobility management component: this component includes the MS and the LRD. It aims to store and manage information about the supervisor and the TDFS at the original cloud. The component centres around the location registration procedure when the TDFS is moved by maintaining connections with its responsible supervisor.

The MS is responsible for creating queries to the LRD. When the data is created, the supervisor invokes the MS to update the information about the TDFS in the LRD. In addition, the MS also supports the establishment of the new supervisor at the visited cloud.

The LRD stores the TDFS information related to data location, data operations, and data owner for data status monitoring purposes. The Visitor Location Register Database (VLRD) is located at the visited cloud and structured similarly as LRD with some additional fields presenting for the location of visited cloud. When a TDFS is subscribed to a cloud, it needs to register and is allocated a supervisor that is responsible for the data welfare including monitoring and raising the alarm if illegal data operations are detected. Therefore, whenever a TDFS moves out of its original cloud, the supervisor will invoke a query to extract information from the database necessary for the establishment of the clone supervisor at the new cloud. Apart from the data, there are also tables holding system data, including information about servers which are allowed to connect to the system. In the design phase, we designed database tables to achieve the following functions:

New data subscription: this function allows users to subscribe information about their data such as data owner and profile data.

Updating changes of location: this function supports the location update procedure when there are requests from the MS to update data locations. Despite the fact that data is stored at a visited cloud, its location is still updated if there are requests to move the data to a new cloud.

Retrieving VLRD lists: this function allows the LRD to locate the VLRD that holds the current location information of the TDFS so that the location management can be utilized.

Providing system data: this function allows the MS to access system data information about LRD and VLRD. As a result, the CSPs are able to identify cloud hosts and the LRD.

For TDFS location registration at a visited cloud, the VLRD is used for storing TDFS locations when the TDFS moves from its original cloud to a visited cloud. The VLRD is the other location register used to retrieve information about data location when data is stored at the visited cloud.

Because a TDFS can be moved anywhere in clouds of different infrastructures, mobility management is very essential. When a TDFS moves about, location registration for tracking and tracing purposes is always needed. Therefore, if visited locations of TDFS are stored and managed as a distributed database system, the MS at visited cloud can query directly the VLRD rather than the home LRD.

When a request to access the data at the visited cloud is permitted, the MS will insert or update the VLRD. If the TDFS location is not stored in the VLRD, the request is forwarded to the original cloud where the MS queries the LRD. When it is authorized, a new record will be added to the VLRD. When a TDFS visits a new cloud from the old cloud, the registration process in the new VLRD is as follows: 1) The MS sends a request to the new visited cloud in order to register its information in the new VLRD; 2) The new VLRD informs the MS's LRD of the TDFS's current location, the address of the new cloud; 3) The MS's LRD sends an acknowledgement including TDFS's profile; 4) The new VLRD informs the TDFS of the successful registration; 5) The LRD sends a deregistration message to the old VLRD and the old VLRD acknowledges the deregistration.

- ***The trusted security management component:*** this component is for executing trusted security management. The CPRBAC service (Chen and Hoang, 2011a) is proposed to define and describe the security boundary on data operations in distributed clouds. Access resource requests that are not specified in the policy database will be rejected. The fine-grained policy structure of the CPRBAC allows users to configure and define specific and secure protection requirements on their data. Authentication and authorization are offered by the service. The AAC (Chen and Hoang, 2011b) 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. Through recording audit data created as the attestations, the CSPs can report the evidence of data violations to their users. There is an increase in number of users adopting the cloud solution as they can establish more acceptable SLA with their subscribed CSPs in a firmed trustworthy relationship. The auditability can be achieved by the AAC.

- ***Cloud interface:*** Cloud interfaces provide data service interfaces to access active data in cloud storage systems. It forwards requests with parameters to security management component to verify access permission.

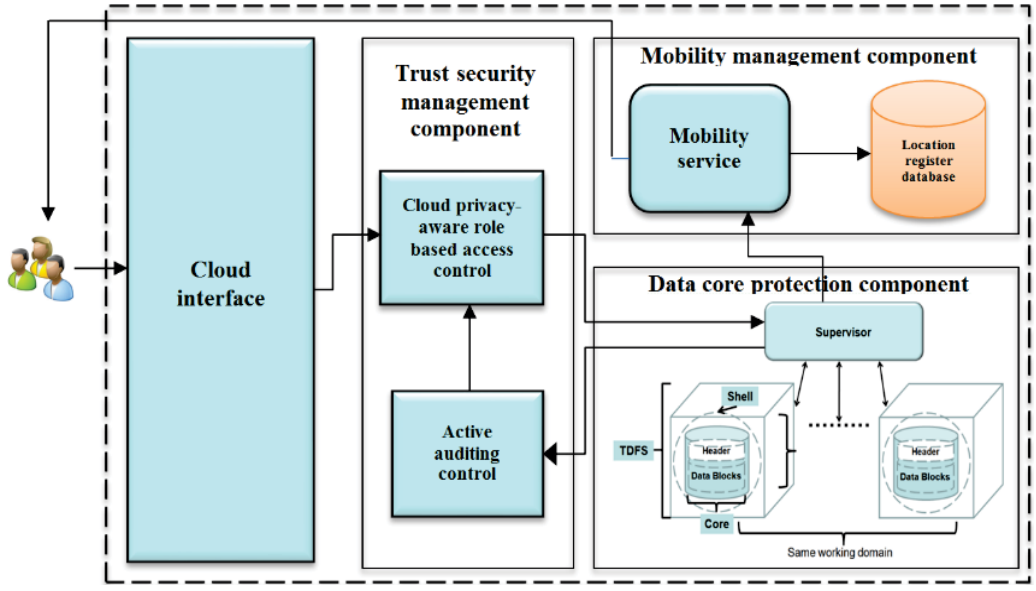


Figure 4.6 The design of data mobility management model

4.5 Implementation

4.5.1 Data structure design

Data is structured utilizing the Active Data-Centric framework (Chen and Hoang, 2012). A special structure called RMD is designed to record information associated with users' access request. The information includes Subject_ID, Data_ID, Operation, Time Stamp, and Cloud location. This information is transparent to its data owner. In other words, it is invisible from users' data operation. The stored information is encrypted using the RSA encryption to avoid possible information leakage. Only the data owner has the corresponding key to disclose them for tracing previous operations. The new TDFS structure is shown in Figure 4.7 and Figure 4.8.

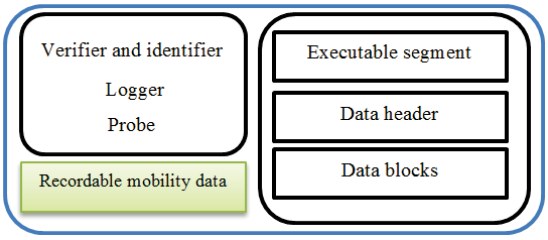


Figure 4.7 Triggerable Data Structure

Subject ID	Data ID	Data operation	Source address	Current address	Destination address	Time Stamp

Figure 4.8 Recordable mobility data structure

4.5.2 LRD design

The design phase had two tasks to complete; the first was to implement the design of basic entities in the LRD's schema and the second included the design of database tables (Figure 4.9). When a TDFS registers with the LRD, information associated with the TDFS is stored in the location registration table. For implementation, we only present the structure of the location registration table in the LRD. The primary keys of supervisor and data owner appear in this table for completeness. A new record, which is composed of six fields including TDFS_ID, Supervisor_ID, Operation, Address, Time Stamp and Data Owner, is inserted into the LRD as initial data location for the TDFS. Whenever a TDFS moves out of its original cloud, the supervisor will invoke the MS which will execute a query to extract information from the database necessary for the establishment of the clone supervisor at the new cloud. TDFS locations in the database are updated at the LRD to allow verifying and tracking of data.

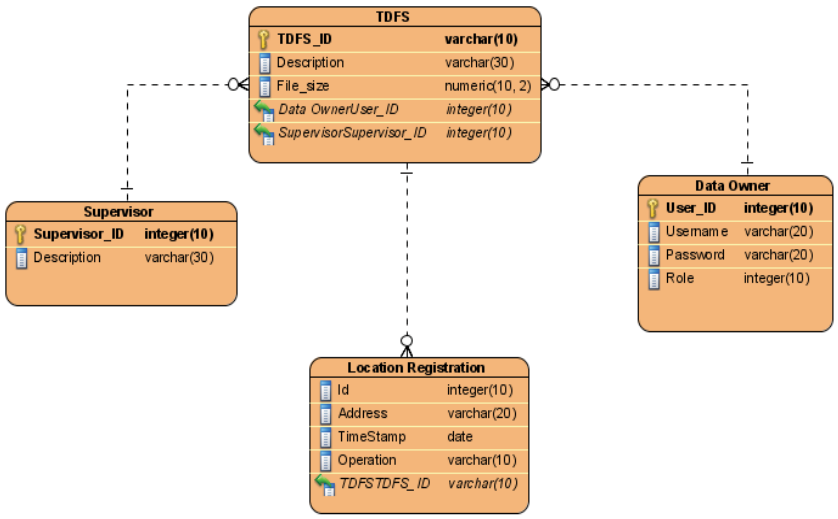


Figure 4.9 LRD design

We assume that the LRD manages n TDFSs. The set of TDFSs in the LRD is defined as $TN = \{tn_1, tn_2, \ldots, tn_i, \ldots, tn_n\}$; the set of associated supervisors is represented by $SP = \{sp_1, sp_2, \ldots, sp_i, \ldots, sp_n\}$, and the set of data operations on TDFSs at time t is represented by $OP = \{op_1, op_2, \ldots, op_i, \ldots, op_m\}$. Similarly the set of address frequencies, the set of registered time stamps and the set of data owners are represented by $IP = \{ip_1, ip_2, \ldots, ip_i, \ldots, ip_n\}$, $TS = \{ts_1, ts_2, \ldots, ts_i, \ldots, ts_n\}$ and $OW = \{ow_1, ow_2, \ldots, ow_i, \ldots, ow_n\}$ respectively. Let LRD table set $L = \{tn_i, sp_i, ip_i, op_i, ts_i, ow_i\}$.

Table 4.2 LRD's data table structure

TDFS_ID	Supervisor_ID	Operation	Address	Time Stamp	Data Owner
tn_1	sp_1	op_1	ip_1	ts_1	ow_1
tn_2	sp_2	op_2	ip_2	ts_2	ow_2
…	…	…	…	…	…
tn_i	sp_i	op_i	ip_i	ts_i	ow_i
…	…	…	…	…	…
tn_n	sp_n	op_n	ip_n	ts_n	ow_n

Regarding the registration of TDFS's locations, a VLRD is used for storing a TDFS location when the TDFS moves from its home location in the original cloud. Because a TDFS can be moved anywhere in clouds of different infrastructures, mobility management is essential. When a TDFS moves, location registration is always needed for tracking and tracing purposes. Therefore, if visited locations of TDFS are stored and managed as a distributed database system, the MS at visited cloud queries directly the VLRD rather than the LRD at the original cloud. A record in the VLRD is composed of seven fields including TDFS_ID, Supervisor_ID, Operation, Source Address, Current Address, Time Stamp, Data Owner. VLRD's data structure inherited from LRD with an additional field as Current address. Current address A set of *CIP* = $\{cip_1, cip_2, ..., cip_i, .., cip_n\}$ and VLRD table set $L = \{tn_i, sp_i, ip_i, cip_i, op_i, ts_i, ow_i\}$.

Table 4.3 VLRD's data table structure

TDFS_ID	Supervisor_ID	Operation	Source Address	Current Address	Time Stamp	Data Owner
tn_1	sp_1	op_1	ip_1	cip_1	ts_1	ow_1
tn_2	sp_2	op_2	ip_2	cip_2	ts_2	ow_2
…	…	…	…	…	…	…
tn_i	sp_i	op_i	ip_i	cip_i	ts_i	ow_i
…	…	…	…	…	…	…
tn_n	sp_n	op_n	ip_n	cip_n	ts_n	ow_n

4.5.3 Data mobility management workflows

When a customer subscribes to a cloud service, the CSP will assign roles associated with the data for users, allowing them to access a virtual user directory and workspace. An initial set of empty active data cubes will be created according to the regular data types. After assigning roles, the supervisor will be invoked to send a request to the MS for data location registration. The request containing parameters such as UserID, DataID, Location and Time Stamp will be processed to update the database. Finally, the user will receive a data location registration acknowledgment message via the MS. Figure 4.10 shows the workflow for a new data location registration.

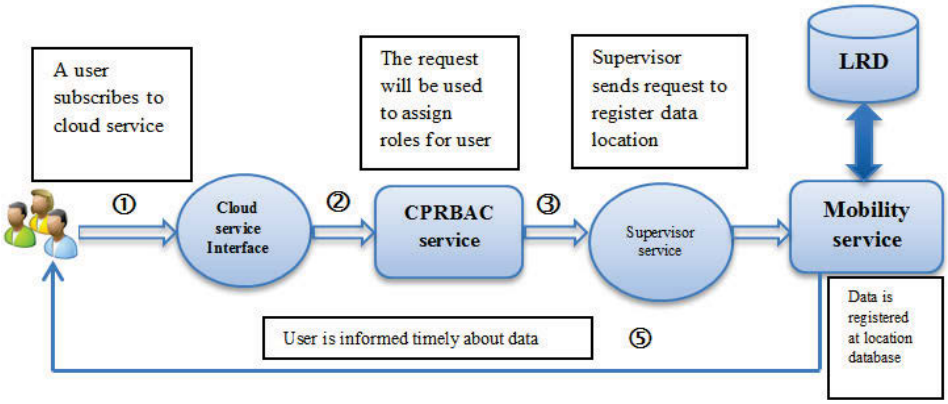


Figure 4.10 New data location registration workflow

When a user needs to execute data operations such as read, insert, write and move the data, he/she will send a request to the cloud interface including a set of parameters such as UserID, DataId, Operation and Location and Time Stamp. In turn, a verification process is created to perform a sequence of steps. Firstly, it invokes the supervisor to access the data. Hence, the supervisor needs to establish the validity of the request by forwarding it to the original cloud where data location is also updated. If the request is not valid or not allowed by the access policy, the supervisor will raise the alarm to notify the system administrator or related legislation organization. In fact, if desired, the data owner may be informed immediately when the original cloud detects the violation through the MS. If the request is permitted, the MS will update data location before approved verification is sent to the supervisor. From this point, data operations will be performed on the TDFS but each operation is recorded inside the TDFS for tracing purposes. Figure 4.11 presents the general procedure for data mobility workflow.

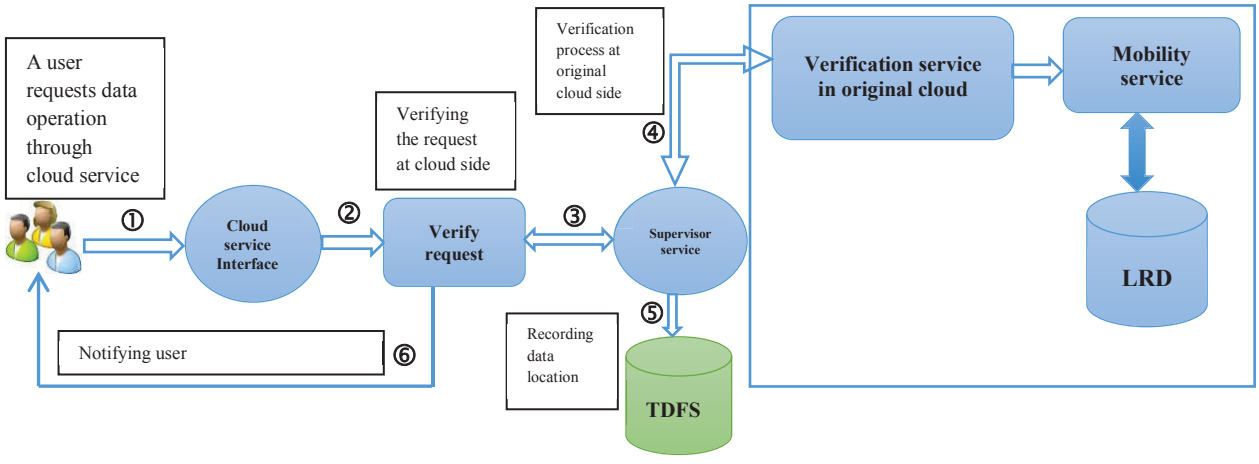


Figure 4.11 Data mobility workflow

When the mobility service is triggered, a request is sent to LRD in order to query data location

for data validation or to update new data location. Further, the MS also informs the data owner with a message concerning the accessed operations via email or mobile devices. Details of the workflow are shown in Figure 4.12.

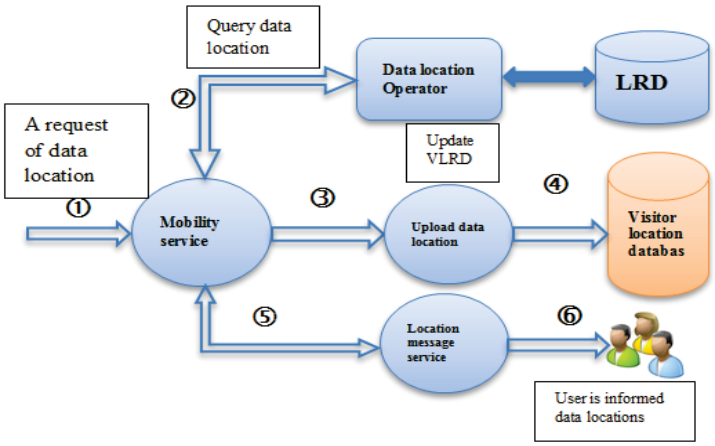


Figure 4.12 MS workflow

4.6 Evaluation and results

4.6.1 Experiment setup

We performed our experiments deploying Amazon EC, Azure and NeCTAR clouds. Our experimental environment has been set up as follows: Amazon EC2 (EC2, 2014) was used as the original cloud to provide cloud resources and verify requests. We created one t2.mocro instance of Windows 2012 server running on Intel Xeron E5-2670 v2 2.5GHz 2.5GHz with 1GB memory and another instance running MySQL to store the LRD with 5 GB storage. Packages of the model were deployed on this instance and requests were accessed via the RMI interface. We also created an Azure cloud (Azure, 2014) and a NeCTAR cloud ((NeCTAR)) as the visited clouds. We used an instance running Windows Server 2012 Datacenter with Intel Xeon E5-2660 2.2GH 2.19GHz and 7GB at Azure and another instance running Ubuntu 14.04 within 1 VCPU, 4GB Ram at NeCTAR to send requests via the RMI interface. In order to demonstrate the working of the mobility model, a message application for notification at users' mobile phone was implemented in Java on an Android 4.4 smartphone with Quad-Core 3GB 2.7 GHz and 3GB RAM. Google Cloud Message (GCM) (GCM, 2014) was devised to send messages informing users when the data was accessed or moved at cloud side. A notification could be triggered via two sources depending on the request of data operations. If the request

was to move or to copy data, the MS would inform the data owner via GCM while operations such as read or write would be triggered by the probe inside the data.

The current experiments were based on the following assumptions: we assumed that the runtime environment (Java Virtual Machine) of the active data behaved correctly at participant clouds, and we assumed that the data would be activated when it moves to a new cloud host. We also assumed that a safe Location Service Provider (LSP) was available on the Internet and CSPs had the agreement to establish new supervisors. At this stage, we tested our model on cases where TDFS and regular files were moved between two clouds. Next, different clouds (with different infrastructures) involved in data moving cases were also used to demonstrate that the model is trustworthy and behaves correctly. To observe the performance of the data mobility management model, we first compared data operation overheads in moving regular passive data and in moving our active data. We also compared the overheads of two types of request: one came from inside the cloud and the other from outside, to evaluate the feasibility and proactivity of the system. Finally, we demonstrated security and transparency of the proposed model.

4.6.2 Evaluation

In our previous ADC framework, a data moving operation was simply performed between two hosts in the private cloud environment. One host creates a request to move the data while another host provides various data access scenarios to ADCu in the same cloud. In this section, we deploy various data moving cases among clouds to address not only data mobility issues but also privacy and security. Requests will be created from both inside and outside the original cloud. Through authorization and authentication schemes based on the CPRBAC service and the LRD in the original cloud, any data moving operations will be verified regarding user's permissions or data locations. The implementation of TDFS is based on our previous work (Chen and Hoang, 2012) and new fields are added to support data mobility.

We assume that data owners do not release sensitive information to unauthorized parties including secret keys which could be used to generate signature and encrypt data and personal privacy. We assume that the LRD and CPRBAC services are trustworthy and behave correctly.

4.6.2.1 Processing data moving case tests

a. Data moving test for ADCu and normal files

We investigate a new data mobility model for ADCu by performing tests to identify the cost of moving data such as time delay on TDFS files compared to normal data. Normal files are the raw files in formats such as PDF file, XML file or DOC file. The idea is to determine the overhead introduced by the security features and the MS performed on an ADCu of our model. We increased the file size from *300KB* to *1000KB* for both TDFS and normal files and compared the costs of moving operations within our model. All results are an average of five trials. The executed time $t_{Request}$ for each request is composed of three determinants: the verification time $t_{Lookup\ service}$, the location register time $t_{Verification\ and\ mobility}$ and the data operation time $t_{Data\ operation}$.

$$t_{Request} = t_{Lookup\ service} + t_{Verification\ and\ mobility} + t_{Data\ operation} \qquad (4.1)$$

1) $t_{Lookup\ service}$: the time that client spent looking up server's RMI interface and sending the request.

2) $t_{Verification\ and\ mobility}$: the security and MS latency. The CPRBAC and Location register will be processed during this period.

3) $t_{Data\ operation}$: the data transfer time between the original cloud and the visited cloud.

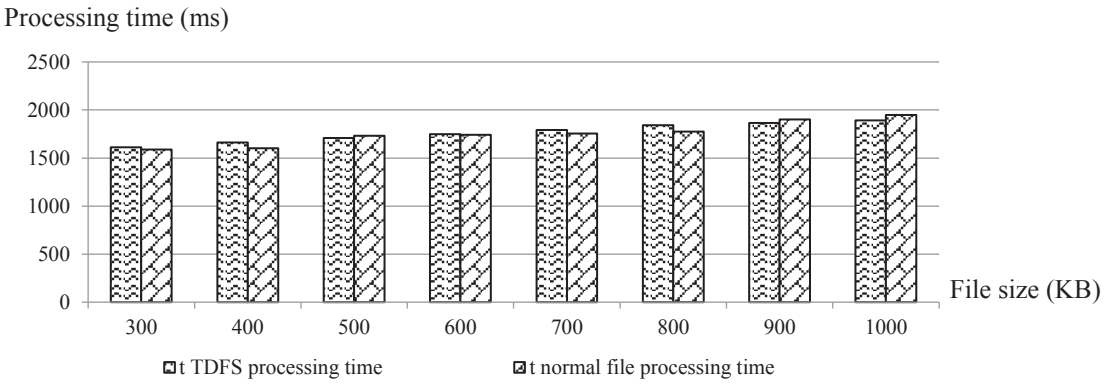


Figure 4.13 Request Processing Duration

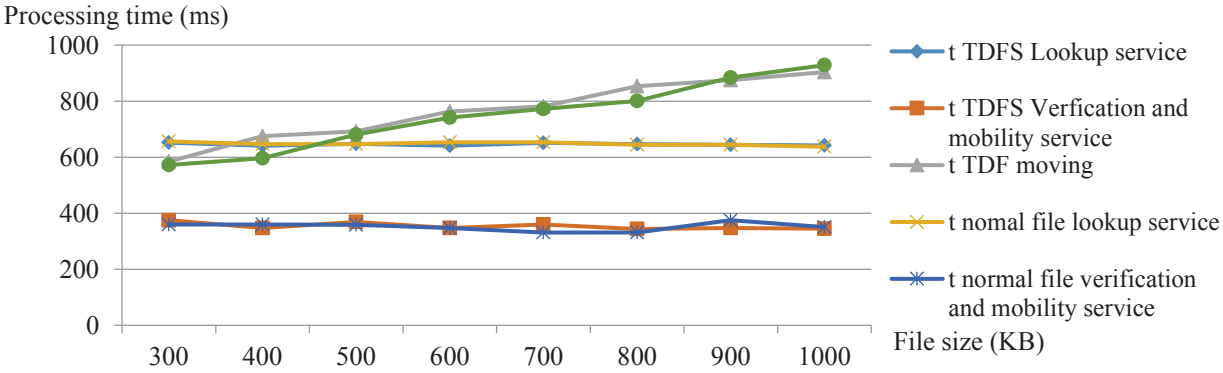


Figure 4.14 Modules Processing Duration

From the results, it is clear that the processing time for a TDFS is slightly longer than that for a normal file (*1765.25ms* in comparison with *1755.4ms* of the later). The main components of the whole moving process are the lookup service time and the moving data time. The verification and MS time, the main process of the model, however, only constitutes a small amount of time (*353.93* of *1765.25ms* for TDFS and *351.98* of *1755.4ms* for normal file respectively). The comparative results are illustrated graphically in Figure 4.13 and Figure 4.14. The x-axis represents the file size, and the y-axis represents the execution time.

From our experiments, we found that the verification time and the lookup service of the model are approximately the same for both TDFS and normal files. Therefore, the source of extra delay must be introduced by the transfer time. Hence, we run the same request with different data sizes. The results show that the transfer data time is indeed the significant source of latency.

This means the data protection and MS did not introduce significant overheads when the data size was increased. Overall, the amount of overheads is considered as a small price to pay for security and data protection. With our proposed model, the supervisor can trigger TDFS into an active state for self-protection; self-defense and record accessed addresses, whereas these operations cannot be performed on normal files.

Figure 4.15 shows alerting messages for move requests as received by the user via the mobile phone in the move process. The message includes essential information about the moving process such as file name, moving addresses, time. It is demonstrated that the supervisor can detect data move operation and invoke the MS to send the alerting message to user's mobile device immediately when there is a request to move the data at both the original cloud and visited cloud.

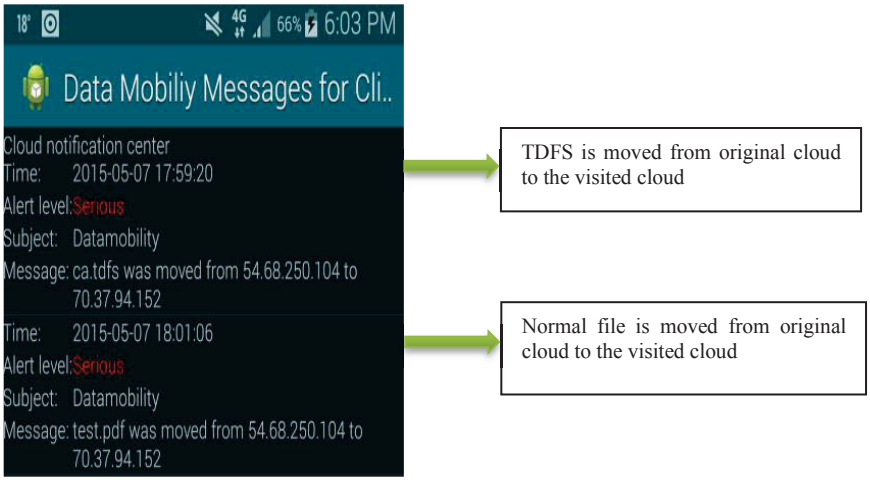


Figure 4.15 Data notification view in Samsung Galaxy Note 4

b. Data moving cases test among different clouds

In the following experiments, we deploy the model for different data moving cases for ADCu described in Section 4.1. Requests were created from both inside and outside the cloud for different data moving cases. Our primary goal is to demonstrate the efficiency, security and transparency of the proposed model. In addition, we recorded and compared the verification and MS service duration of requests from inside and outside the cloud to identify the source of latency. The results of outside requests within different data moving cases are compared in Figure 4.16 and Figure 4.17.

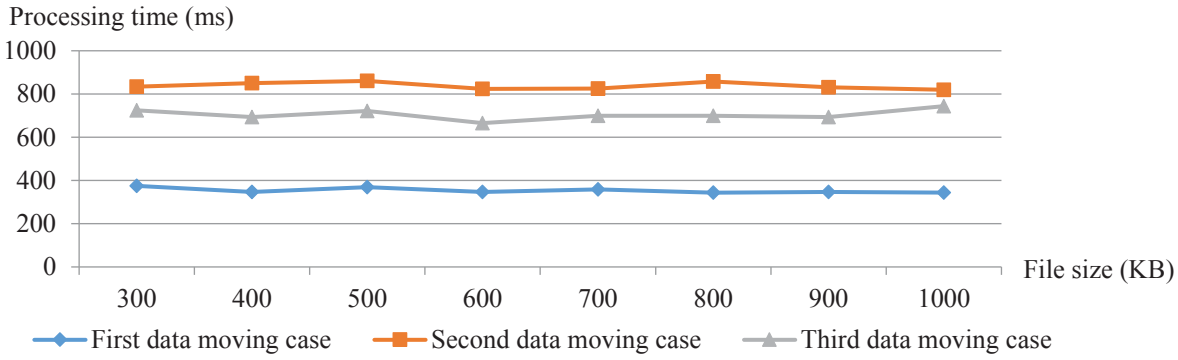


Figure 4.16 Verification and MS duration of different data moving cases

As can be seen from Figure 4.16, the verification and MS costs of the second case and the third case are substantially higher in comparison with that of the first case (the average times are *837.6 ms*, *705.5 ms* and *353.9 ms* respectively) while there is only a small difference in the costs of the last two cases. This can be explained by the fact that for the last two cases the move process requires two stages: sending requests to the old cloud and relaying these requests to the original cloud. All requests have to be forwarded to the original cloud which is responsible for verifying the requests and executing the MS even when requests are created from original cloud (the third data moving case). We also recorded the response time of verification and MS by requests from inside cloud to compare with that of requests from another cloud. The requests from inside the cloud can be created from two sources. One is from inside the original cloud to move the data to a new cloud and the other is from a visited cloud to move the data to a new cloud or back to the original cloud. The transferring data costs are similar for both two kinds of request within moving cases on the same data. In fact, the verification and MS cost is also the same for an inside request at visited cloud for the second and third data moving case. Thus, we only record the average response time for these two kinds of request *(251.4ms* and *612.6ms*

respectively). Similarly, the average response times of the second and third data moving case are about double that of the first data moving case due to the relay process.

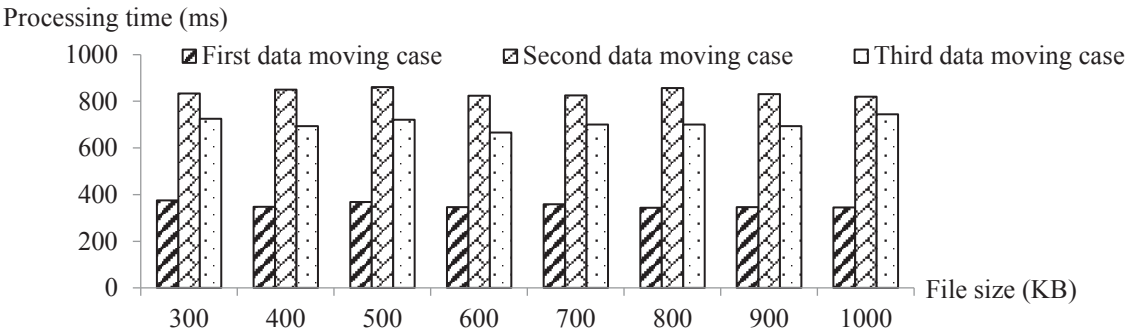


Figure 4.17 Processing duration of different data moving cases

The overhead for data retrieval from a TDFS involves verification and identification, data loading, and network communication cost. To evaluate the costs of the verification and MS we run a number of tests. *50* working threads are allocated to generate user requests in parallel. Hence, we initialized six sets of tests in which the number of requests ranged from *50* to *500*, and each set of tests runs *5 times* circularly. Figure 4.18 illustrates the time cost of executing Time cost of verification and MS for different data moving cases and requests. The x-axis represents the number of requests, and the y-axis represents the execution time. As we can observe, along with the increasing of the number of requests, the time cost of verification and MS increases predictably. As we observe, the average time of inside requests for the first data moving case (*47920.1ms*) is similar to that for the outside requests (*48074.55ms*). This is repeated similarly for the second data moving case and the third data moving case. However, the average time of requests for the second data moving case and the third data moving case (*115285.8ms*) is approximately twice than that for the first data moving case (*47997.34ms*). It can be explained that the requests have to be forwarded back to the original cloud for executing verification and MS. Generally, the time cost grows linearly with the increase of traffic requests. This indicates that the data mobility management model can efficiently avoid an exponential increase in response time when handling multiple requests. Replicas of the services to geographically separate hosts may be used to achieve load balancing and improve performance.

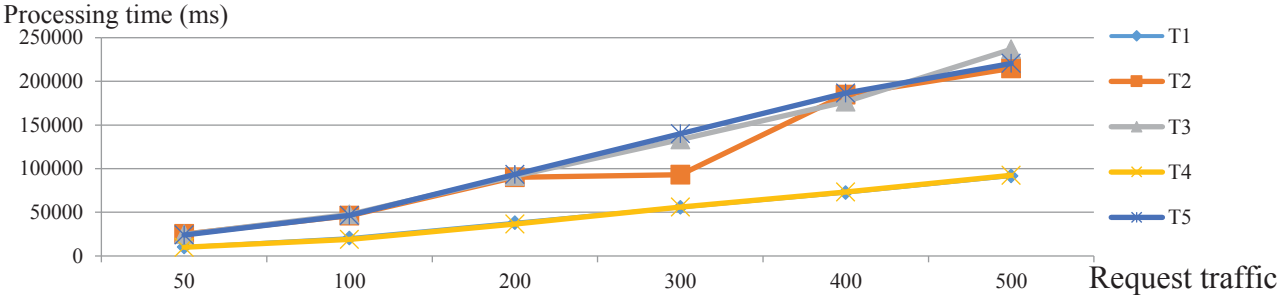


Figure 4.18 Time cost of verification and MS for different data moving cases and requests

Figure 4.19 presented the data moving operations for different data moving cases. All data moving cases will be triggered by the supervisor. The first one is triggering the MS and TDFS when data is moved from the original cloud to the visited cloud; the second one is triggering the MS and TDFS when the data at visited cloud is moved to a new cloud; and the third one is triggering the MS and TDFS when the data at visited cloud is moved back to the original cloud; and the last one is triggering the MS and TDFS when data is moved out from the cloud. In all these test cases, the supervisor was able detect and trigger the MS and the TDFS when data was moved. Meanwhile, our mobile device received the alert message immediately. A message including source address and destination address actively notified the data owner when data was moved to a new cloud.

Performance analysis: Our model does not rely on complex algorithm and encryption requirements. The performance of our model in terms of the computation overhead is as follows:

Verification service: the available computation overhead on these two processes includes access request generation, policy matching, verification between access request and targeted policy, calculating verification token, and access response generation. Among these operations, the policy matching complexity is O(N); verification complexity relies on the number of context variables required to analyze; other operations complexity are O(1).

Mobility service: The MS creates requests to query the database, which only adds AQ6 negligible overhead to database as it does not incur initialization cost and database join algorithms. It only takes charge of triggering tasks, the other fetching data locations will be handled by the database instance.

4.6.2.2 Attack test and security analysis

In this section, we analyze possible attacks on our model. As the peripheral environment protector, the access control layer (ACL) in the entire framework resembles the firewall in a network security system that manages the incoming and outgoing network traffic to secure the internal network or computers. It blocks and defends external attacks caused by adversaries. Through authorization and authentication scheme based on the CPRBAC service, unauthorized requests would be rejected from accessing the active data stored in the cloud data storage layer. However, adversaries may leverage the elevation of privilege or illegal channel to gain higher

administration privilege (it can bypass the ACL) to directly commit an inner attack or penetrate data storage layer.

a. Direct access and Intrusion attacks

The most direct attack is when an adversary tries to access the TDFS's content or move it to unauthorised locations. The verification process in our model operates as follows. Any request to access a TDFS has to be verified at the CPRBAC for access permissions and the LRD for data's identification. Only the entity, whose request passed the verification, may continue to perform operations on the TDFS. If the TDFS's content is accessed, the shell within the TDFS will trigger its shell protection scheme to execute verification and identification. A request without any parameter is immediately regarded as an intrusion attack. Even if the adversary could input correct parameters within the request's structure, the verification process is executed to ensure the consistency of both CPRBAC and LRD. In fact, the data owner may also be informed with a notification message.

b. TDFS external attacks

If an attacker bypasses the cloud firewall and security system, and obtains a root privilege, he or she is able to execute data operations at the OS levels on the TDFS, such as copy, move, and delete the entire TDFS from cloud. These operations would not trigger the TDFS files into an active state without a supervisor's instructions, and hence the adversary is able to expose sensitive information inside the TDFS. Data operations on TDFS are fully monitored based on the establishment of a clone supervisor in the same domain. Once the adversary performs these operations, the supervisor can detect them, and then trigger the TDFS into an active state as well as notify the data owner. In the active state, the TDFS can execute the self-protection process. For data owners, if their data is compromised or violated against the SLA of the subscribed cloud service, the MS can inform them through a mobile device. Furthermore, any illegal violations could be delivered to relevant governance, regulation, and compliance authorities. The screen shot in Figure 4.19 shows notification messages which are triggered by the supervisor.

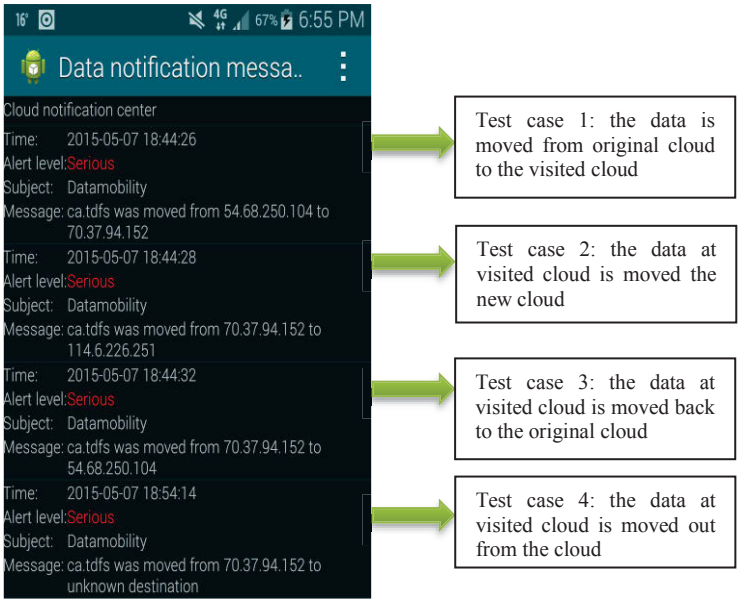


Figure 4.19 Data moving cases notification view in Samsung Galaxy Note 4

4.7 Discussion

The last section focuses on our data mobility management model and the trust-oriented data protection framework. This section discusses several assumptions and measures that can be improved and extended in the future work.

When an ADCu moves from its original cloud to another cloud, the responsibility to protect the data depends on the SLAs between the data owner and its original cloud provider as well as between the original cloud provider and the visited cloud provider. The mechanisms to deal with security and protection can be complex and vary depending on the agreements and assumptions. If we assume that the original cloud is mainly responsible for its registered data then the tasks of authentication and authorization at a foreign cloud have to be done at the original cloud. This implies that the clone supervisor established at the visited cloud has little responsibility as it will pass any requests to do with its data to the original cloud to deal with. However, if we assume that the new cloud will be mainly responsible for its visited data, the supervisor has to be generated with adequate capabilities. In this work, we assume a light-weight supervisor as it passes requests to the original cloud.

The assumptions also affect the performance of the moving process. As in our cases, requests to move data always have to be relayed back to the original cloud and hence the MS processing time and the verification time can be doubled or tripled depending on whether the

move involves two or three clouds. If the authentication and authorization processes are delegated to the visited cloud, these processing times can be reduced.

The way the active ADCu communicates with other entities can be selectively designed. The ACDu when moved to a visited cloud may or may not be able to communicate with its owner. Even if it may, the owner may not wish to be disturbed unnecessarily and some acceptable communications mechanisms should be developed. It is important that the ACDu must be able to protect itself and to communicate with some reliable external entity if the protection model is to be credible.

Clearly, those assumptions can be relaxed and the basic model can be extended to deal with various situations and provide more comprehensive data mobility and protection management.

4.8 Summary

This chapter discusses data protection and mobility management in cloud environments. It also presents an active framework for data protection and an extended trust-oriented framework for data mobility and protection for handling secure data mobility in a cloud environment that involves data moving within and among the original cloud and visited clouds. It also proposed a novel LRD that is capable to serve for tracing and tracking data locations. Furthermore, a new TDFS structure with recordable structure was designed to actively capture locations of requests. More importantly, a proposed establishing supervisor at visited cloud is able to deploy the equivalent data protection scheme at both cloud side. The experimental outcomes demonstrate feasibility, efficiency of the model. Further, the reliability of the system is guaranteed in terms of processing time. With the growth of data in the cloud, data distribution has become an issue for data management in the cloud environment. In the next chapter, we will present our proposed mechanisms to address issues in distributing data efficiently.

CHAPTER 5 A DATA DISTRIBUTION SCHEME AND DATA REPLICATION SCHEME FOR LARGE-SCALE OF EHRS

Chapter 5 presents the data distribution model and data replication management that mainly focus on distributing, retrieving EHRs and maintaining the workload balance efficiently. It utilizes hashing to distribute files into clusters in a controlled way and it utilizes P2P structures for data management.

This chapter focuses on the data distribution and data replication mechanism. The former relies on the HBFC scheme that distribute, store and retrieve EHR efficiently in cloud environments. The HBFC scheme first reduces the searching space and then leverages the P2P searching mechanism to achieve data lookup efficiency. It utilizes the "collision" property of the hash function in a controlled way to gather data into virtual clusters and leverages P2P to search files within a cluster. We are not aware of any existing methods that use "collision" to control the formation of clusters as designed in our scheme. The data replication scheme utilizes the HBFC scheme to distribute new replications in a balanced manner. The proposed scheme can achieve better workload balance and storage efficiency than random replication (refers to replicate data replication randomly at nodes) and status replication (refers to the fixed and static replicate data replication at nodes) which might result in hotspots.

In accordance with the key issues discussed in Section 1.2, and the desirable properties and the objectives of the data distribution model described in Section 1.4, we propose a novel HBFC scheme and data replication scheme for clustering, searching and replicating files. In this chapter, Section 5.1 discussed file distribution efficiencies over different hash functions. It analyzes different hash functions in terms of the balance distribution. Section 5.2 briefly discusses the combination of clustering and searching files. Section 5.3 introduces the HFBC scheme. Section 5.4 presents the data replication scheme. These schemes form the distribution

and replication mechanisms. Section 5.5 presents the simulation results and evaluations of our proposed schemes. The conclusion is drawn in Section 5.6.

5.1 File Distribution efficiencies over different hash functions

Hashing is known as the most efficient mechanism to distribute data within a large-scale systems. However, the use of collision of hash function has not been investigated. In this section, we present our investigations on the efficient distribution of different hash functions.

Hash functions provide the mapping from objects to systems. Objects are identified by keys which are used as inputs for hash functions. The hash functions calculate keys and provide hash values. These hash values are then represented in a ring called a hash space. This space can be used to assign to available nodes in the systems. Hash functions have low complexity in searching objects, especially as they require only O(1) computation time. Hence, it is a good option to distribute and retrieve a large amount of data in the systems. Since hashing is responsible for the distribution of the workload across systems using hash functions, selecting an appropriate hash function results in efficiency in data management in term of distributing and retrieving. We first carry out experiments on different hash functions to identify feasible ones. Next, we apply the selected hash functions in our proposed data distribution scheme.

The first critical issue for looking up space problem is to identify the cut-off point between small clusters and defined clusters to answer the question "what is the best size of the defined cluster". In this section, the issue is studied through experiment analysis. Thus, this enables us to define distribution rates. The relationship among the number of files, the size of the hash table and the number of collisions in distributing files is analyzed through experiments. Based on the relationship, the distribution of files into cubes and nodes is quantified.

5.1.1 Experiment method

We use different hash functions for simulations. Each hash function is run with different hash table sizes and number of files. The sizes of hash tables are *160 bits*, *128 bits*, *64 bits*, *32 bits*, *28 bits* and *24 bits*, respectively. The combinations of different hash functions are also performed for two hashing rounds. The number of files is generated randomly from *1000000 files* to *50000000 files*. The following experiments are carried out on two key spaces:

- Hash data keys into the first key space (the small key space) to reach hits (ID1).

- Hash the current ID (ID1) into the second key space (the large key space) to create a mapping between these two key spaces.

5.1.2 Experimental results

File Distribution efficiencies over different hash functions are represented via the number of files per hit, the number of files at hits, and the number of hits. The simulation results of different hash functions are shown in the below figures.

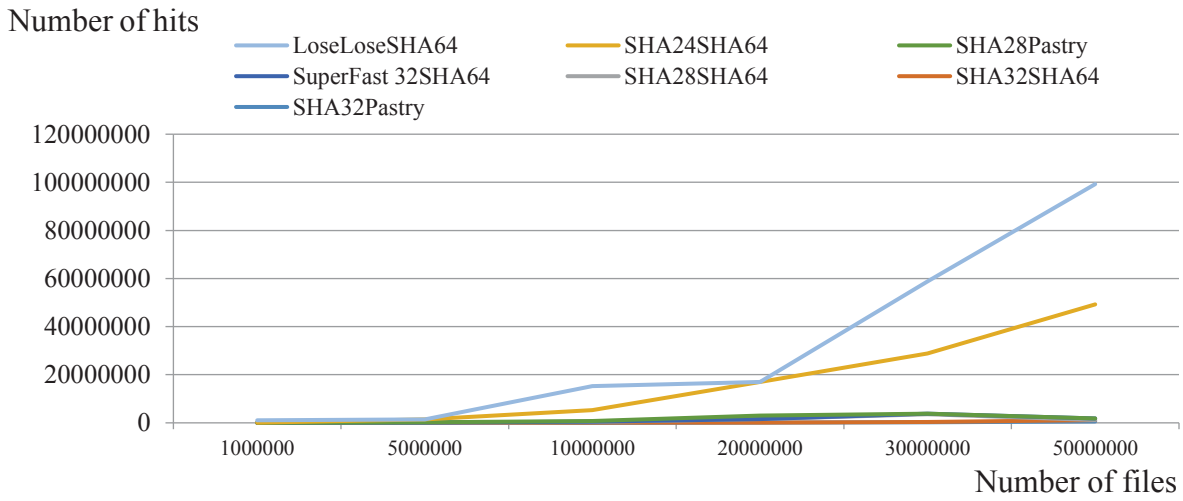


Figure 5.1 The number of hits

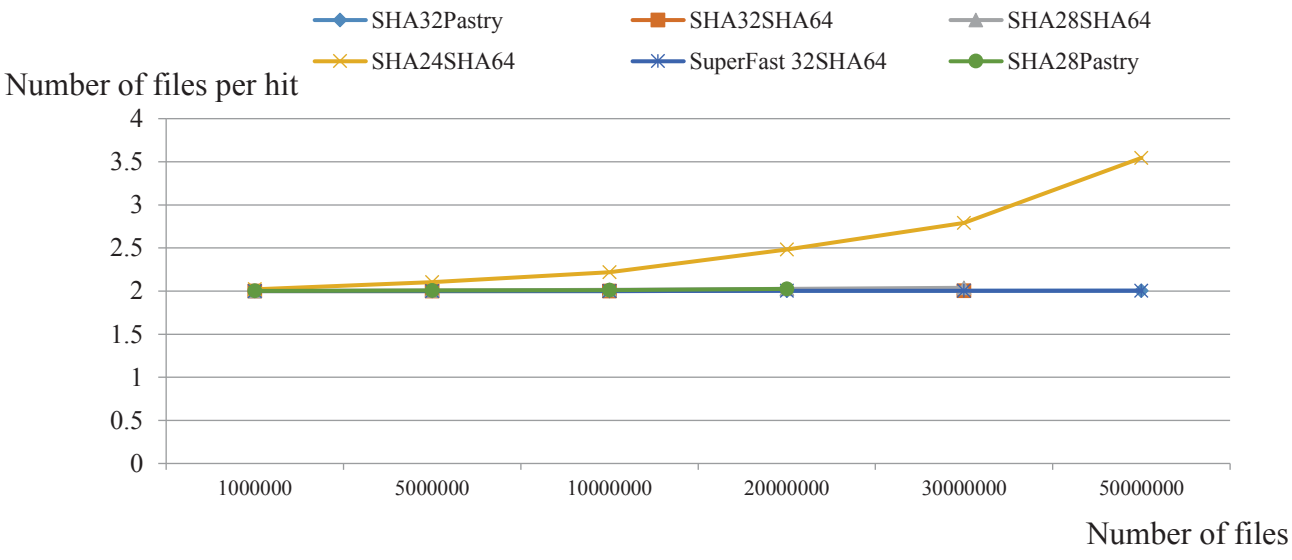


Figure 5.2 The number of files per hit

As can be seen in Figure 5.1, the number of hits is not too large. Thus, it is reasonable to deploy in the real world. However, the number of files per hit on hash functions such as SHA,

Superfast, and Pastry is quite small (Figure 5.2). Hence, it is not efficient to design the file distribution mechanism using these hash functions It is expected that the number of files per hit should be large enough, say around 10^3 files per hit, such that files can be searched in defined spaces efficiently.

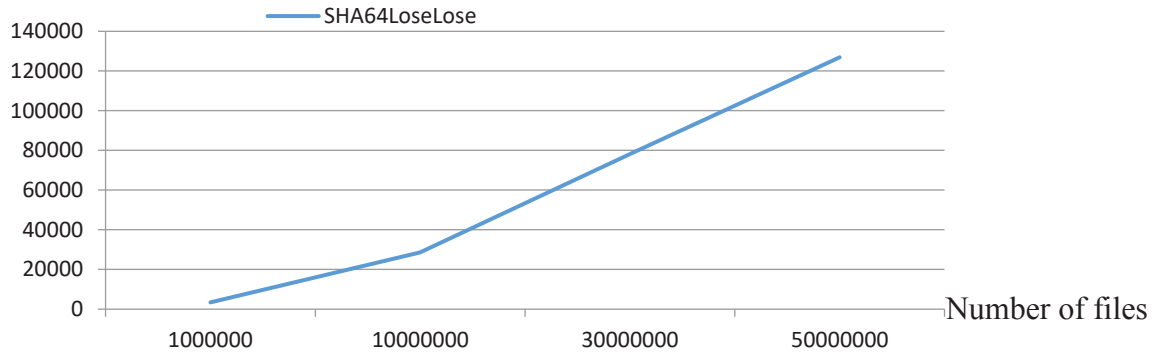


Figure 5.3 The number of files per hit of SHA and LoseLose hash functions

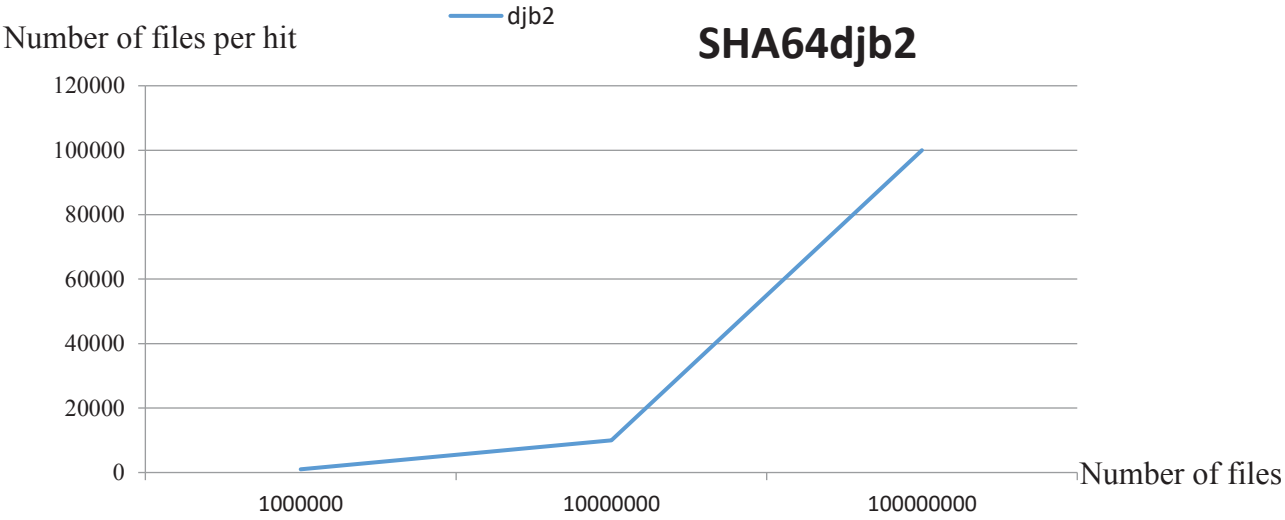


Figure 5.4 The number of files per hit of SHA and DJB2 hash functions

For the SHA64LoseLose function, the number of hits is increased when the number of files per hit is increased (Figure 5.3). In fact, this function produces the smallest number of hits in comparison with that of other functions. Figure 5.4 presents the experiment of SHA64djb2. In this function, we adjust the size of the hash table to obtain the fixed value of a number of files per hit (10^3) by increasing the rate between the number of files and the size of the hash table. As can be seen in the figure, we can adjust the size of the hash table to obtain the expected the number of hits that means most of the files in the system stored in defined nodes within the HFBC. Therefore, we choose djb2 as a hash function for data distribution in our design.

5.2 The combination of clustering and searching files

In overlay sharing files systems, the data lookup process is the most computationally expensive task while in file distribution systems, if the locations of data clusters are well designed, then the expensive task is the data retrieval process. Good searching algorithms enable speeding up of the data lookup process. Hashing is known as the fastest searching algorithm since its complexity is O(1). P2P systems utilize the hash functions to calculate data keys in order to reach peers storing requested files with low latency. To benefit from hash function features, we utilize it in our proposed schemes in both clustering and searching files. On the one hand, considering that it costs only O(1) computation time, the searching performance of the system is not affected. On the other hand, storing files at the clusters in a controlled manner allows us to find requested files in a shorter time frame. Next, we will detail how to control the distribution of files into clusters.

In order to control clustering files, we treat the system's configuration as an adjustable parameter so that scaling up or down can easily be achieved when the system's configuration changes. Parameters defined are the number of files in the system, the expected average number of files stored in one cluster, and the number of nodes in the system. The idea of combining clustering files and searching files stems from two considerations: 1) if we can store most files of the system in defined clusters, the searching process is mainly performed at these clusters; 2) distributing a certain number of files in a controlled number of clusters enables retrieving data more efficiently compared to retrieving it among a large number of files in one cluster. Based on these, we adjust the parameters to control the hashing of files into clusters to achieve the expected file distribution.

Suppose that there are n_f files. The number of real nodes that form the P2P network is denoted by n_n. The number of defined cluster nodes is denoted by n_c. The number of files per node is denoted by f_c. The number of nodes which are not in defined cluster nodes is denoted by $\overline{n_c}$. The average number of files in $\overline{n_c}$ nodes is denoted by $\overline{f_c}$.

The expected number of nodes storing most of the files in the system is derived as

$$n_c = \alpha * n_n \tag{5.1}$$

where $0 < \alpha \leq 1$.

$$\overline{n_c} = n_n - n_c \tag{5.2}$$

The expected number of files distributed in n_c nodes is derived as

$$n_c * f_c = \beta * n_f \quad (5.3)$$

where $0 < \beta \leq 1$

As a result, the computation formula for f_c is derived as

$$f_c = \beta * \left({}^{n_f}/_{n_c}\right) \quad (5.4)$$

$$\overline{f_c} = \left(\frac{n_f - (\beta * n_f)}{\overline{n_c}}\right) \quad (5.5)$$

As can be seen from the formulas (5.1) and (5.3), we are able to control file distribution proportions given n_f and n_n as input parameters of the system. By changing the α and β parameters, we can control the percentage of clusters and the percentage of files distributed within these clusters. The first ring produces expected distribution rates as follows.

1. $n_c = \alpha * n_n$ nodes that store most files of the systems. As a result, the retrieval process is focused on these nodes resulting in efficiency in lookup latency and access latency.

2. $n_c * f_c$ files are distributed across n_c nodes and f_c files per cluster results in a fast look up process since we reduce the lookup space to f_c

3. $\overline{f_c} * \overline{n_c}$ files are left over in a small number of clusters. $\overline{n_c}$ is designed to be much smaller than n_c.

5.3 HBFC scheme

In the HBFC scheme, we use two different ID spaces to store the hash values of each hashing round (Figure 5.5). The first ID space has a small ID range representing file distributions on cluster nodes. Different files may be mapped to the same ID on the first ring as the result of hash collisions. We call a "collision" a "hit" and the number of hits per cluster can be controlled. We call this ring a virtual ID space; an ID on this ring represents a virtual cluster as it only contains a cluster of IDs, not a real cluster that contains files. This ID will be hashed into an ID of a real cluster in the second ring where files are found. The second ID space is designed with a large ID range as is often the case in many existing P2P systems. Large P2P's ID space and proper hash functions allow keys to be hashed into unique IDs in a uniform distribution. An additional feature of our scheme is that only the cluster ID is used in the second hash round and the (file) data key is preserved so that searching for the data file within a file cluster can be

performed. Figure 5.5 depicts the design of HBFC scheme. As can be seen from the figure, there are two rings used to store hash values for each round. The first ring is designed as a virtual ring to gather hits from hash function (*h1*) when hashing different data keys to *s_id* in the first ID space. The aim is to control the expected number of files distributed into the same cluster. The second ring follows the original P2P's ring which hashes the *s_id* to *l_id* in the second ID space. This is based on the feature of one-way hashing that is the hash function only produces the same hash value for each data key. Therefore, if data keys are hashed with a value *s_id*, then this ID is definably hashed into the corresponding *l_id.* In order to obtain this design, we also modify the second hash function and attach the data keys for the second hashing round. By doing this, the requested files can be found at the destination node. The details of this design are presented in following subsections.

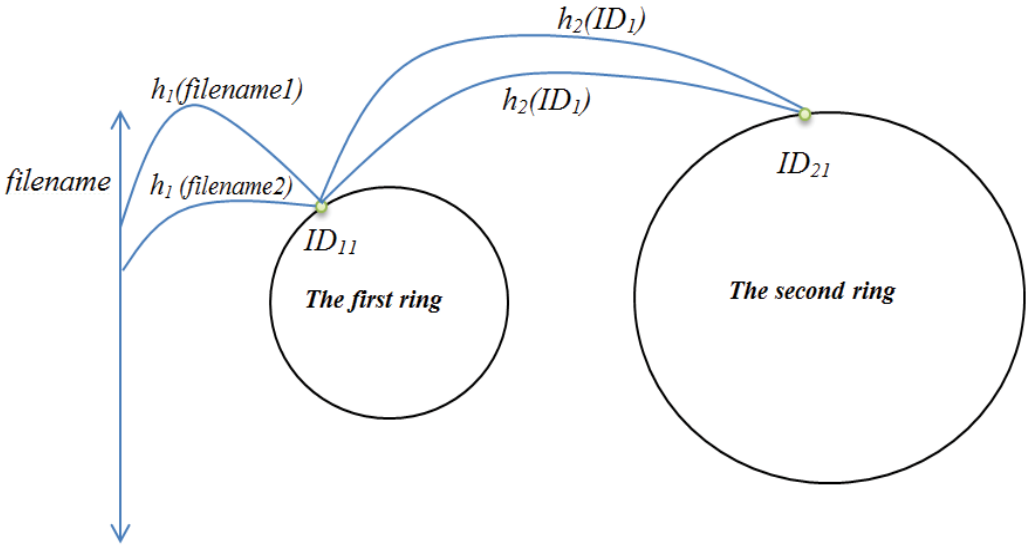


Figure 5.5 The design of HBFC scheme

A data item such as a file or a group of files is represented by a (key, value) pair. A key is a label or name of the data, such as a file name, while a value represents the content of the file. The pair (key, value) is inserted into the system by peers and the key is used for the lookup process in order to retrieve the corresponding value. To start searching a file, the peer first hashes the key to *s_id* which is a defined hit in the first ID space. Following that, *s_id* is used as the data key for the second hash function. It is noted that we attached the data key into *s_id* to keep the identity of the data. In the second hashing round, the *s_id* is hashed to *l_id* which is a unique ID in the second ID space. In other words, hashing different data keys may produce the same ID; this means that these files are stored in the same cluster. The next subsections will detail the design of each ring.

5.3.1 The first ring

The first ring is a virtual ring providing a small ID space for the hash function in order to obtain the balance distribution when many data keys are hashed to one ID. Figure 5.6 depicts the design of the first ring.

Let h_1 denotes the hash function of the first ring, the computation of h_1 is derived as follows.

$h_1(key) = ID1$

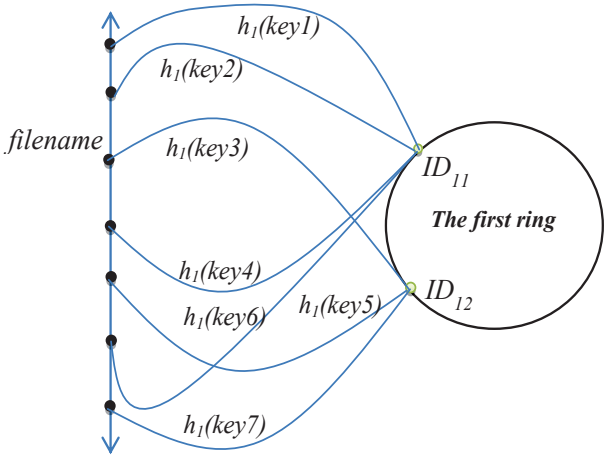


Figure 5.6 The design of the first ring

As can be seen from the figure, different data keys are hashed to the same ID. For example, *key1, key2, key4* and *key6* are hashed to ID_{11} while *key3, key5* and *key7* are hashed to ID_{22}. The hash values such as ID_{11}, ID_{22} then are used as data keys for the second hash function. As a result, we are able to achieve our design in distributions due to controlling a certain number of files stored in one cluster. Firstly, many data keys are hashed to the same ID and that implies that multiple files are stored in the same cluster. In fact, the number of data keys mapped to the same ID is controllable. However, there is an issue of losing the identity of the requested files when we hash the data keys with two hashing rounds. Since ID_{11} are ID_{22} are virtual data keys or temporal keys, they do not provide the identification of requested files which are searched on the real nodes. Their values are only used to reach the nodes which store requested files in the second ring representing real P2P nodes/systems. In order to mitigate this issue, we propose the attached keys within the hash function to search the files at real nodes in original P2P systems. Details of the attached key are presented in the next section.

5.3.2 The second ring

In the second hashing round, the second ring provides *a large ID space so that no collision of nodes is expected*. It follows exactly the original P2P hash functions. The mapping 1-1 is performed between the small ID spaces to the large ID spaces. In this stage, ID obtained from the first hashing round is used as the data keys for the second hash function (h_2). Figure 5.7 depicts the design of the second ring.

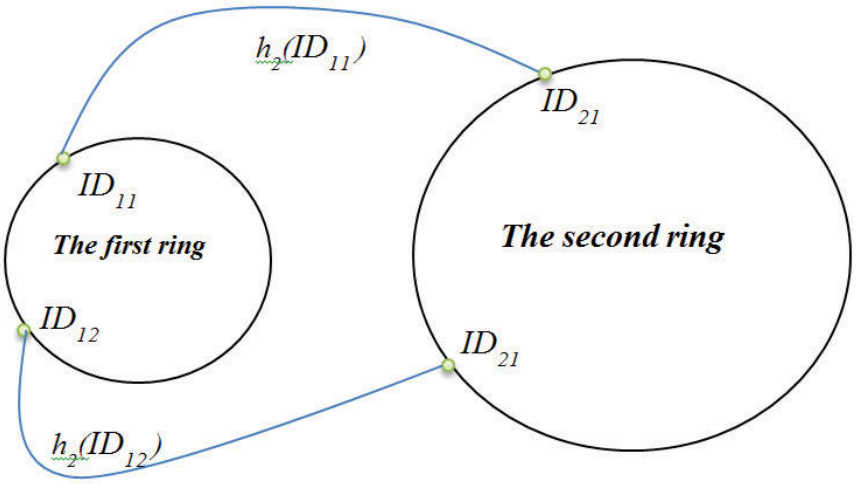


Figure 5.7. The mapping 1-1 between the first ID space and the second ID space

In P2P systems, a requested file is hashed into the ID space in order to reach the peer that stores the intended file. In this process, a real node may not exist at the hashed ID (Figure 5.8) and hence from this hashed ID, the lookup process is continuously performed in order to access closest real nodes. The range of lookup is determined by searching radius over the ring.

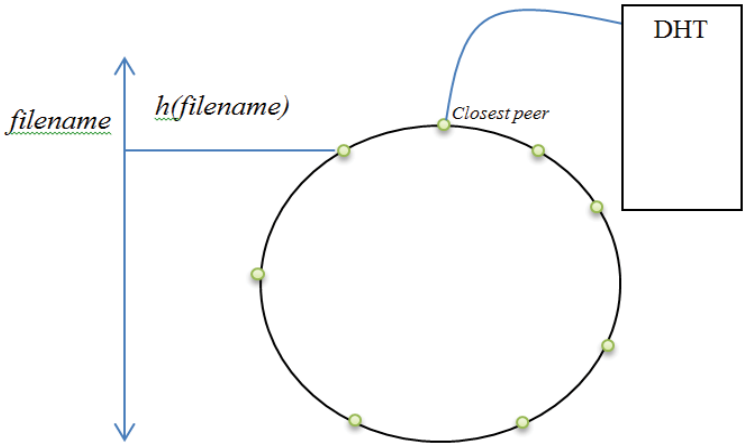


Figure 5.8 Find the closest node in the P2P system

Despite the fact that our scheme is able to gather many files into one virtual node, we still have to deal with the problem of finding the individual file in a real node. We identify the

lookup space of searching individual file in a real node to evaluate the HBFC scheme.

The lookup space of the closest node to the real node for an individual file may fall into the following scenarios: the real node may store a variable number of requested files as their IDs are closest to the real node ID; the node may store a random number of files and/or a variable number of small clusters not in the defined hits as the individual file IDs and the ID of these clusters are closest to the real node ID; the node may store individual files and/or files in small clusters and/or files in defined clusters. Generally, if the requested files fall within defined clusters, the HBFC scheme will return results more efficiently, due to the small lookup space. In other cases, the HBFC scheme costs approximately the same as the original P2P system since they execute the same lookup mechanism.

5.4 Data replication scheme

5.4.1 Data replication establishment

Our earlier work (Chen and Hoang, 2013a) proposed an adaptive data replication management scheme based on the active data-centric framework in cloud environments. Relying on the active data-centric framework, the scheme can adjust the data replication process based on the dynamic-window analysis (refers to the analysis of the collected average value and standard deviation in a dynamic set of the array) with respect to the response time of requesting data resources. Figure 5.9 depicts the data replication establishment

As each replica has different access frequencies, we define priorities based on the type of replications in the proposed scheme. The primary replicas C1 (high priority) are used to avoid a single point of failure. The secondary replicas C2 (medium priority) are used to improve access performance. The source replicas C3 (low priority) are normally used for archiving data. Each priority is assigned a positive representing integer number. For fault-tolerance, it is designed that each portion has at least three replicas: a source replica, a primary replica, and a secondary replica.

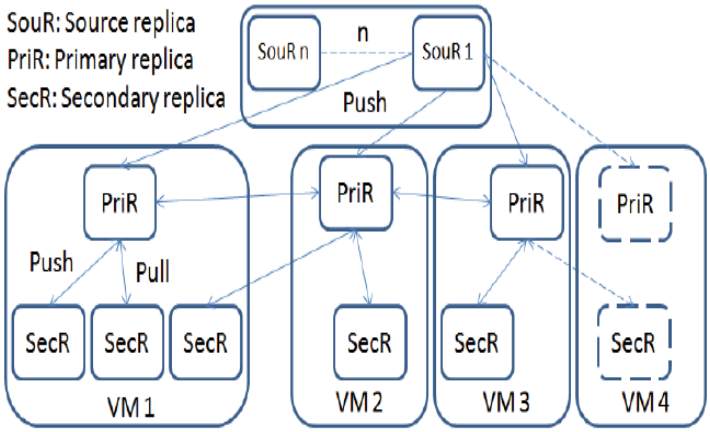


Figure 5.9 Data replication network topology

5.4.2 Data replication management based on HBFC scheme

As mentioned earlier, the first ring in the HBFC scheme is divided into hits and each hit is assigned to a virtual node at the P2P system. Hence, there is a mapping from virtual nodes to physical nodes. Figure 5.10 depicts the design of our proposed scheme.

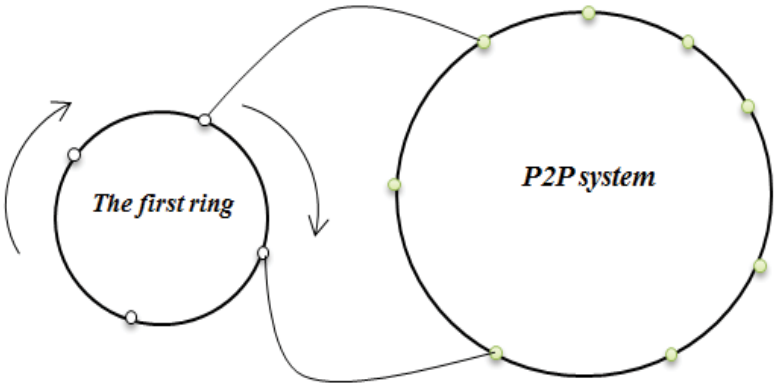


Figure 5.10 The design of Data replication management scheme

In this scheme, a request is first submitted to a peer in order to replicate the data. Then the peer needs to select a peer with the lightest load in the system to store a new replica with the highest priority. This means that a hit associated with the peer is assigned for the replica. The status of a hit such as the number of files at the same hit is used to rank the workload of a physical node. It is obvious homogeneous capacities and homogeneous access probabilities which sequentially store replication with high priority to a light load node is the most beneficial storage strategy.

In the early work (Chen and Hoang, 2013a), we measured the workload based on the response

cost, broadband usage, storage usage, CPU usage.

However, files are dynamically distributed among nodes of the system resulting in changes in the workload and spaces of real nodes. In this case, recalculating the load of each physical node might be a costly operation. Instead of this, we reassign the load for each node based on the cost of hits (or virtual IDs) in the first ring. The next section describes how to measure the costs.

5.4.2.1 Calculating operation cost

We choose the access frequency (*AF*) of each hit and the number of files distributed at a hit ($P_{storage}$) as the characteristic variables for selecting the replication strategy. We calculate $AF = \frac{C_A}{T}$ (refers to the number of access C_A per time unit during *T*), and count the number of files per hit as the capacity storage of nodes. The operation cost (*OC*) is defined as the overall cost for a node. We calculate the load requirements related to *OC* by the following formula

$$OC = AF + P_{storage} \qquad (5.6)$$

It can be seen from the formula that a hit is selected to a replica mainly based on

- The type of replication that the node has to store if hit *i* is assigned to it ($P_{storage_i}$).
- The number of requests submitted to a hit *i*.

If *OC* of a hit is small, a node associated with this hit is not queried frequently. In other words, there is only a light load coming from this hit. Hence, to improve performance while ensuring balancing distribution and high availability, we need to assign new replicas with the highest priority to the smallest hit. Other factors are also taken into account such as response cost, broadband usage, and CPU usage but for testing purposes, we restrict it to the mentioned criteria.

5.4.2.2 Assigning replication and adjusting the ring

The basic idea behind the data replication scheme is that the first ring is used to control the workload balance and data distribution. We have the cost that replica *i* has for the hit *j*. The constraint in the assignment is the space which acts as the cost. So the problem is, we have a value (i.e the operation cost) for replica *i* and hit *j*. There is a weight associated with each replica c_i and the hit has a capacity h_j (i.e the space) to store replicas. We expect to maximize the values as follow

$$\text{maximize} \sum_i \sum_j x_{ij} oc_{ij} \quad (5.7)$$

$$\text{subject to} \ \sum_i \sum_j x_{ij} c_{ij} < h_j \quad (5.8)$$

$$x_{ij} \in 0,1 \ (5.9)$$

$$\sum_j x_{ij} = 1 \ (5.10)$$

Regarding constraints to selected hits, we initially choose the space (the number of files per hit) as the cost. So we have the weight c_i of the replica i and the costs associated with each hit j OC_{ij} which is defined as the knapsack problem (Sinha and Zoltners, 1979) and it is NP-hard. We use a greedy solution to solve this. We calculate operation costs as follows.

$$OC_{ij} = \frac{oc_{ij}}{c_j} \ (5.11)$$

We sort the operation cost in decreasing order. We iterate each replica and assign corresponding replica to the server which has space. Thus, replicas are first assigned to light load nodes and then higher load nodes will be selected.

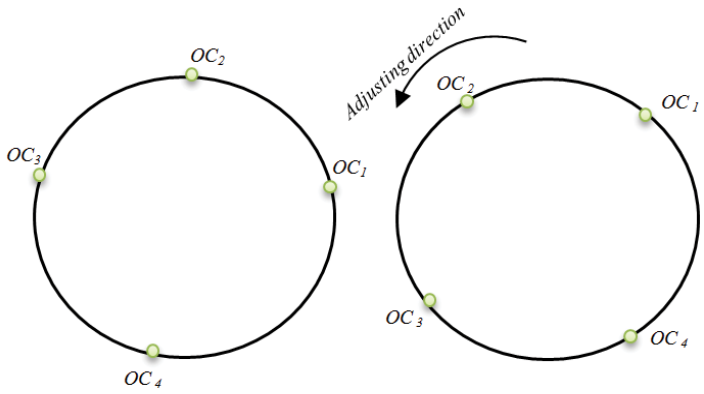


Figure 5.11 Steer the ring when inserting a new replica

We model adjusting the ring the same as controlling the steering of the car in order to get the right direction. It is also performed in parallel with the assigning task to optimize the system performance since replicas with the highest priority are stored for lightest nodes. Figure 5.11 illustrates steering the ring in different directions. When a new replica is inserted, the costs of a hit are recalculated meaning that the ring is steered on the left or right to keep the system balance.

We also define a parameter α as the proportion of the number of replicas N_r which has a great impact on the system performance in terms of workload balance and distribution. This is

because α is used to recalculate the cost of hits after a period T_{OC} and to steer the ring in order to keep a balanced status of the system. The computation for T_{OC} is derived as follow

$$T_{OC} = \alpha * N_r \quad (5.12)$$

with N_r is the number of replicas in a request, α is a tuneable parameter to balance the distribution and the computation cost upon N_r $(0 \leq \alpha \leq 1)$.

By tuning the parameter α, we can observe how the parameter affects the system performance so that we can select an optimal value for α to maximize the performance with a different number of replicas. In the next section, we will show the impact of the parameter α on the system performance for data replications in our experiment. We will verify the optimal α through the simulation results.

5.5 Simulation results

We run the HBFC scheme on P2P systems with different parameters to validate our design. The results of the HBFC scheme are then compared with the original P2P system to determine the performance. For the virtual ring, the djb2 hash function (Bernstein, 2015) is used to hash data keys into the first ID space. We apply the HBFC scheme in P2P system based on Chord (Stoica et al., 2001, JChord, 2016) and modified as our design scheme and compare its performance to the original P2P system which is also following Chord without modifications. The experiment programs are coded using the Java programming language. The results of the HBFC scheme are compared to those of the original P2P system in terms of scalability and system performance. Parameters of Chord's hash function are also processed as normal to hash only the *s_id* to the second ID space. However, we modify the process to attach the file's data key with the ID to allow the lookup process for the requested files in peers. The simulations are based on the following parameters: the number of nodes, the number of files in the systems and the number of requested files.

The idea is to identify the performance efficiency of the HBFC scheme in distributing and searching a number of files compared to original P2P systems. We run the simulations with different parameters. In the first simulation, we select a fixed number of nodes and number of files but input a varying number of requested files to evaluate the performance of the systems. In the second simulation, we increase the number of files in the systems with the same number of nodes and the same number of requested files. In the third simulation, we increase the number

of nodes with the same number of files in the systems and the number of requested files.

5.5.1 HFBC scheme

We perform two simulations. In the first simulation, we conduct the simulation in clustering files to determine appropriate distribution rates by adjusting parameters. In the second stage, we apply the HBFC scheme in the P2P system and compare its performance to the original P2P system which follows Chord without modifications.

Models compared. the HBFC scheme based on Chord (Stoica et al., 2001) in P2P system the original P2P system based on Chord without modifications

Hash functions. djb2 proposed by Dan Bernstein (Bernstein, 2015) is selected as the first hash function to cluster data key into hits. It is good for string hash functions and it has excellent distribution and speed on many different sets of keys and table sizes. The table sizes are defined based on α and β parameters of the HBFC scheme in order to obtain the expected file distribution rates. SHA-1 64 bits is selected as the second hash function to provide unique hash values.

Datasets. For experimental purpose, data keys (file names) are generated randomly to ensure unique filenames.

Parameter Settings. *Computing the lookup duration*. The duration for a request is calculated as $= t_h + t_s$, where t_h denotes the time to hash data keys in the request and t_s denotes the time to search nodes storing files in the request.

In all the experiments, we keep $\alpha = 0.85$ and $\beta = 0.95$ as a general setting unless directly stated otherwise. For the other parameters, we increase values to verify system performance.

5.5.2 Clustering files with parameters

We run the simulation within the same n_n value and various n_f values. The simulation results are presented in Table 5.1.

Table 5.1 Simulation results of two hashing rounds

n_f	α	f_c	n_c nodes ($\alpha * n_n$)	n_n	$n_c * f_c$ files	$\overline{f_c} * \overline{n_c}$files	Rates	$\overline{n_c}$ nodes
10000000	95000	112	85000	100000	9500000	500000	33	15000

250000000	95000	2794	85000	100000	237500000	12500000	833	15000
1000000000	95000	11176	85000	100000	950000000	50000000	3333	15000
2000000000	95000	22353	85000	100000	1900000000	100000000	6667	15000
3000000000	95000	33529	85000	100000	2850000000	150000000	10000	15000
5000000000	95000	55882	85000	100000	4750000000	250000000	16667	15000
10000000000	95000	111765	85000	100000	9500000000	500000000	33333	15000

As the distributions of the files are different depending on n_f, we could achieve the expected distribution rate of n_c and f_c based on the following justifications: (1) if n_f is small, f_c needs to be set equal to the expected rate. This means that n_c is a small percentage of n_n since there is a small number of files stored in each node, and (2) if n_f is large, there are enough files to be distributed and we could achieve the expected number of clusters n_c and the percentage of files in n_c. In general, the average number of files stored in a cluster f_c increases as a rate with the increase of the total number of files in the system n_f. In other words, the HBFC scheme is more efficient when applied to the large-scale systems.

When we have validated the working of the proposed system and understand the way file clustering and distribution can be managed and controlled on the virtual ring, we are able to distribute files into real nodes by adjusting α parameter to achieve f_c files in a cluster. In addition, with n_f and n_c we can design to choose α and obtain the expected f_c. Our scheme enables distributing and retrieving data efficiently compared with other models and original P2P systems as demonstrated by simulations in the next section.

5.5.3 The HBFC scheme and the original P2P system

Test case 1. The number of nodes and the number of files are fixed; the number of requested files is changed.

We initialized two sets of tests in which the number of files (*10000000 files*) and the number of nodes (*5000 nodes*) are unchanged. The number of requests in the system increases from *1500* to *10000* requested files. Figure 5.12 illustrates the execution time of the data lookup process for a different number of requested files. The x-axis represents the number of requested files, and the y-axis represents the execution time. As can be seen from the figure, the time cost

of original P2P (*T2*) is approximately double compared to that of the HBFC scheme (*T1*). It demonstrates that within the same number of requests, the HBFC scheme performs more efficiently than the original P2P since many files are distributed in one or several clusters.

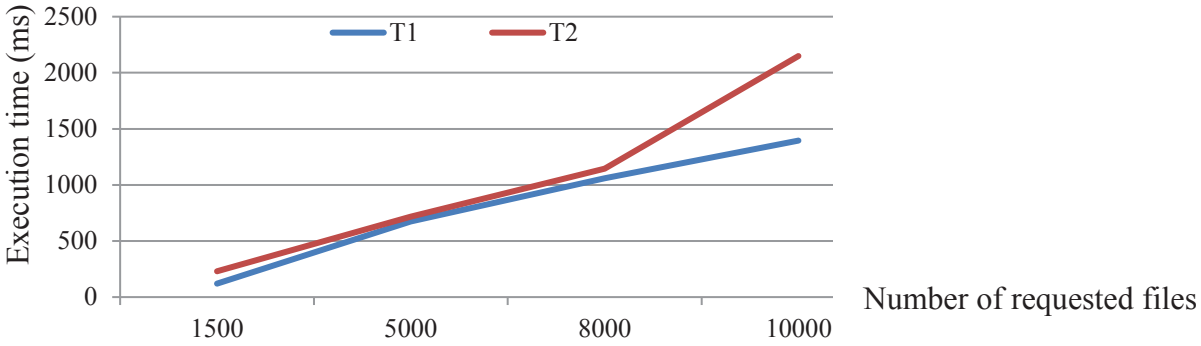


Figure 5.12 Time cost of the HBFC scheme and original P2P for data lookup process within different number of requested files

Test case 2. The number of nodes and the number of requested files are fixed; the number of the files is changed

In this experiment, we initialized two sets of tests in which the number of requests (*1500* requests) and the number of nodes (*5000* nodes) is unchanged. The number of files in the system is increased from *10000000* to *30000000* files. Figure 5.13 illustrates the execution time of the data lookup process for a different number of files. The x-axis represents the number of files, and the y-axis represents the execution time. As we can observe, in fact, along with the increasing of the number of files in the system, the time costs in searching files remain approximately the same for both schemes namely *T1* (*123 ms*), *T2* (*229ms*) respectively. However, *T2* (*229ms*) is still twice longer than *T1* (*123ms*) despite the fact that data lookup space is increased in terms of the number of files. It is concluded that the HBFC scheme is more efficient than the original P2P system in terms of scalability.

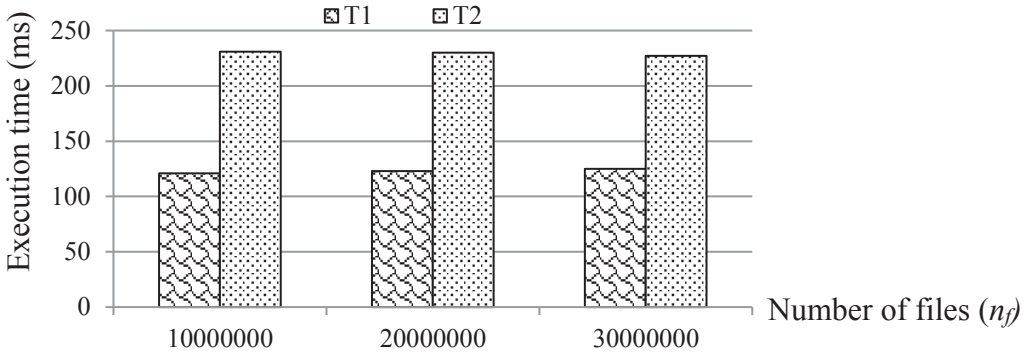


Figure 5.13 Time cost of the HBFC scheme and the original P2P for data lookup process within different number of files in the system

Test case 3. The number of the files and the number of requested files are fixed; the number of nodes is changed.

In the third case, we initialized two sets of tests in which the number of requests (*1500 requests*) and the number of files (*10000000 files*) are unchanged. The number of nodes in the system is increased from *5000* to *10000* nodes. Figure 5.14 illustrates the execution time in the data lookup process for a different number of nodes. The x-axis represents the number of nodes, and the y-axis represents the execution time. Generally, the execution time grows linearly with the increase of nodes in the system but in our proposed scheme there is a slight increase from *120 ms* to *166 ms* compared to the original P2P system from *230 ms* to *349 ms*. Similarly, *T2* (*322 ms*) is still approximately twice longer than *T1* (*154.5 ms*) in this case.

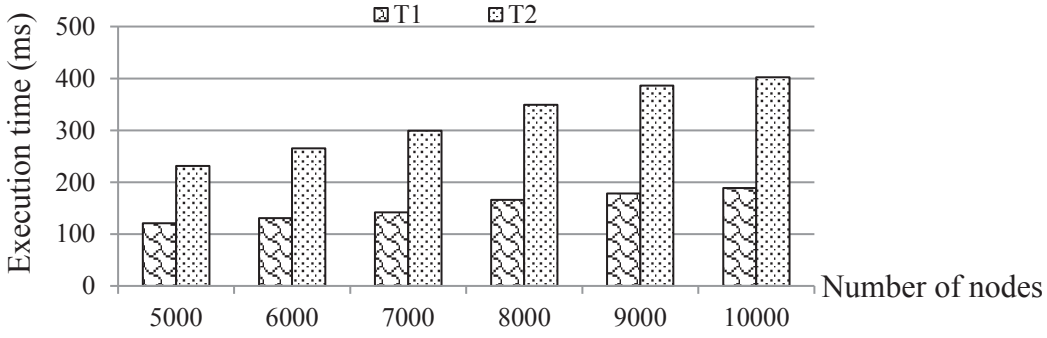


Figure 5.14 Time cost of the HBFC scheme and the original P2P for data lookup process within different number of nodes in the system

Overall, it is demonstrated that our proposed scheme is more efficient than the original P2P in all cases. Specifically, when the size of the system is increased together with the increase in the number of requests, the executing time of original P2P system increases rapidly in

comparison with that of the HBFC scheme.

5.5.4 The simulation results of Data replication scheme

We used djb2 hash function proposed by Dan Bernstein (Bernstein, 2015) to cluster data keys into hits at the first ring. It is good for string hash functions and it has excellent distribution and speed on many different sets of keys and table sizes. Data keys are filenames which are generated randomly to ensure unique filenames. The P2P system implementation is based on Chord (Stoica et al., 2001) and modified as our design scheme. The experiment programs are coded using the Java programming language. Replicas are generated within priorities and inserted into nodes based on the hits on the first ring. For simulation, we used three separated tables to store different types of replication information including hit_ID and replica name. To get some comparative statistics, the replication mechanism based on Chord (Stoica et al., 2001) in the P2P system was also run with the same system parameters and the number of replicas, in which replicas are assigned to nodes based on Chord mechanism.

5.5.5 Comparison of the proposed scheme and the replication mechanism based on Chord

We initialized tests in which the number of files (*1000000 files*) and the number of nodes (*5000 nodes*) in the system are fixed for two replication schemes. Notably, the initial data was firstly distributed into hits by the HBFC scheme using Chord (Stoica et al., 2001) in the P2P system.

We carry out three experiments with a different number of replicas to compare the performance of our proposed scheme with the replication mechanism based on Chord.

In the first experiment, we compare the operation cost of the two schemes. We increased the number of replicas from *1000* to *20000* replicas for both schemes in order to identify the changes of the load at hits and verify the distribution rates of the replication schemes. α is set equal to 0 to obtain the best balance for the proposed scheme.

Figure 5.15 illustrates the distribution and workload balance of the schemes when inserting new replications comparing to the original distribution. The x-axis represents the number of replications inserted into the system, and the y-axis represents OC at hits. With the same number of replicas, the proposed scheme returns the performance which is in order of operation costs better than that of the original scheme. Especially, the highest *OC* (*Max1*) and the lowest *OC* (*Min1*) are approximately the same, namely *253* and *194,* respectively. However, in the original

scheme, the gap between the highest *OC* (*Max2*) and the lowest *OC* (*Min2*) is very large, namely 1738 and 1 respectively. As a consequence, when nodes have large cost, the original scheme suffers severely in terms of both workload balance and distribution.

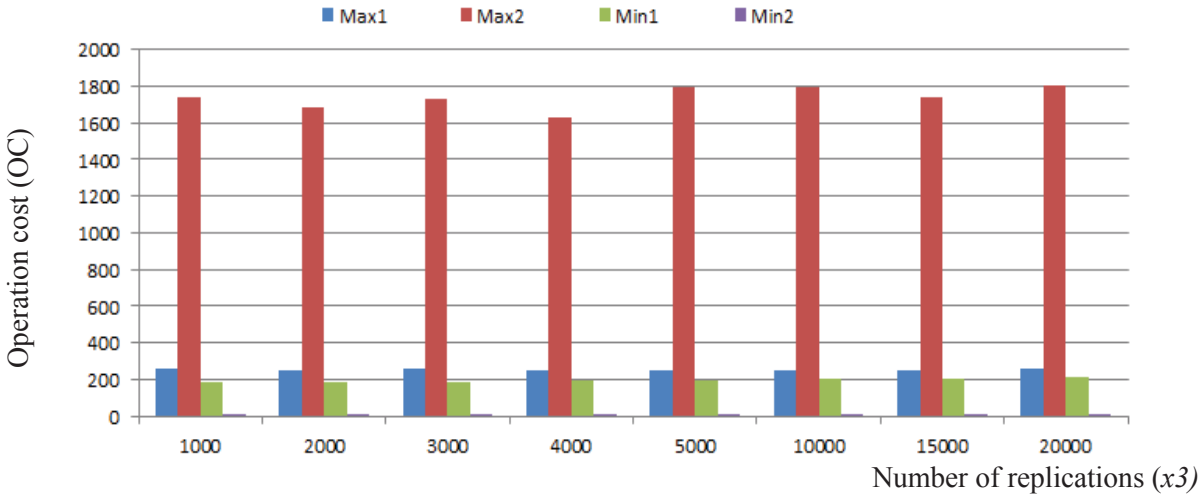


Figure 5.15 The distribution and workload balance of the schemes

In the second experiment, we run the same number of replicas to identify the number of heavy load nodes in the proposed scheme (*HC1*) and the original scheme (*HC2*). The heavy load nodes have OC higher than the average OC. Interestingly, the number of heavy load nodes in the proposed scheme achieves zero nodes for all requests (Figure 5.16). This is because the ring is rebalanced after each request and nodes share approximately the same load. In contrast, the original scheme experimented with a large number of nodes (*1099* out of *5000* nodes). It can be concluded that our scheme has better workload balance and distribution than the original mechanism. With our replication scheme based on the HBFC scheme, hits are rebalanced at the time a new replicate is inserted to nodes while still maintaining the balanced distributions among nodes. Nevertheless, with the original replication scheme when the access pattern is very large, the replicas may not be evenly distributed and might create hotspots leading to workload unbalance among nodes and a bottleneck in the systems due to many requests at the same node.

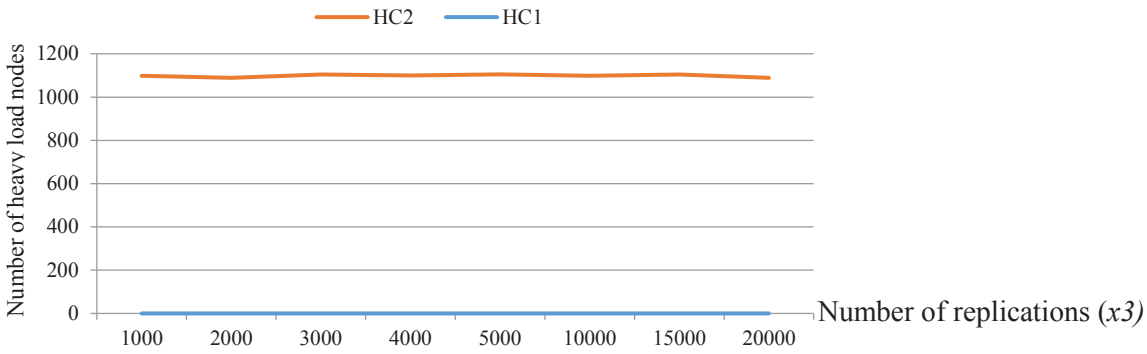


Figure 5.16 The number of heavy load nodes of two schemes

Figure 5.17 demonstrated the replication latency of the two schemes. We observed that the execution time of both schemes increases linearly with the increase of the number of replicas. If we set the parameter $\alpha = 0$ in the proposed scheme, the ring is steered inserting a new replica. Consequently, the execution time of the proposed scheme (*T1*) is much larger than that of the original scheme (*T2*). Especially, *T1* (*4816 ms*) approximately doubles T2 (*2574 ms*). Hence, there might be more computation overhead and latency to maintain the load balancing for the proposed scheme compared to the original scheme. In order to address this issue, the parameter α needs to be set as a percentage of the number of replicas. We conduct the following experiments to identify the effect of T_{OC} on the performance of the system. In the following section, we will verify the impact of the parameter α on the system performance of the proposed scheme.

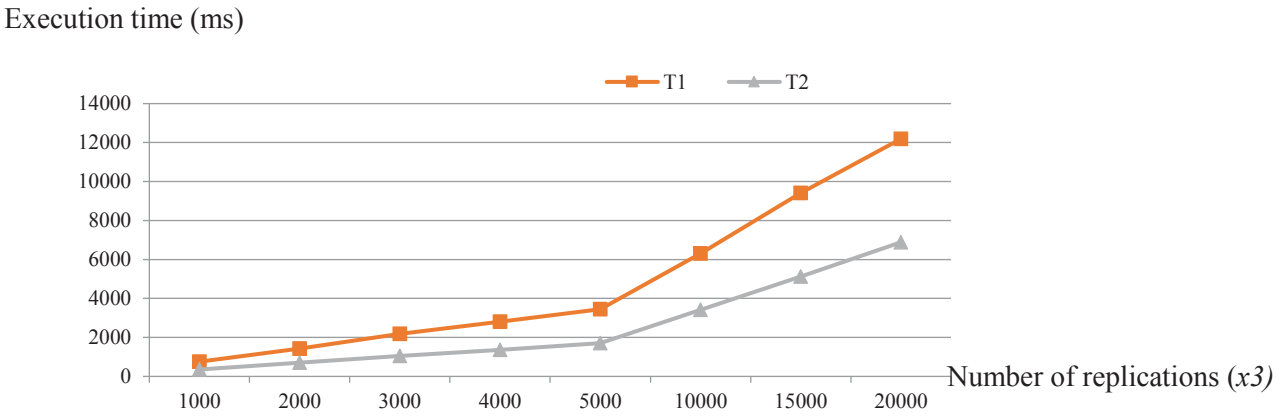


Figure 5.17 The duration when inserting replicas of two schemes

5.5.6 Verifying the impact of the parameter to system performance

In this section, we compare the workload balance and distribution of the proposed scheme with

different values of α. To analyse the overhead and load of the proposed scheme, we increase the number of files in the system (*10000000* files) and also increase the number of replicas from *10000* to *50000* replicas. In fact, we only recorded the latency of steering the ring and the load at hits in this experiment.

Figure 5.18 shows the execution time when α increases from *0.0001* to *0.75*. As illustrated in the figure, the execution time of our proposed scheme is monotonically increasing with respect to α. This is because when the interval of the recalculating process of the ring increases, the system does not need to steer the ring frequently. For example, when $\alpha = 0.75$, the system has the smallest latency (*148 ms*). However, it reaches the highest latency (*2950 ms*) when $\alpha = 0.0001$. Although the execution is small when α *is large,* we observed that there is a small number of nodes which suffered from the heavy load. Since the ring is steered only a few times during performing the request, a large number of replicas are stored at these nodes.

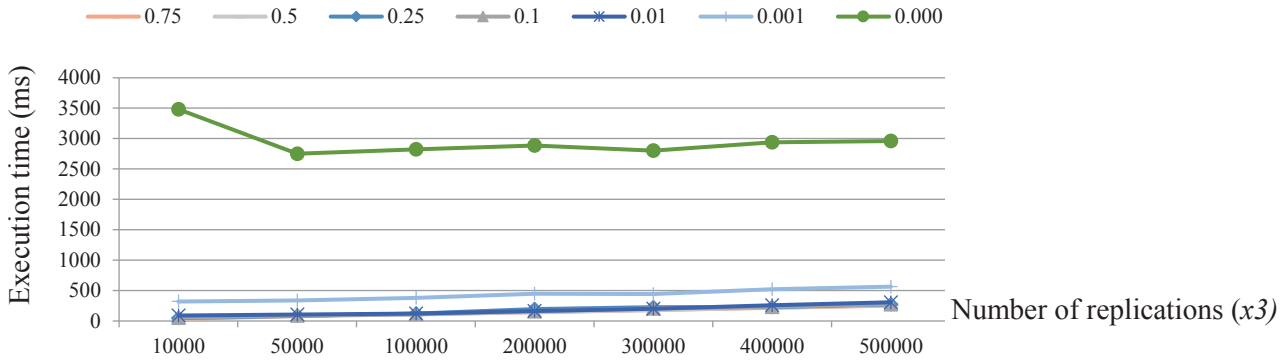


Figure 5.18 The duration of the proposed scheme with different α

Figure 5.19 presents the number of heavy load nodes with different values of the parameter α. When α is small, we observe that the number of heavy nodes reduces to zero. For example, when α=*0.0001*, the system achieves the workload balance with no heavy nodes. Although there is a small number of heavy nodes when α increases, these nodes experience extremely heavy load due to housing large number of replicas. In other words, the ring is not steered frequently to balance the workload and distribution of the system. As a consequence, hotspots create the performance bottlenecks in the system.

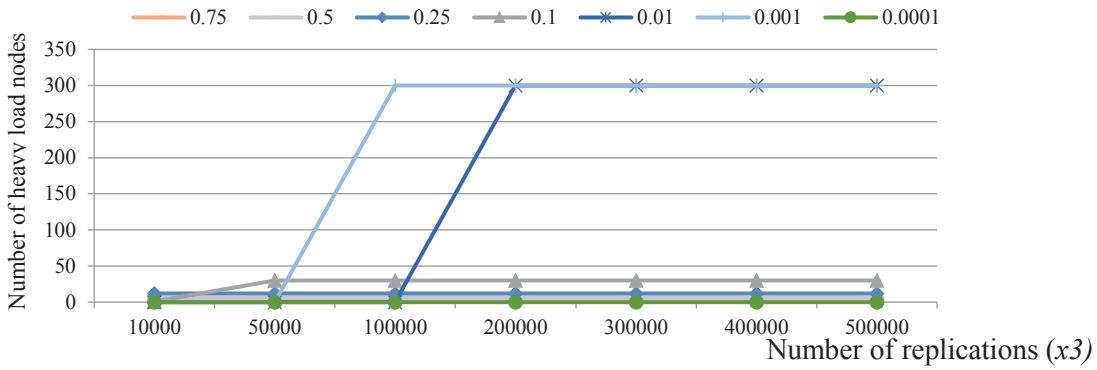


Figure 5.19 The number of heavy load nodes of the proposed schemes ranked after T_{OC}

We observe the maximum OC (*Max1*) at hits with different values of the parameter α. As can be seen in Figure 5.20, the system experiments show the largest OC when $\alpha \geq 0.25$ with the highest OC value (*169187*). In contrast, the system has smallest load nodes when $\alpha \leq 0.01$ with the lowest OC value (*2203*). It is concluded that the larger α is, the more expensive OC the system has to perform.

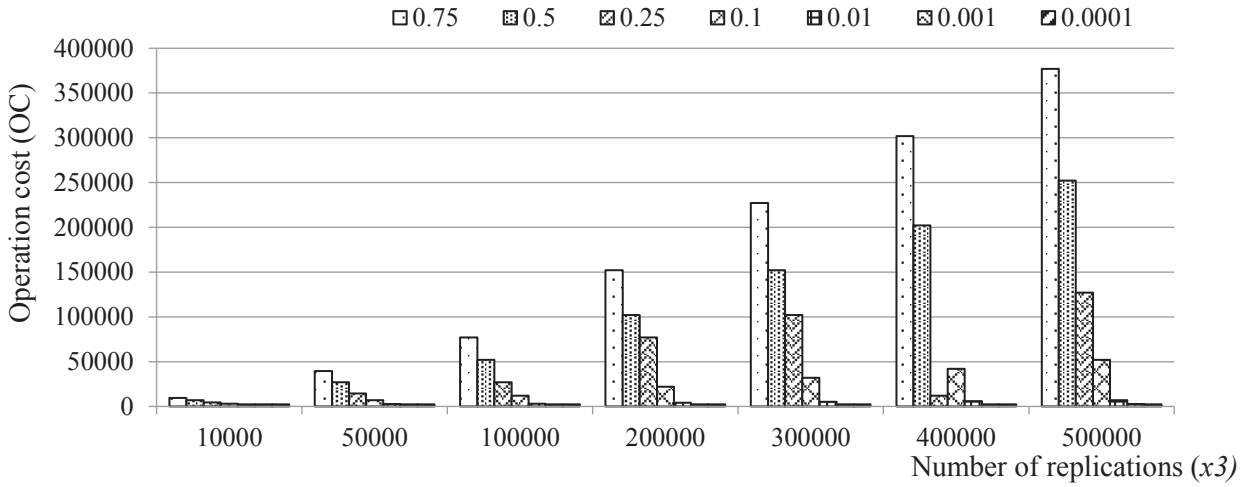


Figure 5.20 The maximum OC (*Max1*) at hits with different values of α

Figure 5.21 shows the minimum OC (*Min1*) with different values of the parameter α. It is approximately the same OC (around *1870*) for most values of α except when $\alpha = 0.0001$ (*2115*). It can be explained that when we steer the ring more frequently, the system maintained the balance status. Thus, the gap between *Max1* and *Min1* reduces, namely *2203* and *2115* respectively.

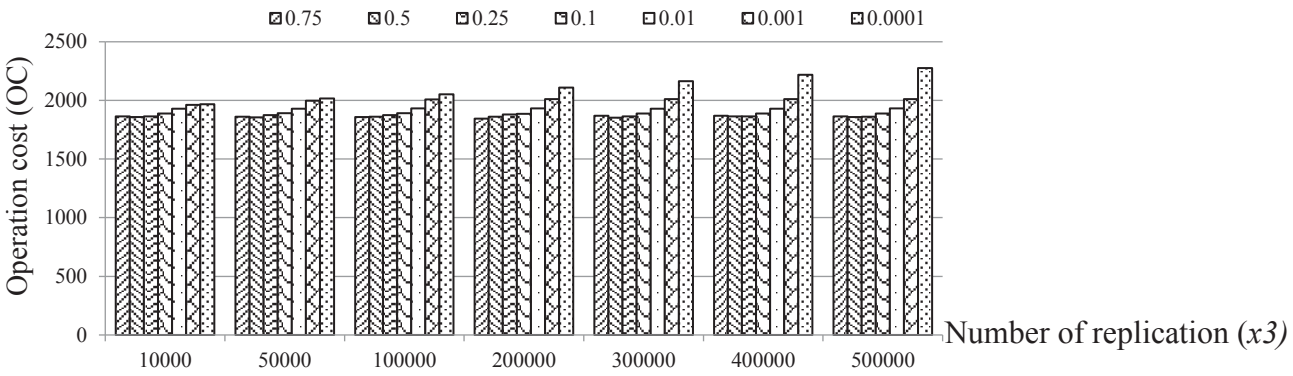


Figure 5.21 The minimum OC (Min1) at hits with different values of α

Although the minimum OC (*Min1*) only presents a small difference with various values of α, the gaps between *Max1* and *Min1* are huge when the parameter α and the number of replicas increase. It can be seen from Figure 5.20 and Figure 5.21 that with the increase in the number of replicas, the larger α becomes, the bigger gap of OC appears in the proposed scheme. For example, with the number of replicas at *500000*, when we set α *equal to 0.75*, the gap is *375144* but it is only *50* if $\alpha = 0.0001$. Thus, it is not efficient in terms of workload balance and distribution when α is set too large.

To summarize, the cost of our proposed scheme is much lower than that of systems without clustering since it performs in virtual IDs and replications are assigned dynamically to physical nodes without reconfiguring the whole system. Thus, the excess latency in recalculating the operation cost at hits may be neglected for the workload balance.

5.6 Summary

In this chapter, we have presented the data distribution scheme and data replication scheme. The HBFC scheme is proposed to meet low computational complexity requirements and cluster files in a controlled way to suit the specifications of a real system. The replication scheme is designed to insert new data replications to nodes in a balanced way by utilizing the ring of the HBFC scheme. The HBFC scheme takes advantage of the hash functions and the computation on the virtual IDs, calculating and ranking workloads. Therefore, the proposed schemes can reduce the computation overhead on the physical servers while still achieving performance efficiency in terms of distributions and workload balance. The two proposed models play significant roles in distributing, retrieving and accessing data in the cloud environment. Beside distributing data, we also consider cloud resources for processing sensitive

data to satisfy security requirements and address users' concerns regarding their sensitive information. The next chapter will describe in detail our proposed scheme to process sensitive data for big data applications.

CHAPTER 6 A TRUST-BASED RESOURCE SCHEDULING MODEL FOR BIG DATA PROCESSING WITH MAPREDUCE

Security and privacy have become a great concern in cloud computing platforms in which users risk the leakage of their private data. The leakage can happen during data processing within a cloud or transferring between different cloud infrastructures, ranging from private to public clouds. A hybrid cloud was used in (Xu and Zhao, 2015, Zhang et al., 2014) to preserve data privacy by processing sensitive data at private clouds only while non-sensitive data was processed at public clouds. It is not desirable, however, to schedule all processing tasks for sensitive data at only private clouds while public clouds can provide adequate protection services but they remain idle. The systems may incur bottlenecks or longer delay when the amount of sensitive-data is huge or data operations are heavily performed at private clouds. It is assumed that the delegated cloud can be trusted at a given security level when processing data of another cloud. Therefore, a measurable metric can be assigned to cloud resources so that CSPs allocate appropriate secured resources to client's requests.

In this chapter, we present a trust-based scheduling model for MapReduce in big data processing tasks. Specifically, we first quantify and assign the sensitive values for data and trust values for map and reduce slots. We then compute the trust value of each resource employed in the big data processing tasks. Depending on the data's sensitivity level of a task, the task requires a given level of trust (i.e., higher sensitive data requires servers/slots with higher trust level). The MapReduce scheduling problem is then formulated as the maximum weighted matching in bipartite graph theory that aims to maximize the total trust value over all possible assignments subject to various trust requirements of different tasks. The problem is known to be NP-hard. To tackle it, we observe that within a computing node (VM), slots share the same trust value granted from the secured transformation phase. This helps reduce the number of slot nodes of a weight bipartite graph. Leveraging this fact, we propose an efficient heuristic

algorithm that achieves 94.7% of the optimal solution obtained via exhaustive search. Extensive simulations show that the trust-based scheduling scheme provides much higher protection for data sensitivity while ensuring good performance for big data applications.

The remainder of the chapter is organized as follows. Section 6.1 presents a trust-based data processing with MapReduce and definitions. Section 6.2 presents the proposed trust metric for cloud resources. Section 6.3 presents the proposed scheduling scheme. The trust-based scheduling framework for realization is presented in section 6.4. The simulation results and evaluations of our proposed framework are in Section 6.5. The chapter concludes in Section 6.6.

6.1 A Trust-based data processing with MapReduce and definitions

A MapReduce program consists only of two functions (Map and Reduce), that are written by the user to process key/value data pairs. The data set is stored in a collection of partitions of a distributed file system deployed on each processing server in the cluster. The program executes in the Hadoop framework in the following manner: The Map function processes key/value records from an input file and then outputs a set of intermediate new key/value records. These output records are partitioned into disjoint buckets by applying a (hash) function to the key of each output record and written to the server's local disk. In fact, multiple instances of the Map function run on different servers of the cluster and each map instance is assigned a distinct portion of the input file by the scheduler to process. The second phase of a MapReduce program executes instances of the Reduce program. The input for each reduce instance is output files from the map server local disks. Each reduce processes the records assigned to it and then writes records to an output file in the distributed file system as part of the computation's final output.

A MapReduce job comprising a number of map and reduce tasks is executed on a cluster of multiple servers. The job's computation consists of a map phase followed by a reduce phase. In the map phase, each map task is allocated to a map slot on a machine and processes a portion of the input data. In the reduce phase, the outputs of the map phase with the same key are processed by a reduce task allocated to a reduce slot. One constraint is that the reduce phase of a job cannot begin until the map phase ends. Hadoop uses Namenode as the master node and Datanodes as worker nodes.

Let T denote the set of map tasks and reduce tasks, and S denote the set of slots on heterogeneous

nodes available in the cluster, respectively. The processing time and the trust of nodes may not be the same. Slots of a node are assigned the same trust value while a task performing on the sensitive data is also assigned a required trust value associated with the data.

Table 6.1. Summary of notation

Basic Notation	Description
J	A set of jobs with different security requirements.
T	A set of pending tasks of submitted jobs.
$\|T\|$	The total number of the task in *T*.
u_i	A user *i* submitting data processed at big data applications.
t	A pending task in *T*.
d_k	The input data set *k* (split) of task *t*.
S	A set of feasible slots satisfying the trust requirement of the set of tasks in *T*.
$\|S\|$	The total number of feasible slots in *S*.
s	A feasible slot in *S*.
TV_{ij}^{jk}	The trust value of the slot *k* provided by cloud C_j for a request from cloud C_i.
δ_{jk}	The trust value of the slot *k* classified by cloud C_j.
θ_{ij}	The trust value that cloud C_i assigns for resources of cloud C_j.
τ_{ij}	The trust value assigned to resources of cloud C_j when cloud C_i runs the third-party cloud ranking system on C_j.
$TV_{d_k}^{l}$	The trust value required by input data d_k with the sensitive level *l*
$TV(t,s)$	The trust value obtained when a task *t* is executed on a slot *s*.
n_s	The node with available slot *s* to run a task *t*
C_i	A CSP *i* providing/requesting slots for big data applications
$w(t,s)$	The weight of the edge (t,s).
$x(t,s)$	The $\{0,1\}$ variable indicates whether the edge is selected *s*.

Definition 1. Each data d_k associated with a sensitive level l which specifies a certain trust requirement

$TV_{d_k}^{l}$ for assigned processing slots.

Definition 2. Each task *t* associated with data d_k requires appropriate trusted slots that satisfy the required sensitive level *l*. If the execution of *t* involves resources less than $TV_{d_k}^{l}$, it is a low-trusted task.

Definition 3. Each slot *s* is assigned a trust value that specifies the provided trust when tasks are executed on it.

Definition 4. Each task *t* on node *x* is also associated with a data retrieval limit which reflects the maximum number of hops to access the data on node *y* from node *x*, *h(x,y)*.

Definition 5. Unlike the traditional MapReduce problem, the Trust-based Scheduling MapReduce (TSMR) problem considers the tasks and slots of a system with different security requirements and execution performance, respectively. The main objective of the TSMR problem is to find an optimal task scheduling (allocation) strategy which can maximize both the security level and highly trusted resources utilization when performing a number of map and reduce tasks. In fact, it also minimizes the number of low-trust tasks due to resource availability.

6.2 Trust metric for cloud resources

In this section, we present a novel composite trust assignment for cloud resources. First, we propose the concept of trust space as a range of trust values which then are used to assign for cloud resources. Second, we present how trust components are defined, mapped and calculated by taking into account three trust components: 1) the third-party cloud ranking systems; 2) the trust assignment among CSPs; 3) the cloud resource classifications. Furthermore, we introduce a mapping mechanism to translate data sensitive requirements into the defined trust space.

6.2.1 Trust Space

As the trust value may be calculated based on different metrics and trust assignments, it is essential to define a uniform trust unit which describes trust values of cloud resources. These trust values shall be calculated by using different mapping functions.

Definition 6. Trust space ***TP*** is a set of ***{T, D}*** where ***T = [0,1]*** is a collection of trust, called ***trust range***. ***D*** represents the description of a trust value associated with *T*. In the ***trust range***,

0 represents *complete no-trust* and *1* represents *complete trust*. For any element x_i, there exists a random number $y = \mu(x_i)$. x_i is the trust value in the trust range. The distribution of T is called trust cloud.

Depending on SLAs, cloud resources are classified into different trust levels. For example, Table 6.2 presents six trust levels and how they are mapped into *TP*. In this, *T* is defined a trust range [*0.02,1*] to present for trust levels. *D* is represented by Credibility.

Table 6.2. Trust levels, Credibility values in trust range

Trust levels	Credibility	Value on Trust range
1	Completely	[0.02]
2	Not very credible	[0.2,0.4]
3	Basic credible	[0.4,0.6]
4	Fairly credible	[0.6,0.8]
5	Credible	[0.8,0.9]
6	Very credible	[0.9,1]

6.2.2 Assigning Trust Values to Cloud Resources

6.2.2.1 Trust assignment based on the third-party cloud ranking systems

According SLAs established between clients and CSPs, cloud resources could be evaluated and classified into defined trust levels. Suppose that there are n CSPs C_1, C_2, …, and C_n and k ranking q_1, q_2,..,q_k. When a cloud C_i needs more resources to complete requested tasks, it searches for available resources from other CSPs such as C_1, C_2, …, C_{i-1}, C_{i+1},.., C_n. However, it needs to justify which resources and from which CSPs should be employed in this process to satisfy required trust levels. Thus, the third-party cloud ranking system is a reliable tool which could provide a consistent trust level regarding CSPs' ranking. Since there are different cloud ranking systems on the market, a cloud C_i may receive a different ranking of C_j when running these systems. This leads to an inconsistent ranking of a CSP. In this work, we use the SPECS framework (Casola et al., 2015, Modic et al., 2016) which enables the deployment of secure cloud applications covered by a Security SLA and ranks CSPs based on the security Service Level Objectives (SLOs). The security SLO can be assigned values as security levels. CSPs are ranked for the best match with users' requirements. Users consume cloud services and resources based on SLAs established with CSPs. Hence, security requirements are specified as dedicated SLAs, termed Security level agreements (*SecLAs)*. Service based assurance is provided under SecLAs and represented by a collection of security statements. The security statements define services to be provided by CSPs, i.e., security SLOs.

The CSPs ranking mechanism in (Casola et al., 2015, Modic et al., 2016) is based on the Analytic Hierarchy Process (AHP) which uses quantitative criteria to compare decision criteria and weights assigned to each criterion. The SecLA assessment and the ranking of CSPs are performed as follows: 1) Define user SecLA and CSP SecLA requirements; 2) Quantify the security level associated with each SecLA; 3) Apply ranking algorithm based on AHP. Based on the exact required minimal level and all possible higher levels for SLOs that CSPs can provide with their resources, CSPs can be scored according to SLOs. A CSP C_i with S_i SLOs is evaluated and ranked as r_i by SPECS framework (Casola et al., 2015, Modic et al., 2016) as follows

$$r_i = SCORE(C_i, S_i)$$

as SPECS framework also allows a CSP to be ranked by other CSPs with their SLOs. Hence, we define the ranking C_j run by cloud C_i with its S_i SLOs as follows

Definition 7. A cloud C_i runs SPECS framework on cloud C_j upon S_i SLOs of C_i. Hence, the SPECS framework provides the ranking of C_j according to the overall quality of provided security features. The ranking meets with the Cloud Service Customers (CSCs) security requirements of cloud C_i.

$$r_{ij} = SCORE(C_j, S_{ij}) \quad (6.1)$$

where *SCORE* is a function that gives the security levels overall assessment and a final ranking of the cloud C_j. S_{ij} consists of SLOs of cloud C_i and cloud C_j. r_{ij} is the ranking of C_j when C_i runs SPECS for C_j and S_{ij}.

As SPECS only provides the ranking of CSPs, cloud resources need to be assigned trust values enabling them to join the resource selection process. Suppose that there is a set of ranking R and TP. Hence, we define one-one mapping as $f: R \rightarrow TP$ which maps an obtained ranking into a trust value. Suppose that r_{ij} is the ranking value that a cloud C_i received when running SPECS on the cloud C_j. The trust value assigning to cloud resources of cloud C_j is calculated as follow

$$\tau_{ij} = f(r_{ij}) \quad (6.2)$$

where f is a function that maps the rank r_{ij} of CSP C_j to a trust value τ_{ij}, with $\tau_{ij} \in TP$. f must be linear and monotonic increasing.

6.2.2.2 Trust assignment among CSPs

A cloud C_i may have its own rankings of other CSPs. Therefore, we use another trust assignment scenario to calculate trust values when C_i use resources from $C_1, C_2, \ldots, C_{i-1}, C_{i+1}, .., C_n$. The trust evaluation is employed by both combining direct trust and recommendation trust. C_i calculates the trust value for resources of C_j as follow

$$\theta_{ij} = T(C_i, C_j) \qquad (6.3)$$

If C_i run trust evaluation for $C_1, C_2, \ldots, C_{i-1}, C_{i+1}, ..., C_n$, we obtain a trust matrix A as follow

$$A = \begin{pmatrix} 1 & \theta_{12} & \ldots & \theta_{1n} \\ \theta_{21} & 1 & \ldots & \theta_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \theta_{n1} & \theta_{n2} & \ldots & 1 \end{pmatrix}$$

If cloud C_i does not trust cloud C_j, it assigns *0* to all resources of cloud C_j ($\theta_{ij} = 0$). In addition, two clouds may trust each other differently. Therefore, $\theta_{ij} \neq \theta_{ji}$.

6.2.2.3 Cloud resource classification

Each CSP owns different cloud resources and issues them to clients on demand. When cloud C_i receives requests within trust requirements, it then issues its available resources $S_{i1}, S_{i2}, ...,$ and S_{ip} associated defined trust values. These resources are classified into trust values namely $\delta_{i1}, \delta_{i2,}, \ldots, \delta_{ip}$ ($\delta_{i1} < \delta_{i2,} < \cdots < \delta_{ip}$). The resources are presented by the number of VMs assigned to the application, and the properties of the underlying resources bound to each VM, e.g., CPU, memory, storage, and network resources. Basing on SecLA, CSPs are able to provision VMs matching with users' requirements. There are many parameters to this classification process regarding to SecLA. We have chosen two parameters based on the work (Ullah and Ahmed, 2014) as follows: 1) the *VM container security profile (VMSCP)*, which indicates the physical security of the machines where the VMs will be deployed; 2) the *VM Application security profile* (*VMASP*) which indicates the application level security requirements of the VM. These two parameters are assigned security levels which are mapped into ***TP.*** *VMSCP* represents physical security parameters such as availability of trusted platform module (TPM), hypervisor hardening, and other physical security issues of the data center. *VMASP* depends on security parameters such as password policy, encryption algorithm in use, firewall status, antivirus status. The security value of VM is composed of *VMASP* and *VMSCP*. When C_j provisions resource S_{jk}, the computation of δ_{jk} is as follows.

$$\delta_{jk} = VMSCP_k + VMASP_k \quad (6.4)$$

6.2.2.4 Trust assignment for individual resource

From the discussion above, each cloud resource may be assigned different trust values. The trust values for cloud resources may be obtained from the third-party cloud ranking system, trust values assigned by CSPs on one another and a cloud's own resource classification. We calculate the final trust value for a resource by combining the three above trust values.

Definition 8. The trust value for a cloud resource can be defined as follows: given that the number of CSPs' ranking can be transferred to the trust value as Eq. (6.2), direct trust assignment as Eq. (6.3), and classification of resource Eq. (6.4), then the normalized trust value *TV* of a CSP's resources is derived as follows.

$$TV = \tau \,.\, \delta \,.\, \theta \qquad (6.5)$$

Suppose that cloud C_i requests a resource k from cloud C_j, individual trust value will be calculated as follow.

$$TV_{ij}^{jk} = \tau_{ij} \,.\, \theta_{ij} \,.\, \delta_{jk} \qquad (6.6)$$

6.2.2.5 Example

Example 1. Suppose that there are two clouds C_1, C_2. C_1 needs computation resources from C_2. It first retrieves the ranking of C_2 and then transfers the ranking to a trust value 0.8. C_2 also issues two available slots s_{21} *and* s_{22} which are classified and assigned trust values as 0.8 and 0.6, respectively. In this case, C_1 relies totally on the third-party cloud ranking system. Therefore, $\theta_{12} = 1$. This means C_1 is not concerned about direct trust with C_2. Hence, from Eq. (6.6) the trust values for slots s_{11} *and* s_{12} are calculated as follows

$$TV_{12}^{21} = \tau_{12} \,.\, \theta_{12} \,.\, \delta_{21} = 0.8 \,.\, 1 \,.\, 0.8$$

$$TV_{12}^{22} = \tau_{12} \,.\, \theta_{12} \,.\, \delta_{22} = 0.8 \,.\, 1 \,.\, 0.6$$

Example 2. Parameters of two clouds C_1, C_2 are set the same as Example 1. However, C_1 does not rely on the third-party cloud ranking system. Therefore, $\tau_{12} = 1$. C_1 assigns direct trust value 0.7 for C_2. Hence, from Eq. (6.6) the trust values for slots s_{11} *and* s_{12} are calculated as follows

$$TV_{12}^{21} = \tau_{12} \,.\, \theta_{12} \,.\, \delta_{21} = 1 \,.\, 0.7 \,.\, 0.8$$

$$TV_{12}^{22} = \tau_{12} \,.\, \theta_{12} \,.\, \delta_{22} = 1.\, 0.7 \,.\, 0.6$$

Example 3. Parameters of two clouds C_1, C_2 are set the same as Example 1. However, C_1 relies on both the third-party cloud ranking system and direct trust with trust values 0.8 and 0.7, respectively. Hence, from Eq. (6.6) the trust values for slots s_{11} *and* s_{12} are calculated as follows

$$TV_{12}^{21} = \tau_{12} \,.\, \theta_{12} \,.\, \delta_{21} = 0.8 \,.\, 0.7 \,.\, 0.8$$

$$TV_{12}^{22} = \tau_{12} \,.\, \theta_{12} \,.\, \delta_{22} = 0.8.\, 0.7 \,.\, 0.6$$

6.2.2.6 Assigning Trust Requirement of Tasks within Data-sensitive Levels

Each user's data submitted to the scheduler is provided a sensitive level associated with it. Hence, the sensitive level is mapped to trust space and assigned as trust requirements of the data. Once completing this step, the data is dispatched to jobs and ready for processing. Given data d with the sensitive level l that splits into $d_1, d_2, \ldots, d_p$ and is assigned the same sensitive level. The trust requirement of d_k with the sensitive level l is obtained by the following equation:

$$TV_{d_k}^{l} = \varphi(d_k, l) \quad (6.7)$$

where φ is the function that maps a sensitive level s of the data d_k into a trust value. A task involving the data is also assigned this trust value (called trust requirements) and scheduled appropriate trusted resources prior to trust requirements.

6.3 The proposed scheduling scheme

6.3.1 MapReduce Scheduling Scheme

In this section, we present a new MapReduce scheduling scheme, which considers multiple trust requirements from different MapReduce jobs and various trusted slots from CSPs of different cloud infrastructures. The scheme consists of trust transformation, bipartite graph modeling, and scheduling transformation. As the bipartite graph is used to perform MapReduce scheduling, hence the proposed scheme is called the Bipartite Graph Trust-based Scheduling MapReduce (BGTSMR).

The basic idea of the BGTSMR is given in Figure 6.1. When a job is submitted, the job is associated with a required trust value and a sensitive data to be put in the ready queue. The trust requirement of jobs and data have been defined as in definition 2 and definition 4. After the trust transformation, the job is divided into two sub-trust requirements: a map trust and a reduce

trust requirement. When one or more jobs in the ready queue run their map tasks or reduce tasks, such map and reduce tasks are selected from *T*. If the trust value of slot *s* satisfies the trust requirement of task *t*, the slot s can provide a secured execution to meet with the trust requirement of task *t* of the original job. For all running tasks, their executable slots are kept in the set *S*. Based on sets *T* and *S*, we can form a weighted bipartite graph to represent the scheduling (allocation) relationship between *T* and *S*. By transforming the TSMR to the maximum weighted bipartite matching (MWBM) problem, we can obtain the optimal task scheduling to maximize the trust value, thus, providing high secured resources for users' data when it is processed at different CSPs.

Input: A set *J* of jobs with different trust requirements and a set *S* of slots with heterogeneous trust values.

Output: Trust-aware scheduling for tasks of *J*.

1: **while** jobs run their map or reduce tasks **do**

2: **/* Trust transformation. */**

3: **for** each job *j* of *J* in the ready queue **do**

4: Perform the trust transformation to the set the trust requirements for map and reduce tasks of *j*.

5: **end for**

6: **for** each available slot *s* at CSPs **do**

7: Perform the trust transformation to the set the trust value for slot *s*.

8: **end for**

9: **/* Bipartite graph modelling. */**

10: $T \leftarrow$ select running map and reduce tasks of such jobs.

11: **for** each task *t of T* **do**

12: $s_t \leftarrow$ find appropriate slots associated with the trust value *TV* responding to *t*.

13: Select s_t in *S*.

14: **end for**

15: Form a weighted bipartite graph based on *T* and *S* sets.

16: **/* Scheduling problem transformation. */**

17: Apply maximum weighted bipartite matching (MWBM) to obtain optimal task scheduling of *T*.

18: **end while**

Figure 6.1 BGTSMR algorithm

6.3.2 Bipartite Graph Formation

To deal with scheduling high secure resources to tasks, we use the weighted bipartite graph to model appropriate allocation cases between tasks and slots as shown in Figure 6.2.

Definition 9. A weighted bipartite graph $G = ((T,S),E)$ is a graph whose nodes can be divided into two disjoin node sets (T,S), where $T = \{t_1, t_2, \ldots, t_m\}$ denotes the list of tasks within trust requirements, $S = \{s_1, s_2, \ldots, s_n\}$ denotes the list of slots within trust values, $E \subseteq T \times S$ the set of edges, and $W: E \rightarrow \mathbb{R}^+$ the weights of the edges.

The weight of the edge between a task $t \in T$ and a slot $s \in S$, $w(t,s)$, reflects the obtained trust value if the task is matched to the slot.

Definition 10. **Weight:** For each edge (t,s) in the graph, its weight $w(t,s)$ is basically defined as the sum of trust value when the task t at cloud i is run on the slot s of cloud j, $w(t,s) = TV_{d_k}^{l} + TV_{ij}^{js}$.

Definition 11. **Achieved trust:** For each edge (t,s) in the graph, when a task t at cloud i is run on the slot s of cloud j *or* $w(t,s)$ is selected, the achieved trust is defined as the trust value of slot s, $TV(t,s) = TV_{ij}^{js}$.

Definition 12. **Trust gain** is defined as the difference between the achieved trust of the proposed scheme and the Baseline when performing all tasks on cloud resources.

The map and reduce tasks of all submitted jobs are collected in the set task T. Based on those sets, slots with high trust value are selected. If a slot s is feasible, it must meet the following conditions:

The slot s of cloud j can be allocated for the execution for the task t at cloud i without violating the security constraints. In another word, its trust value is not less than the required trust value of t, $(TV_{d_k}^{l} \geq TV_{ij}^{js})$.

The slots are collected to maximize the trust value for the execution of task *t*. We assume that cloud *i* requests *m* slots $s_1, s_2, \ldots, s_m$ with trust values $TV_{i1}^{1s_1}, TV_{i2}^{2s_2}, \ldots, TV_{ii-1}^{i-1s_{i-1}}, TV_{ii+1}^{i+1s_{i+1}}, \ldots, TV_{im}^{ms_m}$ from C_1, C_2, …, C_{i-1}, $C_{i+1,..}$, C_m , respectively. These slots are considered as feasible slots for task *t* which performs on data d_k with sensitive level *l,* $\min(TV_{i1}^{1s_1}, TV_{i2}^{2s_2}, \ldots, TV_{ii-1}^{i-1s_{i-1}}, TV_{ii+1}^{i+1s_{i+1}}, \ldots, TV_{im}^{ms_m}) \geq TV_{d_k}^{l}$. The slot *s* of cloud *j*, which is assigned to *t,* has $TV_{ij}^{js} = \max(TV_{i1}^{1s_1}, TV_{i2}^{2s_2}, \ldots, TV_{ii-1}^{i-1s_{i-1}}, TV_{ii+1}^{i+1s_{i+1}}, \ldots, TV_{im}^{ms_m})$.

We assume that a node *n* is with the input of task *t*. *x and y* have available slots s_i and s_j, respectively to serve for task *t*. There are hop limits $h_1(n,x)$ and $h_2(n,y)$. If s_i and s_j have the same trust value, the slot assigned for *t* is owned by the node which has smaller hops.

Input: A set *J* of job with different security requirements and a set of slot *S* with heterogeneous trust levels.

Output: A weight bipartite graph *G=((T,S),E)*

1: **for** each job *j* of *T* **do**

2: **if** *j* is map phase **then**

3: $T \leftarrow T \cup$ (map task of job *j*)

4: **end if**

5: **if** *j* is reduce phase **then**

6: $T \leftarrow T \cup$ (reduce task of job *j*)

7: **end if**

8: **end for**

9: /* Forming a weighted bipartite graph. */

10: **for** each map task *m* of *T* **do**

11: **for** each slot *s* **do**

12: **if** *s* meets with the trust requirements of *m* **then**

13: $S \leftarrow S \cup s$

14: $w(m,s) \leftarrow$ the edge's weight is associated with the trust value of *m* on *s*.

15: $E \leftarrow E \cup (m,s)$

16: **end if**

17: **end for**

18: **end for**

19: /* Labeling the edge weights for solving slot contention. */

20: Initialize $G' = G$ /* Create a new graph G' to solve slot contention */

21: Fill(w',0); /* Assign 0 to all the edges' weight */

22: $min\_trust \leftarrow -\infty$

23: **for** each feasible slot s of S **do**

24: **for** each map edge (m,s) of s **do**

25: **for** each reduce edge (r,s) of s **do**

26: **if** $w(m,s) = w(r,s)$ **then**

27: Assign the edge trust of (m,s) by re-labelling $w'(m,s)$ as $w'(m,s) = min\_trust$

28: **end**

29: **end for**

30: **end for**

31: **end for**

Figure 6.2 The algorithm of bipartite graph formation

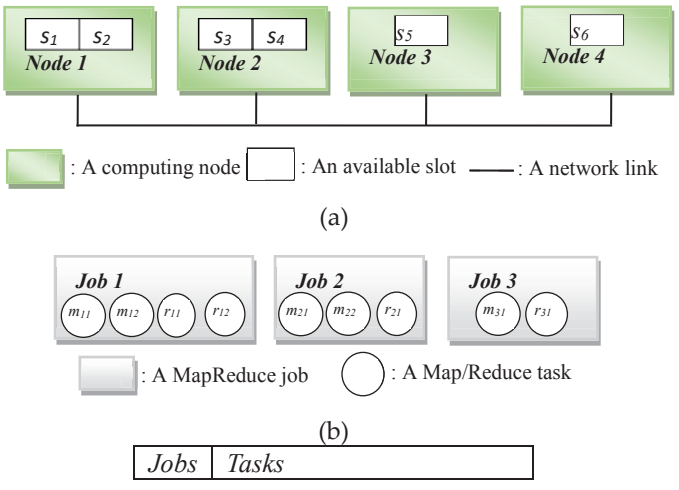
The BGTSMR is given in Figure 6.2. BGTSMR builds queue T and S to keep all executed tasks and available slots, respectively based on their trust values (line 1-8). Then, it forms the weighted bipartite graph (line 9-18). With the above graph modeling, the bipartite graph G represents all task scheduling cases between T and S. However, in the G, a slot s may be a feasible slot of multiple tasks as it satisfies the trust requirement level (line 12-15). If the task m_1 and m_2, having required trust value t_1 and t_2 ($t_1 > t_2$), respectively, will content the slot s, the slot s should be first allocated to m_1. BGTSMR uses a copy graph G' of G to solve slot contention (line 19-31). Suppose that m_1 is a map task, m_2 is a reduce task, and they have the same required trust value. m_2 will be allocated as reduce the task of a job is the last execution phase. If a reduce task is not allocated to its feasible slot, the completion time of the job may

exceed the job deadline. The proposed algorithm can re-label map trusts in *G'* after map phase (line 23-31). Based on the copy graph *G'*, the reduce tasks can obtain feasible slots. Finally, a job can still be completed with highest trust level while achieving better completion time. To assign higher priority to a reduce task when both the map task and the reduce task have the same required trust and request the same slot, the weights of map edges on the copy graph *G'* are re-labelled (see lines 27 in Figure 6.2).

6.3.3 A formation example

In this section, we describe the TSMR problem by modeling a weighted bipartite graph with a simple concrete example. A system model and a job set are given in Figure 6.4a. We assume that there are four nodes (VMs) and with different trusts and numbers of available slots in a cluster. These VMs are connected to a virtual network. As shown in Figure 6.4b, there are three jobs to be run on MapReduce program. Each job consists of several tasks to be performed. All tasks of three jobs are associated with a trust requirement and ready to be assigned to the available six slots of four VMs. Figure 6.4c presents the required trusts of tasks and provided trusts by slots.

Based on the Figure 6.2, a weighted bipartite graph is formulated as in Figure 6.3. In the bipartite graph, the tasks are connected with their feasible slots. For instance, the task m_{11} is connected to its feasible slots s_1, s_2, s_5, s_6. The corresponding weights are also assigned based on constraints. For each edge (t,s) in the graph, its weight is assigned to the obtained trust of the task t running on slot s following the second constraints as shown in Figure 6.3. To solve the slot contention, the weights of copy graph *G'* are relabeled by the minimum value, as shown in Figure 6.4e.



j_1	m_{11}	m_{12}	r_{11}	r_{12}
j_2	m_{21}	m_{22}	r_{21}	*N/A*
j3	m_{31}	*N/A*	r_{31}	*N/A*

Jobs	*Tasks*	*Trust requirements*
j_1	m_{11}	*7*
	m_{12}	*4*
	r_{11}	*8*
j_2	m_{21}	*6*
	r_{21}	*7*
j_3	m_{31}	*6*
	r_{31}	*8*

Slots	*Trust values*
s_1	*7*
s_2	*8*
s_3	*6*
s_4	*6*
s_5	*9*
s_6	*8*

(c)

Figure 6.3 An example of TSMR problem. (a) A small-scale system model. (b) The settings of a set of jobs in their trust requirements. (c) The trust requirements of tasks and provided trusts of slots.

6.3.4 Scheduling Problem Transformation

When the weighted bipartite graph has been formulated, the optimal solution of TSMR problem can be obtained by transforming the problem to the maximum weight bipartite graph problem.

Definition 13. Given a weighted bipartite graph, the maximum weight bipartite matching problem is to select a maximum number of non-adjacent edges, such that the total weight of selected edges is maximized.

Based on the weighted bipartite graph, if an edge (t,s) is selected, it represents that the task t executed on slot s. Non-adjacent edge selection finds edges which do not have the same task and slot nodes. In other words, it also means that one task will only be assigned to one slot, and one slot will be only used to run one map or reduce task at a time. By maximizing the number of non-adjacent edges, the trust satisfied tasks can be allocated as much as possible.

6.3.5 MWBM and Integer Linear Programming

To solve the MWBM problem, we formulate it as an Integer Linear Programming (ILP) problem (Dasgupta et al., Chen et al., 2015), i.e. an optimization problem in which the variables are restricted to integer values and the constraints and the objective function are linear as a function of these variables. Given a matching M, let its incidence vector x where $x(t,s)=1$ if $(t,s) \in M$ and 0 otherwise. One can formulate the maximum weight perfect matching problem as follows:

$Maximize \sum_{t \in T} \sum_{s \epsilon S} x(t,s)\, w(t,s)$ (6.8)

Subject to

$\sum_{t \in T} \sum_{s \in S} x(t,s) \geq \min(|T|, |S|)$ (6.9)

$\forall t \, \epsilon \, T \sum_{s \in S} x(t,s) \leq 1$ (6.10)

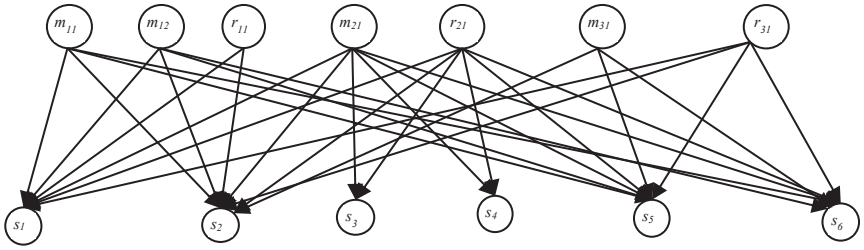
$\forall s \, \epsilon \, S \sum_{t \in T} x(t,s) \leq 1$ (6.11)

$\forall t \, \epsilon \, T \wedge \forall s \, \epsilon \, S, x(t,s) \in \{0,1\}$ (6.12)

In the above, the Eq. (6.8) is an objective function, which aims to maximize the total weight of selected edges. In the Eq. (6.8), $x(t,s)$ is a decision variable with *0* or *1* value, and represents the weight of the edge *(t,s)*. If the variable is equal to *1*, the edge *(t,s)* of the weighted bipartite graph will be selected. The weight of the selected edge will be added into the total weight. In the MWBM problem, the number of selected edges is based on the number of tasks and slots in the weighted bipartite graph. Thus, Eq. (6.9) represents the number of selected edges. Eq. (6.10) and Eq. (6.11) indicate that a task or a slot can only be selected at once. Eq. (6.12) represents the solution domain. The variable *x* can only be *1* or *0*. In the canonical form, there are $|T| \times |S|$ variables to be determined. When $|T|$ or $|S|$ is large, the ILP will take much computational time to obtain the optimal solution.

It can be seen that the optimal solution to this ILP is indeed a maximum matching in *G*. Therefore, an algorithm to solve ILP can be used to get a maximum matching in *G*. However, ILP is known to be NP-complete and hence there is no polynomial-time algorithm known for it.

We now describe how BGTSMR algorithm works by considering an example. First, we assume that there are three jobs with four map tasks and three reduce tasks, and VMs within slots $\{s_1, s_2, s_3, s_4, s_5, s_6\}$.



(a)

Slots Tasks	s_1	s_2	s_3	s_4	s_5	s_6
m_{11}	7	8			9	8
m_{12}	7	8			9	8
r_{11}	7	8			9	8
m_{21}	7	8	6	6	9	8
r_{21}	7	8			9	8
m_{31}		8			9	8
r_{31}		8			9	8

(b)

Slots	trust gain of map tasks *TV(m,s)*				trust gain of reduce tasks *TV(m,s)*		
s_1	$m_{11,s1}$	$m_{12,s1}$	$m_{21,s1}$		$r_{11,s1}$	$r_{21,s1}$	
	7	7	7		7	7	
s_2	$m_{11,s2}$	$m_{12,s2}$	$m_{21,s2}$	$m_{31,s2}$	$r_{11,s2}$	$r_{21,s2}$	$r_{31,s2}$
	8	*8*	*8*	*8*	*8*	*8*	*8*
s_3			$m_{21,s3}$			$r_{21,s3}$	
			6			*6*	
s_4			$m_{21,s4}$			$r_{21,s4}$	
			6			*6*	
s_5	$m_{11,s5}$	$m_{12,s5}$	$m_{21,s5}$	$m_{31,s5}$	$r_{11,s5}$	$r_{21,s5}$	$r_{31,s5}$
	8	*8*	*8*	*8*	*8*	*8*	*8*
s_6	$m_{11,s6}$	$m_{12,s6}$	$m_{21,s6}$	$m_{31,s6}$	$r_{11,s6}$	$r_{21,s6}$	$r_{31,s6}$
	8	*8*	*8*	*8*	*8*	*8*	*8*

(c)

Slots	Original map edge trust (*w*)				Original map edge trust (*w*)		
s_1	$m_{11,s1}$	$m_{12,s1}$	$m_{21,s1}$		$r_{11,s1}$	$r_{21,s1}$	
	14	11	16		15	14	
s_2	$m_{11,s2}$	$m_{12,s2}$	$m_{21,s2}$	$m_{31,s2}$	$r_{11,s2}$	$r_{21,s2}$	$r_{31,s2}$
	15	*12*	*14*	*14*	*16*	*15*	*16*
s_3			$m_{21,s3}$			$r_{21,s3}$	
			12			*13*	
s_4			$m_{21,s4}$			$r_{21,s4}$	
			12			*13*	
s_5	$m_{11,s5}$	$m_{12,s5}$	$m_{21,s5}$	$m_{31,s5}$	$r_{11,s5}$	$r_{21,s5}$	$r_{31,s5}$
	16	*12*	*15*	*15*	*17*	*16*	*15*
s_6	$m_{11,s6}$	$m_{12,s6}$	$m_{21,s6}$	$m_{31,s6}$	$r_{11,s6}$	$r_{21,s6}$	$r_{31,s6}$
	15	*12*	*14*	*14*	*16*	*15*	*16*

(d)

Slots	Original map edge trust (*w'*)				Original reduce labeled edge *(w')*			Re-label map edge trust		
s_1	$m_{11,s1}$	$m_{12,s1}$	$m_{21,s1}$		$r_{11,s1}$	$r_{21,s1}$		$m_{11,s1}$		
	0	*0*	*0*		*0*	*0*		*-1*		

s_2	m_{11,s_2}	m_{12,s_2}	m_{21,s_2}	m_{31,s_2}	r_{11,s_2}	r_{21,s_2}	r_{31,s_2}	m_{11,s_2}		
	0	0	0	0	0	0	0	-1		
s_3			m_{21,s_1}			r_{21,s_1}				
			0			0				
s_4			m_{21,s_1}			r_{21,s_1}				
			0			0				
s_5	m_{11,s_5}	m_{12,s_5}	m_{21,s_5}	m_{31,s_5}	r_{11,s_5}	r_{21,s_5}	r_{31,s_5}	m_{11,s_1}	m_{21,s_5}	m_{31,s_5}
	0	0	0	0	0	0	0	-1	-1	-1
s_6	m_{11,s_6}	m_{12,s_6}	m_{21,s_6}	m_{31,s_6}	r_{11,s_6}	r_{21,s_6}	r_{31,s_6}	m_{11,s_6}		
	0	0	0	0	0	0	0	-1		

(e)

Figure 6.4 The formulated weighted bipartite graph corresponding to the example. (a) the bipartite graph. (b) Original weights of the edges. (c) Trust gains when map and reduce tasks on slots. (d) Weight trust edges when map and reduce tasks on slots. (e) Re-labeled weights of map edges of copy graph G'.

6.3.6 An efficient heuristic algorithm for the maximum weight bipartite matching problem

The number of nodes in weighted graph *G* may be very large. In such a case, the computation time for solving the MWBM problem increases exponentially with the size of the graph. As discussed in the above section, tasks within required trusts and slots within trust values can be categorized into different "tiers" to provide better matching. Hence, we make two assumptions:

For a slot, the trust value associated with it is known during the trust transformation phase. For a task, the trust requirement is mapped into the trust space. With the trust inputs, the heuristic algorithm can find a feasible slot to satisfy the trust requirement of a map (reduce) task.

Within a computing node (*VM*), the slots share the same trust value granted from the secured transformation phase. This can reduce the number of slot nodes of a weight bipartite graph.

The proposed algorithm is presented in Figure 6.5. In the first place, nodes are grouped by task and slot to reduce the number of nodes in the weighted bipartite graph. Based on the number of different trust requirements, the task nodes can be re-allocated into task group nodes. Similarly, the slots in the same VM hold the same trust value. In addition, slots in different VMs can be also assigned the same trust value. According to the trust value, the slot nodes of the weighted bipartite graph can be classified into a number of slot group nodes.

Definition 14. Reduction weighted bipartite graph $RG = ((TG,SG),EG)$ is a graph whose nodes can be divided into two disjoint sets (TG,SG), where $TG = \{tg_1, tg_2, \ldots, tg_m\}$ denotes the list of

task group nodes within the same trust requirement, $SG = \{st_1, st_2, ..., st_n\}$ denotes the list of slot group nodes with the same trust value, $EG \subseteq TG \times SG$ the set of group edges, and $WG: EG \rightarrow \mathbb{R}^+$ the weights of the group edges.

Input: A weighted bipartite graph *G =((T,S),E)*.

Output: A set of selected edges *RSG*.

1: Obtain a reduced weighted bipartite graph *RG =((TG, SG),EG)* by apply node group.

2: Sort the edges in *EG* by their weights in decreasing order.

3: $RSG \leftarrow \emptyset$.

4: **while** $RSG \neq \emptyset$ **do**

5: Select an edge *(tg, st)* from *EG* with maximum weight.

6: $EG \leftarrow EG \setminus (tg, st)$.

7: **end while**

Figure 6.5 The heuristics algorithm for the maximum weight bipartite matching problem

Figure 6.6 presents the reduced weighted bipartite graph after grouping task nodes and slot nodes having the same trust value to weighted bipartite graph for task group nodes and slot group nodes. From Figure 6.6, we can see that the task nodes m_{11}, r_{21} are grouped into tg_1 due to the same trust value. The slot nodes s_2 and s_6 are combined into slot group node st_1 since they have the same trust value. In this manner, we can form a group nodes weighted graph as shown in Figure 6.6.

After grouping task nodes and slot nodes, the heuristic algorithm sorts the edges of the graph in decreasing order of their corresponding weights. Next, the algorithm runs in iterations. In each iteration, the edge with the lowest weight is selected. The corresponding task group and slot group on the selected edge are stored in a solid array. The selected edge is then removed from the graph. The algorithm is terminated when no edges are left in the graph.

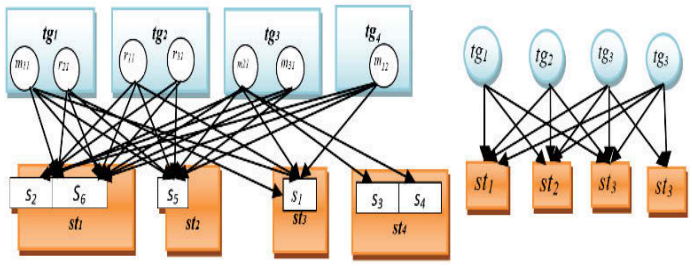


Figure 6.6 The reduced weighted bipartite graph

Based on the reduced weighted bipartite graph, the BWBM algorithm can perform with less computation time to find a solution. However, the solution needs to be processed further for resolving the task allocations since a node reduced weighted graph includes a number of the same tasks or slots. By finding the original tasks (slots) in a task (slot) group node, the task allocations can be finally completed. For example, we assume that the (tg_1, st_2) is an edge of the MWBM solution. The task group tg_1 includes tasks m_{11} and r_{21}, and the slot group st_2 consists of s_5. r_{11} and r_{31} are allocated to the slots s_2 and s_6, respectively.

From the above description, the heuristic algorithm includes three main components: graph reduction, edge sorting, and edge selection. In the graph reduction, it takes $O(|TG| \times |SG|)$ to classify the tasks (slots) into a number of task (slot) group nodes. Furthermore, the edge sorting of RG takes $O(|EG| log\ |EG|)$. In the edge selection, there are $|EG|$ iterations. Each iteration takes a constant time. In sum, the time complexity of the heuristic algorithm is $O(\max\{|TG| \times |SG|, |EG| log\ |EG|\})$.

6.4 The proposed scheduling framework for realization

This section briefly describes our proposed Trust-based scheduling framework for big data applications and its architecture in securing big data processing in the cloud environment. Figure 6.7 and Figure 6.8 illustrate the overall blueprint for securing data processing of applications and the big data processing from the perspective of users and programmers, respectively. The framework is structured into three components: the cloud service interface that provides data access and data process, the trust transformation component that maps cloud resources and users' sensitive data into trust values based on the trust metric, and the Trust-based scheduling component that provides optimal trust assignment resources for tasks.

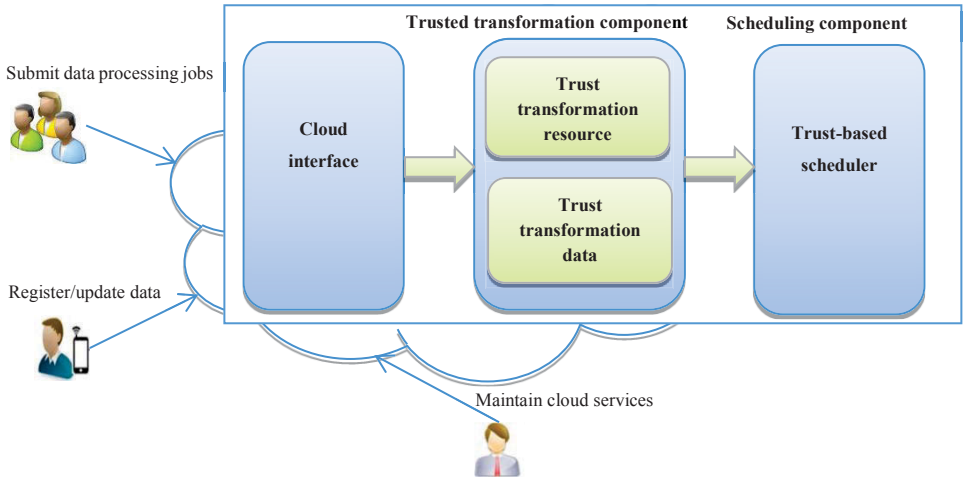


Figure 6.7 The design of Trust-based scheduling framework for big data applications

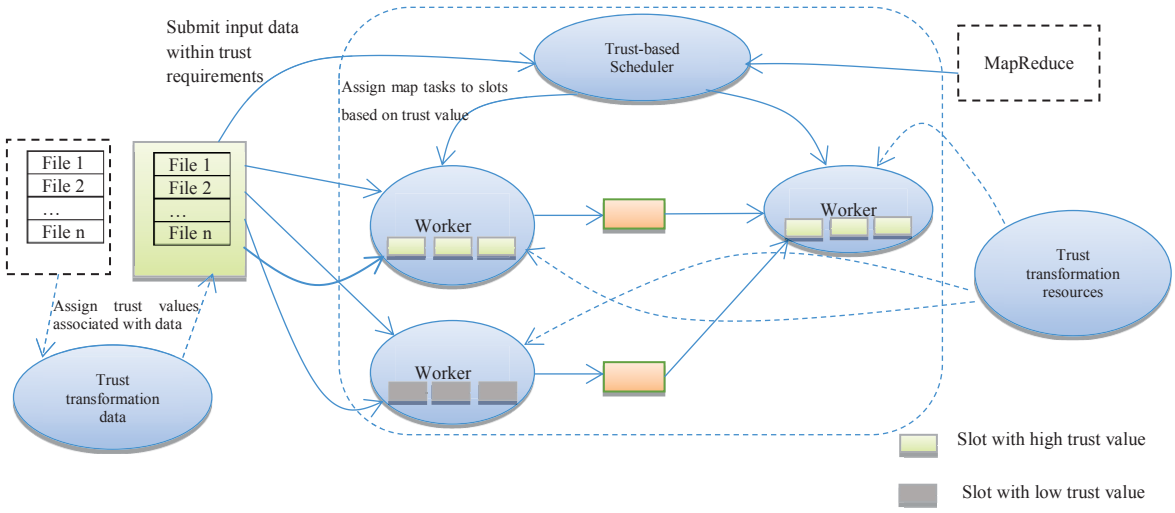


Figure 6.8 The Trust-based scheduler for big data applications from the perspective of users and programmers

Trust transformation component: The Trust transformation component provides two trust transforming services in the framework: the trust transformation resource service and the trust transformation data service. The trust transformation resource service assigns trust values to slots of nodes based on returned trust values in trust metric. A node involving in the big data processing is allocated a trust value, then the node transfers this value for its slots which are used as resources in scheduling for tasks within different trust requirement. The Trust-based scheduler can provide high trusted computational resources for tasks which perform on these data (i.e. high trusted slots from private cloud resources). Non-sensitive data is still processed the same as sensitive data based on its assigned trust value. Based on this strategy, the framework only adds overheads when matching between data with required trusts and resources

with provided trusts.

Trust-based scheduling component: It consists of the bipartite provisioning which forms tasks and available slots for the application and the Trust-aware scheduler which provides an optimal trust scheduling to assign slots for tasks.

Cloud interface: Cloud access interfaces provide data service interfaces to access or update data in cloud storage systems. It forwards requests with parameters to cloud security service to verify access permissions.

6.5 Simulation results

In this section, we evaluate and compare the performance of our proposed scheme to the baseline MapReduce on the aspect of execution time. In addition, the execution time and the trustworthiness of the proposed scheme are also investigated within different percentages of sensitive data to validate the convergence, accuracy, and resiliency of our proposed design. In order to show the advantage of the proposed scheme in term of achieved trusts, we introduce the competitor, called the baseline which assigns any feasible slots to tasks. The trust values of these slots may be equal or higher compared to the required trusts of tasks. The achieved trust value using this assignment for all tasks is called the Baseline trust (TV_B). The trust gain is defined as the difference between the required trust value and the achieved trust value.

6.5.1 Simulation Settings

The simulation is based on MapReduce simulation (MapReduceSimulator, 2016) and modified for our proposed scheme. The experiment programs are coded using the Java programming language. The simulation parameters are presented in Table 6.3.

Algorithms compared. We choose to evaluate the algorithms on trust gain and system performance aspects: 1) the baseline scheduling strategy to compare the achieved trusts and the trust gains; 2) the private cloud-based scheduling for users' sensitive data to compare the system performance in terms of job execution and resource utilization. For optimal trust, we adopt exhaustive search of all feasible matchings in each workload to obtain optimal results. However, due to the increase of workloads, the searching time increases exponentially. Hence, we only verify optimal results with our proposed algorithm in defined workloads.

Workloads. We investigate small-scale workloads and large-scale workloads to verify the

performance of the system based on trust-aware framework compared to the baseline solution. The workloads are presented in Table 6.4.

Trust initiations. We initialize the trust values in [*0,1*] to assign for Map and Reduce slots that involved in data processing. The required trust values of tasks associated with data sensitive levels are also provisioned in the range [*0,1*]. High trust values are set equal to or greater than *0.5* while low trust values are lower than *0.5*.

Computing the execution time of tasks. In the first experiment, we choose the sensitive s = 100% for the input data to compare the performance of the proposed scheme with the baseline mechanism when running on a different number of tasks.

Computing trust values. In the following experiments, we record both the trust values and performance when performing the proposed scheme within different percentages of sensitive data. We also compare the achieved trust values and trust gains between our proposed scheme and the baseline scheduling in responding to the trust requirements. The achieved trust values are also analyzed to identify the effects of trust components on these trust gains.

Table 6.3 Simulation parameters

Parameters	Values
int corePoolSize	12000
timerLimit	12000
blockSize	262144// unit is byte
execSpeed	102400
ioSpeed	51200
slotsPerNode	2
reduceStartPercentage	0.9
machinesPerRack	200
Racks	1000
Replica	3

Table 6.4 Workload for MapReduce jobs

Workloads	Map tasks	Reduce tasks
W1	16	28
W2	64	112
W3	128	224
W4	192	336

6.5.2 Results

6.5.2.1 The achieved trust values and the trust gains

We first tabulate the achieved trust values of both the proposed scheme and the baseline at the required trust of tasks under different data-sensitive levels varying from *10%* to *100%* (Figure

6.9). It can be seen that the achieved trust values of the proposed scheme remain unchanged at highest values compared to that of the baseline and present large differences between these achieved trust values. The results illustrate that the proposed scheme obtains very high trust values *0.8243239408* at *10%* data-sensitive levels while the Baseline acquires only a small marginal trust value (*0.2004775065*) compared to the required trust (*0.1088841364*) at the same data-sensitive level. As the data-sensitive level increases, higher trust requirements of tasks require the schedulers to allocate feasible slots with higher trust values, resulting in higher achieved trust values in both schemes. However, the Baseline still presents small differences between the required trust values and the achieved trust values, *0.3829328436* and *0.4630410187*, respectively at *10%* data-sensitive levels. Overall, the proposed scheme offers large differences of the achieved trust values (*0.934681* on average) in comparison with those of the baseline at different data sensitive levels. As a consequence, CSPs and users may derive benefits from our scheduling strategy in terms of profits and security satisfactions.

The achieved trust values of the proposed scheme and the baseline scheduling responding to trust requirements.

We next validate the proposed scheme with different sets of cloud resources. Specifically, the data-sensitive level and the workload are kept fixed while varying the percentage of high trust slots that involve in the process. We increase the h igh trust slots from *50%* to *100%* and set the required trust value to *0.5* for all tests. We run two types of workloads to evaluate the achieved trust from the proposed scheme and the baseline scheduling. The small-scale workloads consist of *20* tasks while the large-scale workloads include *100* tasks.

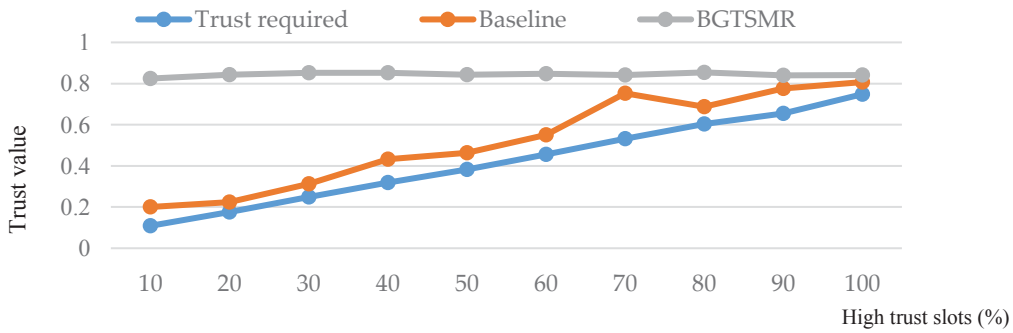


Figure 6.9 The achieved trust values of the proposed scheme and the baseline scheduling responding to trust requirements

Figure 6.10 and Figure 6.11 show the achieved trust values for the small-scale workloads and the large-scale workloads, respectively. Note that in the small-scale workloads no matter how

much the percentage of high trust slots is set, the achieved trust values of the proposed scheme increase slightly and reach very high trust values (*0.993817971* at *50%* high trust slots) given that the scheme has already reached very high trust values at initial percentages of the high trust slots.

Although achieved trust values of the baseline increase gradually, its achieved trust values are much lower than those of the proposed scheme (*0.657317595* for the former and *0.993817971* for the later, respectively). It can be seen that the Baseline only shows small differences between the required trust and the achieved trust values. For instance, with the small-scale workloads, it is only *0.050348305* at *50%* high trust slots and *0.134812219* at *100%* high trust slots. The big gaps, however, are obtained in the proposed scheme with *0.550348305* at *50%* high trust slots and *0.634812219* at *100%* high trust slots. Furthermore, as seen in these figures the performance of the proposed scheme is very close to the optimal performance that was obtained via exhaustive search.

Similarly, the same trend is repeated in large-scale workloads even though the achieved trust values of the proposed scheme are slightly lower than those in small-scale workloads since there are more tasks with high required trusts. In addition, achieved trust values of proposed scheme remain close to the optimal values. In summary, the advantages of our proposed scheme in providing highly secure resources for big data processing can be observed under any values of the achieved trusts due to its flexibility in exploring and selecting the computation resources.

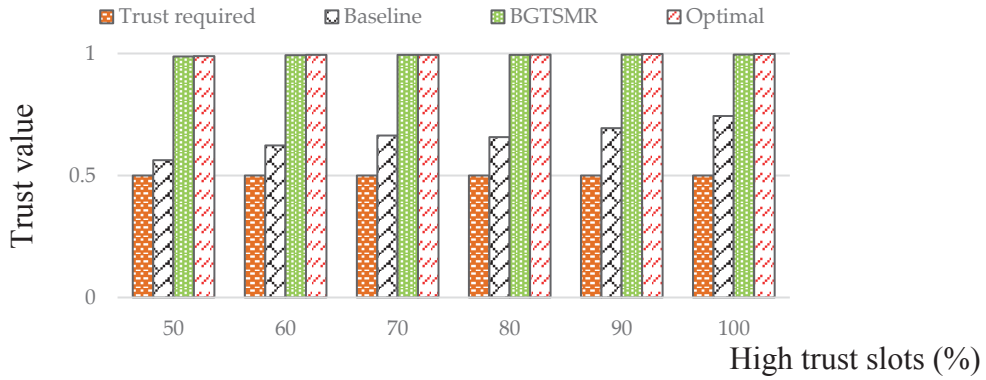


Figure 6.10 The achieved trust values of the proposed scheme and the baseline scheduling responding to high trust slots for the small-scale workloads

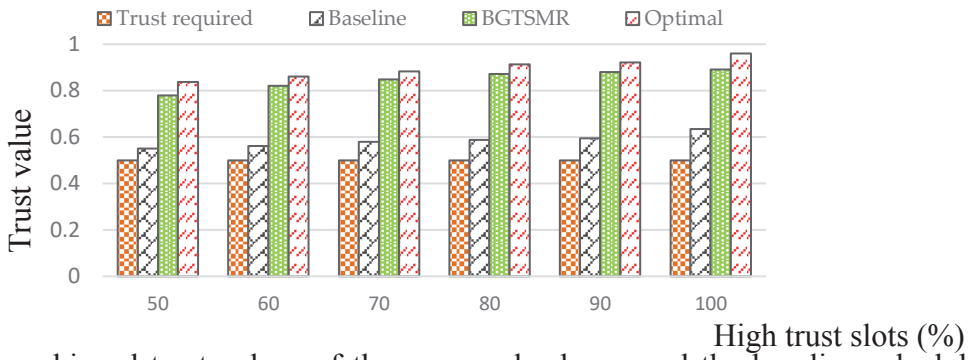


Figure 6.11 The achieved trust values of the proposed scheme and the baseline scheduling responding to high trust slots for the large-scale workloads

6.5.2.2 High trust resource allocation

We next compare the resources with high trust values that are assigned to perform tasks between the proposed scheme and the Baseline. The two different workloads and the sensitive's proportions of input data are used in experiments. We keep the same the percentage of high trust slots and low trust slots. Figure 6.12 shows the percentages of high trust slots for the small-scale workload of the Baseline (L1), the small-scale workload of the proposed scheme (L2), the large-scale workload of the Baseline (L3), and large-scale workload of the proposed scheme (L4). For the Baseline, when we increase data-sensitive levels for both workloads, the high trust slots are also increased in usages, resulting in the increase of achieved trust values presented in above experiments. The proposed scheme highly utilizes high trust resources to schedule for tasks as its highest priority. The results show that 100% of tasks in the small-scale workload are assigned high trust slots while 100% of high trust slots were employed to fulfill requests and the rest of tasks were performed by slow trust slots. Overall, the proposed scheme demonstrates better scheduling resource mechanism compared to the Baseline, resulting in higher achieved trust values and trust gains.

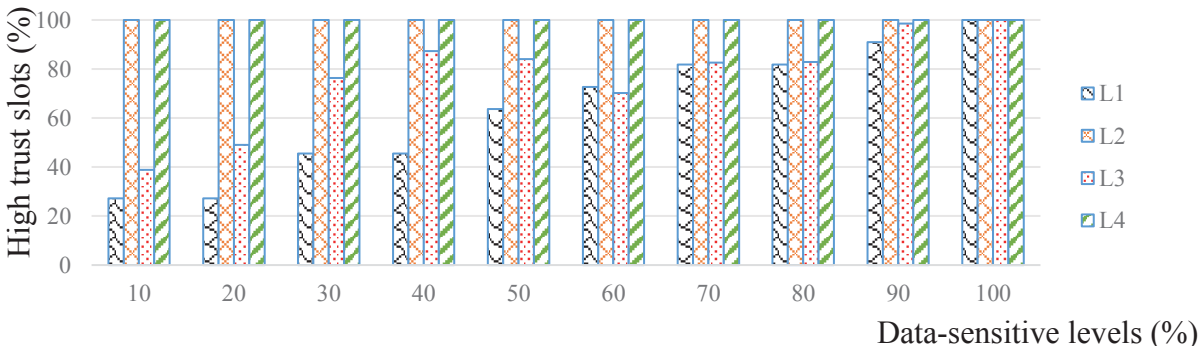


Figure 6.12 The percentage of number of tasks executed on high trust slots for the proposed

scheme and the baseline scheduling responding to high trust slots for the large-scale workloads

Table 6.5 Trust gains for the small-scale workloads

High trust slots (%)	Required Trust	Baseline	BGTSMR
50	0.5	0.063255415	0.488505817
60	0.5	0.123003563	0.493691802
70	0.5	0.16325847	0.494171342
80	0.5	0.156747525	0.49438056
90	0.5	0.193457412	0.495867673
100	0.5	0.244183182	0.496290634

Table 6.5 Trust gains for the large-scale workloads

High trust slots (%)	Required Trust	Baseline	BGTSMR
50	0.5	0.050348305	0.279495216
60	0.5	0.062278514	0.320889286
70	0.5	0.080073967	0.348674499
80	0.5	0.087995517	0.371585058
90	0.5	0.09533419	0.379898832
100	0.5	0.134812219	0.39096445

It can be seen from Table 6.5 that the trust gain of the proposed scheme (*0.493817971*) is approximately tripled compared to that of the baseline (*0.157317595*). This is because the baseline only searches for slots which are greater than the required trusts. A similar phenomenon can be observed in the large-scale workload in Table 6.6. Although the workloads increase, we can still have higher trust gains thanks to the optimal scheduling strategy of the proposed schemes. However, when workloads increase, more tasks need to be allocated high computation resources that lead to the decrease in the trust gains compared to the small-scale workloads. Overall, our proposed scheme can always find a balance scheduling strategy for appropriate task assignment to achieve better trust gains on any workloads.

6.5.2.3 The significance of trust gains from CSPs' perspective

Based on the trust requirements of tasks, feasible slots are listed out to perform these tasks. The trust values of these slots may be equal or higher compared to the required trusts of tasks. Hence, any feasible slot can be assigned to execute tasks without violating the SecLAs. Our proposed scheme searches for maximal trust values in scheduling tasks to slots. From the Eq. (6.8), the trust value can be related to three components of Eq. (6.6) as follows.

The total $TV = \max \sum w(t,s) = f(\tau, \delta, \theta)$ (6.13)

And the trust gain can be derived

$$TG = max(\sum w(t,s)) - feasible \sum w(t,s) = \Delta f(\tau, \delta, \theta) \quad (6.14)$$

where $\max(\sum w(t,s))$ is the achieved trust of the proposed scheme calculated as the total trust value of feasible slots with maximal trust and $\text{feasible} \sum w(t,s)$ is the achieved trust of the Baseline calculated as the total trust of feasible slots.

We next analyse the significance that trust gains provide based on trust components in Eq. (6.14). Clearly, the trust gains are significantly affected by trust ranking components including the ranking by the third-party systems (τ), how a cloud trusts another (δ) and how a cloud evaluates its resources (θ). If the trust gains lead to gains in business revenues for CSPs and customers in term of reputation, service consumption rates or profits, they are derived from the gains of these ranking components. We now analyze how the gains of trust components affects trust gains in the following cases: 1) the rankings are changed while trusts among clouds and resources' trusts are kept at fixed values; 2) trusts among clouds are changed while rankings and resources' trusts are kept fixed at values; 3) resources' trusts are changed while trust among clouds and rankings are kept at fixed values.

For the first case, the gains are contributed by the ranking of the third-party systems. To see the effect of the trust gain by the third-party ranking, we keep δ and θ fixed to mean that CPSs trust each other at the same trust level and a CSP trusts its resources at the same level. Thus, the more CSPs with high ranking are involved in the process, the higher gains are achieved from the scheduling strategy. From the Eq. (6.14), the trust gain for both schemes can be derived as follows.

$$TG = \Delta f(\tau, \delta, \theta) = \Delta f(\tau)|_{\delta,\theta} \begin{cases} TG = 0 \; when \; \tau = \tau^0 \\ TG > 0 \; when \; \tau = \tau^1 \end{cases} \quad (6.15)$$

where τ^0 is the gain of trust values based on ranking by the third-party systems for the Baseline, τ^1 is the gain of trust values based on ranking by the third-party system for the proposed scheme.

From Eq. (6.14), we have

$$\tau^1 > \tau^0 \Rightarrow r^1 > r^0 \Rightarrow higher \; ranking \; clouds \; involved$$

A gain in the third-party ranking is exactly the objective of a cloud provider as this implies gains in its benefits in term of revenues and resource utilizations. This is the basis for the SPECS

ranking system framework. CSPs with higher ranking offer their customers a highly secure data processing services with the rich pool of resources as per Security SLA. As presented in (Taha et al., 2014, Luna et al., 2015), the SecLAs determine the overall security level according to the customer's requirements or SLAs between CSPs and customers, resulting in the costs covered by customers, the more resources are consumed, the higher revenues represented by service subscriptions that CSPs can achieve. Furthermore, a CSP has more options in selecting on-demand trust resources from either ranked or trusted CSPs instead of limited private clouds. Consequently, this improves significantly the system performance when requests are huge and skewed.

Suppose that there are a set of trust gains TG based on ranking by the third-party systems and a set of gains G representing for profits achieved by CSPs. Thus, there exists a one-one mapping between TG and G as $f': TG \rightarrow G$. The trust value assigning to cloud resources of cloud C_j is calculated as follows.

$$g = f'(tg) \qquad (6.16)$$

where f' is a function that maps a trust gain *tg* based on ranking by the third-party systems to a gain $g \in G$ achieved by CSPs.

Similarly, we keep τ and θ fixed in the second case; that means CSPs are ranked the same ranking and a CSP trusts its resources at the same level. Hence, the gains are derived as $\Delta f(\delta)$. For the third case, we keep τ and δ fixed; that means CPSs are ranked the same ranking and CSPs trust each other at the same trust level. Hence, the gains are derived as $\Delta f(\theta)$.

6.5.2.4 The processing time and resource ultilization

We carry out three experiments with a different number of workloads and the sensitive's proportions of input data to compare the performances between our proposed scheme and the Baseline. In the first two experiments, we compare the performance of the proposed scheme and the baseline when running on the small-scale workloads and large-scale workloads. We keep the same required trust and the percentage of high trust slots. In the next experiment, we also verify the performance of the proposed scheme with different data-sensitive levels (*S*) when running on the large-scale workloads.

Comparison of running time

In the first simulation, we run on the small workloads ranging from *10* map tasks and *10* reduce

tasks (*M10R10*) to *20* map tasks and *20* reduce tasks (*M20R20*). Figure 6.13 shows the execution time of the proposed scheme and the Baseline when running on a different number of map and reduce tasks. With the same number of tasks, the proposed scheme introduces slight overheads in comparison with the Baseline. In average, the execution time of our proposed scheme *T1* (*11502ms)* is slightly higher than that of the Baseline *T2* (*10464 ms*). This is because the matching between tasks and slots needs to be carried out as a part of the scheduling process of the proposed scheme before assigning slots to tasks, leading to an additional delay of the whole process. Although the proposed scheme requires intensive scheduling of high trusted slots to tasks for the sensitive data in order to achieve high trust values, the slight overheads may be considered negligible when security consideration is taken into account.

In the second simulation, we run the large-scale workloads to verify the overhead of our proposed scheme compared to the original mechanism. Figure 6.14 shows the running time of both schemes. When the workloads increase, the results, here, are consistent with the other results in the previous experiment. Specifically, *T1* (*55997ms*) is still slightly higher than *T2* (*54106ms*).

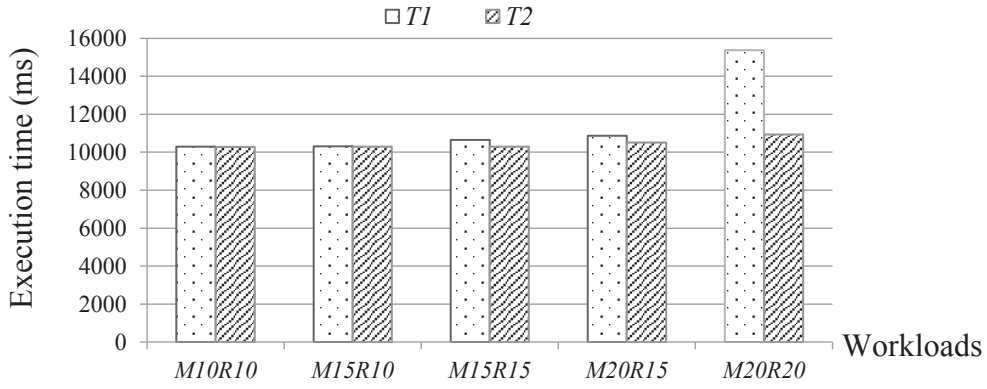


Figure 6.13 Performance Comparison between Trust-based scheduling and the Baseline running on different workloads

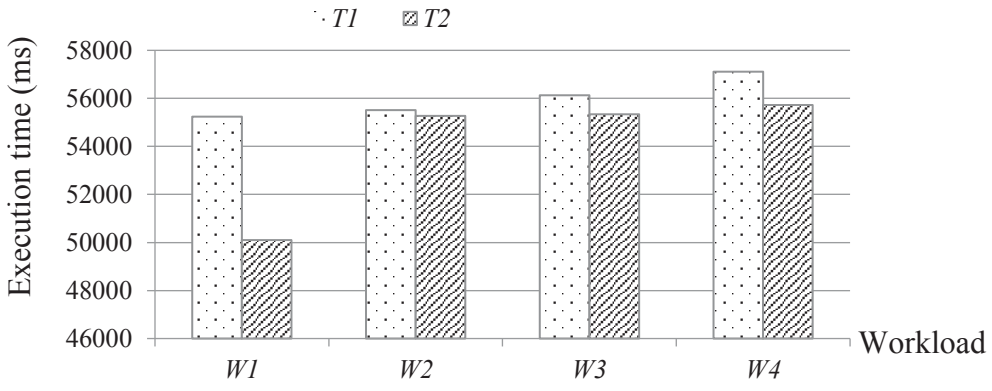


Figure 6.14 Performance Comparison between Trust-based scheduling and the Baseline running on different workloads

Figure 6.15 shows the running times of the proposed scheme on different large-scale workloads and data-sensitive levels. We see that apparently when workloads increase from *W1* to *W2*, *T1* increases rapidly. After that, it continues increasing but with much lower rates. Looking into the last three workloads, *T1* remains slightly higher for all percentages of sensitive data. When *S* is set from *1%* to *50%*, *T1* reaches approximately the same value around *54190ms*.

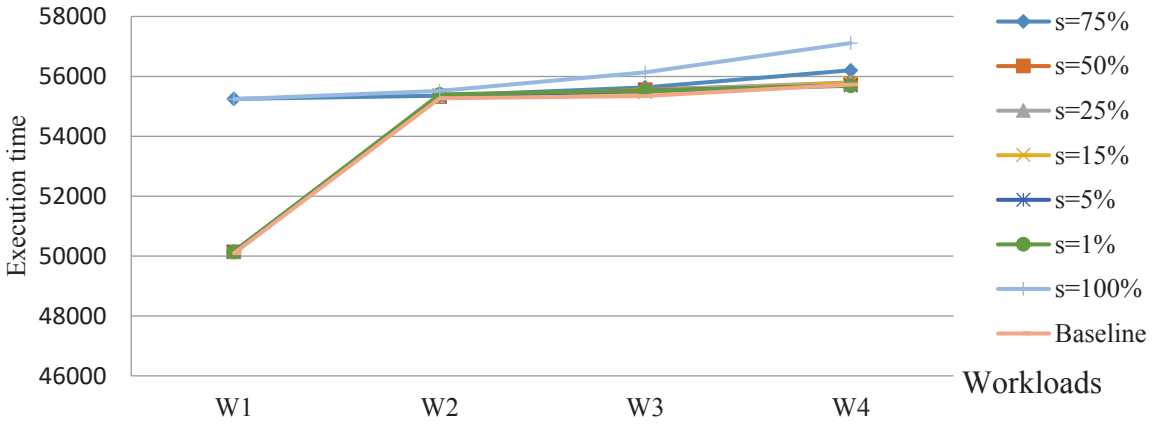


Figure 6.15 Performance Comparison for Different Sensitivity Ratios

Comparison of resource ultilization

In order to measure the resource usage of the proposed system compared to the systems that process sensitive data at private cloud only, we set simulation parameters as follows: slotsPerNode =2, machinesPerRack =10, and racks =10.

The ratios of sensitive data are set equal to *100%*. *30%* of resources are owned by private clouds and *70%* by public clouds. In our proposed scheme, the number of nodes that have high trusted slots is set as *70%* and *30%* of nodes are low trusted resources. Figure 6.16 shows the resource usage of the two systems. Since the systems that process sensitive data at private cloud which have a limited amount of resources, the overheads increase significantly compared to the proposed scheme. It indicates that a large number of available resources at public cloud are not used in data processing. Thus, nodes at our proposed system do not experience heavy workloads compared to those of private cloud only.

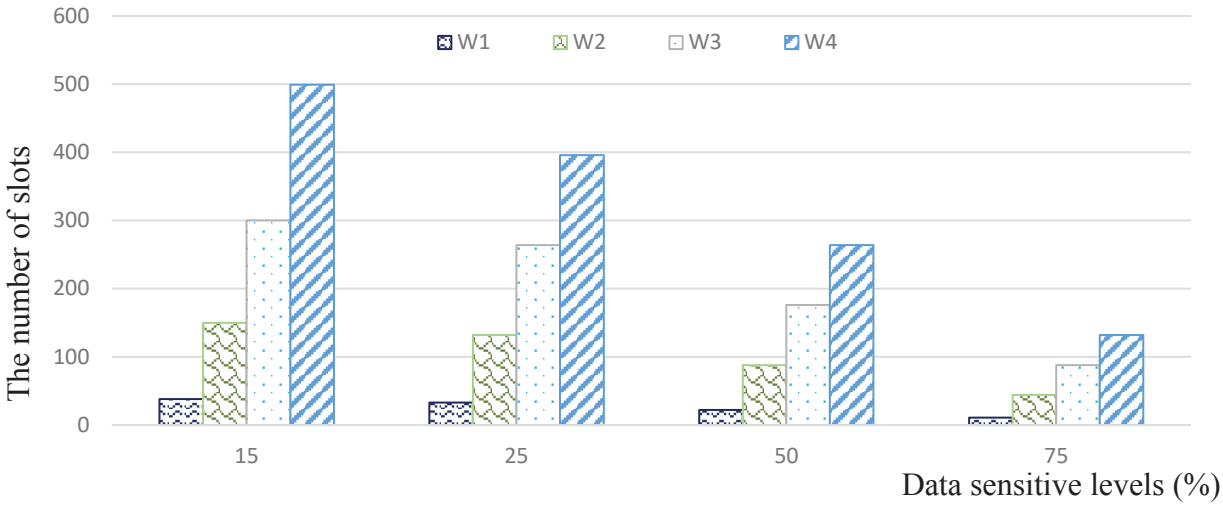


Figure 6.16 Low trust slots ultilization in the Trust-based scheduling within different data-sensitive levels

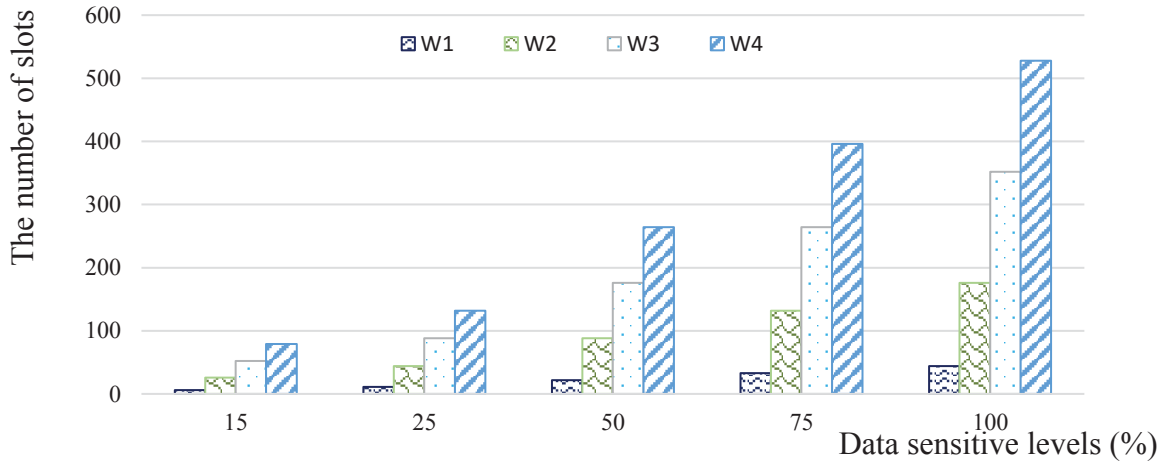


Figure 6.17 High trust slots ultilization in the Trust-based scheduling within different data-sensitive levels

It can be seen in Figure 6.16 that within small-scale workloads when data-sensitive levels increase, the low trust slot numbers decrease for both workloads. This is because requests require more high trust resources to schedule for tasks. However, it shows the opposite trend in Figure 6.17. When the data-sensitive levels increase, more high trust slots are employed. It can be concluded that our proposed scheme utilizes highly resources on demands to achieve highest trust values and trust gains in executing tasks.

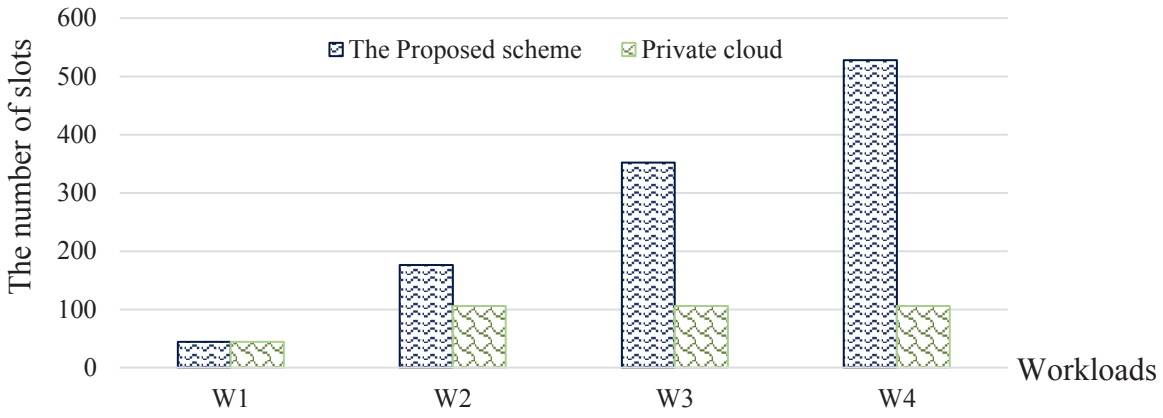


Figure 6.18 Comparison of resource ultilization between the Trust-based scheduling and the private cloud

Figure 6.18 shows the comparison between the proposed scheme that does not differentiate public and private clouds provided their resources satisfy the required trusted levels and the scheme that utilized private clouds only for processing sensitive data. The private clouds with limited high trust slots do not have enough resources to serve for tasks when the workload increases. Thus, tasks have to wait for feasible slots to be executed, leading to high latency of the process. Nevertheless, due to the availability of high trust slots enabled by our scheme the proposed scheme experiences little delay and the task completion time was shortened as a consequence.

To summarize, although the system performance of our proposed scheme marginally surpasses state-of-the-art methods, with our scheme the system achieves significantly higher processing security levels.

6.6 Summary

This chapter introduces and discusses the Trusted-based resource scheduling scheme for big data applications that involve users' sensitive data concerns at the cloud. We propose a trust-aware framework that enables CSPs to high trusted resources for big data applications when processing sensitive data. We introduced a trust metric that enables a CSP to provision relevant trust resources from other CSPs for processing users' data. We also formulated the trust-based scheduling problem as the weight bipartite graph and introduced an optimal algorithm that schedules tasks for high-security requirements while improving resource consumption rates. We performed simulations on MapReduce framework with trust-based scheduling to determine the achieved trust, trust gains and resource utilization. We analyzed

the implication of the actual trust gains from the proposed trust optimizing scheduling scheme. The results showed that our proposed scheme benefits CPSs and customers, enabling low costs data processing but achieving high gains in security. With the increase of applications with low sensitive-response requirements, Cloud computing does not handle well the local performance issues. Thus, in the next chapter, we will discuss data processing model for Fog computing in handling both security and local performance issues efficiently.

CHAPTER 7 A DATA PROTECTION MODEL AND A TASK SCHEDULING SCHEME FOR FOG COMPUTING

Cloud computing has established itself as an alternative IT infrastructure and service model. However, as with all logically centralized resource and service provisioning infrastructures, cloud does not handle well local issues involving a large number of networked elements and it is not responsive enough for many applications that require immediate attention of a local controller. Fog computing preserves many benefits of cloud computing and it is also in a good position to address these local and performance issues because its resources and specific services are virtualized and located at the edge of the customer premise. However, data security and resource management have become critical issues in Fog computing due to the complex distribution, high mobility of fog devices, and high frequent data movements.

This chapter introduces a data protection model for Fog computing and a resource scheduling scheme between Fog computing and cloud computing. The chapter introduces a concept of "Region" for Fog computing. It handles requests and provides resources for nearby users. The data protection model provides region-based trust establishment, MS, and Fog Privacy Role Based Access Control. It enables fog devices in different regions to share and access resources in a secured manner. It also enables clients to keep track of changes of data location among regions periodically by using a LRD as well as an enhanced verification procedure at FPRBAC. The chapter also introduces FBRC in which requests are locally handled not just by a region but multiple regions when additional resources are needed. The proposed scheme considers a scenario where computation can be processed either at local regions and/or remote cloud servers. By obtaining efficient task schedules at both regions and cloud servers, it can choose the appropriate computational resources to minimize the computation and transmission latency of all requests.

The structure of this chapter is as follow: Section 7.1 presents the overall system architecture

and Fog-based Region scenario. Section 7.2 presents the data protection model for Fog computing. Section 7.3 comprehensively describes task scheduling in FBRC. Section 7.4 provides the simulation setup and Section 7.5 presents the evaluation of the proposed schemes. Finally, Section 7.6 summarizes this chapter.

7.1 Fog-based Region

Fog computing has become a new computing model in providing local computing resources and storage for end-users rather than cloud computing. It provides a popular platform to facilitate a wide range of applications such as smart transport, healthcare, smart grid applications. Users' applications and data, however, are increasing not only in number, volume, and variety but also in complexity with strict latency requirements. As a large number of physical devices move about in a large area, processing tasks may experience high latencies and jitters as needed computing resources may not be optimally distributed and are located far from their users. This section introduces a local computing concept called "Region" to deal with these issues, the logic view of FBRC, and the Fog-based Region scenario.

7.1.1 Definition of Region

A Region centres on a physical location which provides services for users within its coverage. It includes all fog devices such as high-end servers, smartphones, and vehicles connected to one another via wired or wireless connections in a defined geographic location. Some fog devices, which may share computing resources in multiple regions, may move to a new region but still request and/or provide computing services.

7.1.2 System model

The logical view of FBRC is shown in Figure 7.1, where there is a set R of fog-based regions, a set C of cloud servers. All these nodes are inter-connected. Clients at the original region can request computation resources from other regions placed around it or cloud servers. Thus, tasks can be scheduled to minimize the completion time either at regions or cloud servers. In each region, a fog node is selected to manage the region. This fog node handles tasks submitted in the region such as join/leave requests, receiving and submitting computation requests, scheduling tasks to appropriate computation nodes. Hence, these fog nodes also collect and update throughputs and resources availability of closest regions and cloud servers. In our model, regions, and cloud servers are required to periodically notify each other of the average

throughput and the amount of data that they can sustain.

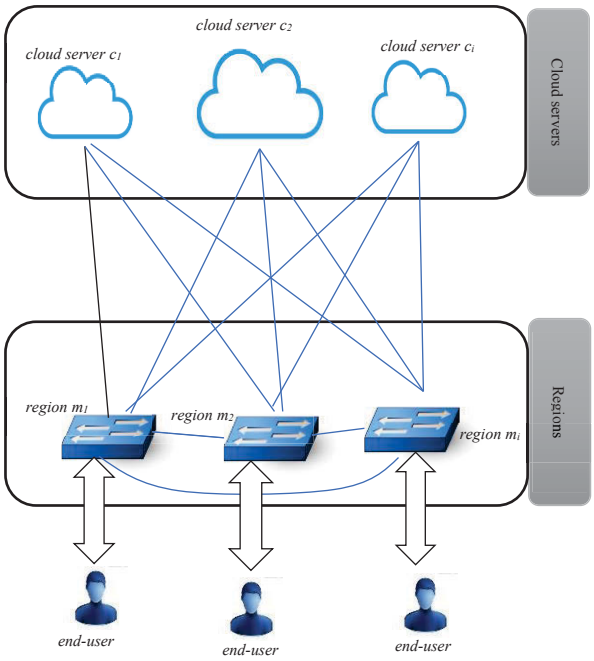


Figure 7.1 The Logic view of FBRC

7.1.3 Fog-based Region scenario

Figure 7.2 presents a fog-based region scenario. In this design, a region can be structured by one or several fog nodes. A fog node consists of several fog devices with weak performance which are deployed at an edge network. It can provide computation, network resources and storages. The fog devices are heterogeneous ranging from high-end servers to end devices such as mobile devices, wearable devices. For example, Region 1 is structured by fog node 1 and fog node 2 while Region 2 is formed by fog node 3 and fog node 4.

Fog election: It is essential to delegate a fog node to manage a region's activities and computing resources due to frequent join and leave node requests. Furthermore, task executions with a region need to be secure to protect sensitive data. We use a decentralized method (Liu et al., 2015) to select the delegated fog node in the region. Each fog node sends its vote for other fog nodes. In turn, it receives votes from other fog nodes. Thus, the votes in the region are collected into high capacity nodes among which the delegated node may be selected. A heartbeat is sent by every fog node to other fog nodes in a region periodically, at a heartbeat interval. Heartbeats are used by a fog node as a means to inform all fog nodes it is alive. A delegated fog also sends a message to all fog nodes in its region every time the region changes by the detection of an

event, which is either a new fog that has entered the region or one that has left or crashed.

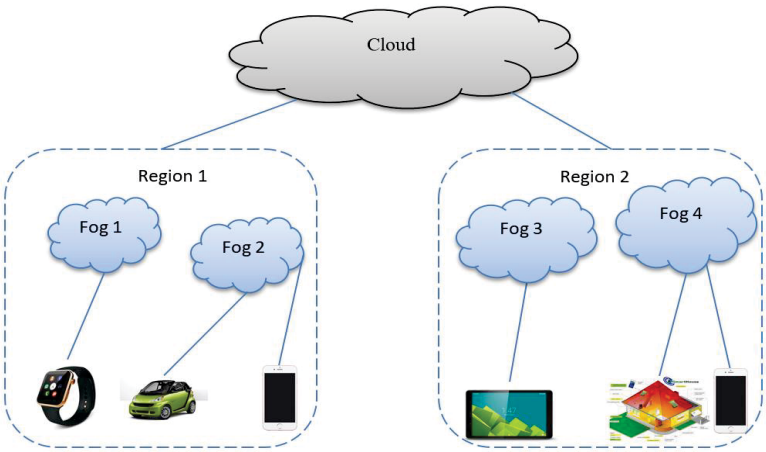


Figure 7.2 The overview of RBTA scenario

7.2 Adversarial model

7.2.1 Adversarial model assumptions

We present our assumptions regarding fog-based region data protection in the presence of an adversary as follows: 1) *fog devices integrity:* technical and non-technical approaches to prevent such fog devices being tampered with have been taken to prevent the issues of device tampering to regions; 2) *physical security:* fog devices are owned by both users and service providers. These devices can be physically observed, enforced and verified through known best practice on duty management by organizations. This assumption is important for building high-level hardware and software security guarantees for the components of fog-based region infrastructure; 3) *Cryptographic security:* we assume symmetric and public-key encryption schemes are semantically secure and that an adversary cannot obtain plain text of encrypted data when it is sent and received by fog devices, and that the message authentication code algorithm correctly verifies message integrity and authenticity; *4) defined policies:* defined policies correctly authorize valid requests associated with users, fog nodes, fog devices, and operations. The adversary cannot modify defined policies to grant and bypass FBRBAC by their own permissions.

7.2.2 Adversary Capabilities

In this section, we describe specific capability for adversaries, denoted by *ADV*. We adopt the Yao-Dolev (Dolev and Yao, 1981) threat model in which, an *ADV* can overhear and generate any requests to regions and is only limited by the constraints of cryptographic mechanisms.

We define two complementary adversarial types. One acts as a user, denoted by *ADV A*, to compromise the confidentiality and integrity of data in a region. The other type acts as a fog device, denoted by *ADV B*, has capabilities to send requests and receive responses over regions. They are able to perform the following actions: 1) Send a valid request with arbitrary roles and operations to regions it can reach; 2) Attempt to impersonate other fog devices; 3) Issue arbitrary policies within its region; 4) Use the cryptographic material to decrypt network traffic that is sent and received by other fog devices.

7.2.3 Attack vectors

From the adversary model, we identify potential attacks relevant to Fog computing from fog-based regions perspective as follow: 1) *Direct access and intrusion attack*: the adversary bypasses the trust of regions and generates requests with different parameters to FPRBAC; 2) *Man-in-the-Middle Attack*: the adversary may compromise fog nodes and replace them with fake ones. It then intercepts the communication among other fog nodes, and attempts to make them believe that they are communicating with each other over a correct and safe connection; 3) *Attack on policies and roles in FPRBAC*: the adversary may issue malicious policies that overwrite or disable legitimate policies already in place.

7.3 The proposed architecture

In this chapter, we introduce a data protection model which allows users to access secure resources based on assigned roles. The new features of our proposed model include a region-based trust component to deal with newly joined devices coming from other regions; FPRBAC for verification and authorization; Mobility management to deal with changes of fog devices' location, tracing, tracking, and triggering an alarm on any operation, data or policy violations. Figure 7.3 depicts the model and its three core components: 1) the RBTA component; 2) Fog-Base Privacy-Aware Role Based Access Control; 3) the mobility management component. These components will be described in the next subsections.

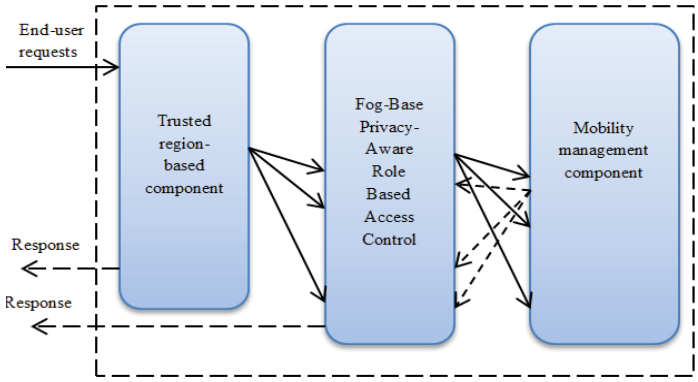


Figure 7.3 The design of data protection framework for Fog computing

7.3.1 RBTA for fog nodes

Fog election: It is essential to have a fog node which delegates the management of computing resources and task executions of a region. A region has to deal with not only high frequencies of join and leave requests but also with sensitive data protection. We use a decentralized method (Liu et al., 2015) to select the delegated fog node in the region. Each fog node sends its vote among other fog nodes and its received votes to them, so that the votes in the region finally are collected into high capacity nodes. A *heartbeat* is sent by every fog node to other fog nodes in a region periodically, at a *heartbeat interval*. Heartbeats are used by a fog node as a means to inform all fog nodes it is alive.

7.3.2 Trust establishment between two regions

Figure 7.4 presents a trust relationship establishment scenario. When a region receives a request to establish the trust relationship with a new region, it will verify and analyze the request to obtain the destination address. A general procedure has to be executed between the two regions as request and response messages of two delegated fog nodes. The procedure is as follows: 1) the new region sends a request to the original region for establishing trust between them; 2) the original region analyses the request as to whether it can or cannot establish the trust and replies by a message; 3) Once two regions have accepted the trust establishment, the trust relationship is updated at databases of these regions.

Two delegated fog nodes from each region carry out the Challenge/Response process (Nguyen et al., 2016). Hence, Trust evaluation is performed to obtain the trust value. The

delegated fog node in the requested region tests all requests from other regions and evaluates the trust degree on each region before making a decision whether to accept it and establish a trust relationship between two regions. Specifically, it raises a number of questions or commands which are performed by the delegated fog in another region. Based on the number of correct answers, the requested region decides whether the trust relationship is or is not established.

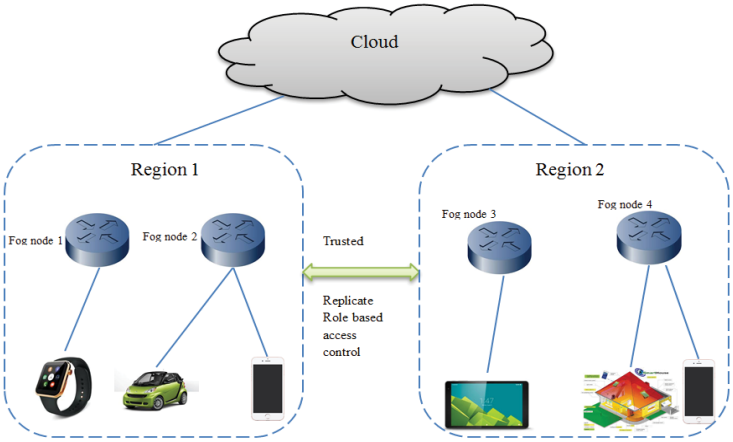


Figure 7.4 Trust establishment between two regions

7.3.3 Join/leave a region

A fog with high mobility requirements switches more frequently among regions. Hence, it needs to have its roles verified in order to use computing resources in new regions. The join operations of fog devices are similar to a mobile device in the mobile network (Jae-Woo, 2007). When the delegated fog node receives a join request from fog devices, it will verify and analyse the request to obtain the Fog_ID and the region. Thus, the trust relationship can be verified correctly between the current region and the home region of fog devices based on this information. After the join process completes, LRD will be updated at both regions. When a fog device wants to leave the region, it notifies the fog nodes to update LRD. Sometimes, fog devices will leave the system without notice since its connections are lost. To handle abrupt leaving, fog nodes scan device status and update LRD periodically.

7.3.4 Mobility management component

Fog devices are highly dynamic and frequently switch among regions. It is vital to provide a

MS at regions handling location requests such as update and query. The mobility management component includes the MS and the LRD.

The MS is responsible for creating queries submitted by fog nodes to the LRD. When a fog device is granted permissions in a new region, the MS updates the information about the fog in the LRD. In addition, it also supports verifying a new fog participating in the region.

The LRD provides both a user and fog devices register when there is a change of location from one region to another. It stores information related to Fog_ID, Fog_location, Operations, and Timestamp for monitoring purposes. Fog_ID represents entities such as users, fog nodes, and fog devices. When a Fog_ID is generated for a request subscribing to a new region, the registration procedure is performed that involves a binding location service for the data welfare including monitoring and raising the alarm. Therefore, whenever a fog device moves out of its home region, a request is submitted to the MS to extract information from the database necessary to perform registration. In addition, LRD is also served for location analysis, decision and exchange information.

7.3.5 Fog-based Privacy-aware Role Based Access Control

Role-based access control (RBAC) model (Sandhu et al., 1996) lacks context information to satisfy sophisticated scenarios. To achieve a light data protection scheme with sensitive-latency requirements, we introduce new components: *Region (Rg), Conditions (Co)*, *Obligations (Ob), Location (L)*, *Operation (Op), Purposes* and *Location (L)* to adjust policy description for the distribution of fog node concerning authorization delegation, cross-realm role assignment and privacy-aware scheme beside *Subject (S), Object (O), Role (R)*.

In the model, *Subject (S)* is an entity which accesses relevant Object. It can be a user, a fog node or a fog device, and its attribute is used to determine a specific role. *Object (O)* represents any computing resources or data relating to the identified *S*, such as fog devices or smart devices. *Role (R)* is a functional entity associated with specific authority and responsibility within a region. For instance, in Smart Traffic Lights application, Connected Vehicle can request and receive an optimum route to its destination based on the estimated time of arrival but Emergency Connected Vehicle can access and update a new route for other vehicles based on its current location. *Operation (Op)* binds *O* and consists of a set of actions that a subject can execute (such as read or write privileges). *Region (Rg)* is a domain identifier that defines *R*. In Fog computing, the set of *Rgs* and *Rs* defines distributed role allocation. *Condition (Co)* is a

prerequisite to be met before any *Op* can be executed; for example, a school bus's route can be only disclosed to another Connected Vehicle when the current time is between 9 AM to 3 PM. *Purpose (Pu)* specifies the intended reason of the *Op*. Emergency Connected Vehicle can access the route of a Connected Vehicle only when the purpose is to respond to a reported emergency. *Location (L)* is a function which must be executed before an *Op* is executed on *O* or after the execution.

We proposed the Fog-Based Region verification algorithm for the proposed model to verify requests at a region. A request is checked if it is trusted or needs a trust establishment (line 2-4). In line 3, FPRBAC is executed to verify roles in requests. The MS is performed to update LRD and notify data owners (line 8-11).

Algorithm 1. Fog-Based Region verification algorithm

Input: A set *RQ* of requests to access the data with parameters.

Output: granted or denied requests

1: **for** each request *r* in *RQ* **do**

2: **if** *r.region_id* exists in Trust database **then**

3: *r.trust = true* // Request is trusted and passed to FPRBAC

4: **else**

5: *trust_establishment*(*r*) // Establish a trust relationship

6: **end if**

7: *result* = *verify_FPRBAC*(*r*) // verify access roles in the request

8: **if** *result = true* **then**

9: *MS_Update_LRD*(*r*) // Update LRD and notify data owner

10: e**lse**

11: *MS_Notify_User*(*r*) // The MS notifies data owner

12: **end if**

13: **end for**

7.3.6 Use case scenarios: **Healthcare applications**

In this section, we introduce our use-case scenario to demonstrate the proposed model in supporting HIS. EHRs are integrated into the HIS which allow patients to manage and control their health records through the internet. Patients can access their health data and share the data with physicians, health care providers, insurance practitioners, researchers and family members. We demonstrate the FPRBAC in protecting data.

A patient named Alice is recently diagnosed with gastrectomy cancer. Surgical removal of the stomach (gastrectomy) is the only curative treatment. Alice is assigned to a general practitioner, named Bob from the hospital (e.g., hospital A, located as Region A), which he regularly visits. Bob is granted full permissions to access Alice' data including read and write operations. Under Alice's health conditions, he needs to seek additional consultation regarding his treatment from different hospitals (e.g., hospital B). Alex is assigned as an invited practitioner. He is required to analyze solving Alice' medical case and suggest a possible solution to Bob. Specifically, Alex can read the patient's EHRs for treatment purposes only if the current time is from 9 AM to 6 PM. However, the patient and Bob will be informed by email. The above-mentioned scenario requires the FPRBAC's verification for any data operations. Sample access roles are presented in Table 7.1.

Table 7.1 Sample access roles

Subject	Role	Object	Operation	Permission
Alice	Patient	Alice'personal info, Alice' old medical records, and Alice's summary treatment report	Read	Grant
Bob	General practitioner	Alice'personal info, Alice' old medical records, Alice' private Notes, and Alice's summary treatment report	Read/write	Grant
Alex	Invited practitioner	Alice'personal info, Alice' old medical records, and Alice's summary treatment report	Read	Grant

7.4 FBRC scheme

7.4.1 Problem statement

The transmission latency between any two nodes, for example, is denoted as l_{ij} and other major

symbols are summarized in Table 7.2.

Table 7.2 Notations

Basic Notation	Description
I	A set of fog nodes in the original region
T	A set of tasks
R	A set of regions
C	A set of cloud servers
T	A pending task in T.
R'	A set of fog nodes in other regions ($R \backslash I$).
γ_{om}	Transmission latency between the original region and other region $m \in R'$
γ_{oc}	Transmission latency between the original region and cloud server $c \in C$
η_c	Service rate for cloud server $c \in C$
μ_r	Service rate for other region $m \in R'$
St	Average request rate of task t from client i
x_{tic}	The $\{0,1\}$ variable indicates whether the cloud server c is selected.
x_{tim}	The $\{0,1\}$ variable indicates whether another region m is selected.
x_{tio}	The $\{0,1\}$ variable indicates whether the original region is selected.

We consider applications in Fog computing consisting of a set of T tasks that need to be completed in a required time. Tasks can be run in parallel. Let R and C be the set of computing resources of regions and cloud servers for the applications. The computation resources in regions and cloud have different capacities and characteristics. Generally, cloud servers own more computation resources and process tasks faster compared to those of regions, but they are often located far away from their clients. In other words, these servers can provide faster computation but have higher latency because of the long transfer time between clients and servers. In contrast, the computational results may be transferred quickly from a region to its

surrounding clients but the processing time to complete a client's request may be very large because of limited computing resources. For these reasons, FBRC effectively deploys resources of multiple regions as well as cloud servers to reduce task completion time. We assume that the processing time of tasks and the number of hops for transferring data between regions and cloud servers are known. In doing so, we could pre-compute the processing and transmission time of tasks along with the amount of data throughput.

Task processing requests are submitted randomly at each fog node. For each task t in a set T of tasks, the average task arrival to a fog node $i \in R$ is τ_{ti}. Without loss of generality, we assume that a region $r \in R$ has the computational capacity μ_r for FBRC and a cloud server $c \in C$ has computation rate η_c. Let s_t denote the size of task $t \in T$.

In the FBRC, a task can be processed by the fog nodes in a region itself, or in multiple regions, and/or in cloud servers. Although static scheduling may be feasible for task processing locally, limited resources prevent handling all tasks locally in a region. In fact, it is difficult to estimate accurately computational requirements of task requests. Alternatively, more flexibility and higher efficiency could be obtained if the task scheduling process can choose stochastic strategies based on the distributions of submitted tasks and their requirements. We denote p_{tim} as the probability that a task $t \in T$ is submitted from a fog node $i \in I$ in the original region to other regions R' where $R' \subseteq R$. Let p_{tic} be the probability that a task t request submitted from the original region is sent to cloud node $c \in C$ for handling disk reading and task processing. Let p_{tio} be the probability that task $t \in T$ request is processed at the original region itself.

A task may be processed at different regions or cloud servers other than always in a particular region due to the availability of computing resources. Usually, cloud servers have higher service rates than fog nodes in regions since they are shared by multiple clients for many tasks. How to balance the requests among the original region, other regions, and cloud servers is a critical issue to task completion time. In addition, the transmission latency is also another critical issue because tasks may be processed faster at cloud servers or multiple regions that have available resources but transferring data among nodes results in a high delay in the whole process. Clearly, the probability distribution of submitted requests and strategies of the scheduling process play big roles in minimizing task completion times.

7.5 Problem formulation

In this section, we provide a formal description of our problem with consideration of task scheduling by formulating it into an Integer program problem.

7.5.1 Task completion time analysis

7.5.1.1 Computation time

The computation time of a task depends on where the processing is scheduled. If *t* is distributed on cloud server $c \in C$ with the service rate η_c, the server *c* may be shared by multiple clients for different tasks. The overall task arrival rate at cloud server *c* thus can be calculated as

$$\theta_c = \sum_{t\in T}\sum_{i\in I} p_{tic}\,\tau_{ti}, \forall c \in C \qquad (7.1)$$

Recall that the task computation time is exponentially distributed on a cloud server, which is based on an M/M/1 queue. The average computation time of all tasks at cloud server $c \in C$ can be calculated as

$$\sigma_c^{tc} = \frac{1}{\eta_C - \sum_{t\in T}\sum_{i\in I} p_{tic}\tau_{ti}}, \forall c \in C, \qquad (7.2)$$

where we must ensure that

$$\eta_C > \sum_{t\in T}\sum_{i\in I} p_{tic}\,\tau_{ti}, \forall c \in C \qquad (7.3)$$

If *t* is sent to another region $m \in R'$ with the service rate μ_r, the region *m* may be requested from clients for different tasks. Therefore, the overall task arrival rate at region *m* can be calculated as

$$\theta_m = \sum_{t\in T}\sum_{i\in I} p_{tim}\,\tau_{ti}, \forall m \in R' \qquad (7.4)$$

Similarly, the computation on another region is also based on an M/M/1 queue. It can be also calculated as

$$\sigma_m^{tc} = \frac{1}{\mu_r - \sum_{t\in T}\sum_{i\in I} p_{tim}\tau_{ti}}, \forall m \in R'. \qquad (7.5)$$

We shall also guarantee that

$$\mu_r > \sum_{t\in T} p_{tim}\,\tau_{ti} = 1, \forall r \in R' \qquad (7.6)$$

Finally, the computation is processed at the original region. The overall task arrival rate at the original region can be calculated as follows.

$$\theta_0 = \sum_{t\in T}\sum_{i\in I} p_{tio}\,\tau_{ti}, \forall i \in I. \qquad (7.7)$$

We can derive the average computation time on the fog node $i \in I$ at the original region as

$$\sigma_o^{tc} = \frac{1}{\mu_o - \sum_{t\in T}\sum_{i\in I} p_{tio}\tau_{ti}}, \forall i \in I. \quad (7.8)$$

We shall also guarantee that

$$\mu_i > \sum_{t\in T} p_{tio}\,\tau_{ti} = 1, \forall i \in I. \quad (7.9)$$

7.5.1.2 Transmission time

If the computation tasks and data retrieval at the original region *i* are handled by another region *m* and cloud server *c*, respectively, the transmission latency between *i, m* and *c* shall be considered. We use binary variables to indicate other region and cloud server selection as

$$x_{tic} = \begin{cases} 1, \text{the task } t \text{ from the original region } i \text{ is handled by cloud server } c, \\ 0, \text{otherwise} \end{cases} \quad (7.10)$$

Similarly, we define

$$x_{tim} = \begin{cases} 1, \text{the task } t \text{ from the original region } i \text{ is handled by region } m, \\ 0, \text{otherwise} \end{cases} \quad (7.11)$$

We also define

$$x_{tio} = \begin{cases} 1, \text{the task } t \text{ is handled by original region}, \\ 0, \text{otherwise} \end{cases} \quad (7.12)$$

Hence, we have the relationship between the probabilities and decision variables as follows: 1) when $p_{tim} > 0$, the value of x_{tim} shall be 1, indicating that the region *m* is selected; 2) when $p_{tic} > 0$, the value of x_{tic} shall be 1, indicating that cloud server *c* is selected; 3) when $p_{tio} > 0$, the value of x_{tio} shall be 1, indicating that the original region is selected. Therefore, we have the following relationships

$$p_{tim} < x_{tim} < A_{ptim},\ \forall t \in T, \forall i \in I, \forall m \in R' \quad (7.13)$$

and

$$p_{tic} < x_{tic} < A_{ptic},\ \forall t \in T, \forall i \in I, \forall c \in C \quad (7.14)$$

and

$$p_{tio} < x_{tio} < A_{ptio},\ \forall t \in T, \forall i \in I \quad (7.15)$$

where A is an arbitrarily large number.

For tasks scheduled onto cloud servers and other regions, all the transmissions for data retrieval process happened between the original region *i* and $m \in R'$, $c \in C$. Let n_{tm} and n_{tc} be the average data retrieval during task execution from the original region *i* to another region *m* and cloud *c*, respectively. The expected transmission time of task t from the original region *i* allocated to *m* and *c* can be calculated as

$$\sigma_{tim}^{tt} = 2n_{tm} + \gamma_{om}, \forall t \in T, \forall i \in I, \forall m \in R'. (7.16)$$

and the expected transmission time of task *t* between the original region *i* and a cloud server *c* can be calculated as

$$\sigma_{tic}^{tt} = 2n_{tc} + \gamma_{oc}, \forall t \in T, \forall i \in I, \forall c \in C. (7.17)$$

7.5.1.3 Task completeness constraints

To ensure the Quality of service (QoS), it is required that all requests submitted to the original region must be processed, either at regions or cloud servers. This results in

$$\sum_{m \in R'} p_{tim} + \sum_{i \in I} p_{tio} + \sum_{c \in C} p_{tic} = 1, \forall t \in T, \forall i \in I. \quad (7.18)$$

Tasks need to be completed without exceeding the deadline. This leads to

$$\sum_{c \in C} x_{tic} . (\sigma_m^{tc} + \sigma_{tim}^{tt}) + \sum_{m \in R,} x_{tim} . (\sigma_c^{tc} + \sigma_{tic}^{tt}) + \sum_{i \in I} x_{tio} . \sigma_o^{tc} \leq D, \forall t \in T, \forall i \in I. \quad (7.19)$$

7.5.2 An FBRC-IP formulation

Multiple tasks are submitted to fog nodes at an original region. These tasks will be allocated to appropriated fog nodes at the current region, other regions and cloud servers based on their requirements. If we allocate a task to a high performance cloud server which is located far from the client, this task may not be finalized in the expected time due to large transmission time. The aim of the FBRC is to minimize the maximum average task completion times. Let φ be the maximum time introduced in completing the task *t*. Thus, we have

$$x_{tic} . (\sigma_m^{tc} + \sigma_{tim}^{tt}) < \varphi \quad (7.20)$$

and

$$x_{tim} . (\sigma_c^{tc} + \sigma_{tic}^{tt}) < \varphi, \quad (7.21)$$

and

$$x_{tio} . \sigma_o^{tc} < \varphi. \quad (7.22)$$

The problem is solved by minimizing φ. In short, we can formulate the task maximization completion time-minimization with consideration of task scheduling as an Integer Programming problem (Called FBRC-IP), as follow:

FBRC-IP:

$minimize\ \varphi,$

subject to the following constraints:

- *Service rate as (7.3), (7.6), (7.9)*
- *Computation resources as (7.13-7.17)*
- *Task completeness as (7.18), (7.19)*
- *Maximum completion time (7.20-7.22)*

The objective function is to minimize the task completion time when executing requests at regions and cloud servers.

7.5.3 Algorithm design

In this section, we present the design algorithm to find an optimal resource scheduling algorithm for FBRC that minimizes the completion time. The steps of the strategy are given as follows:

1. Requests are sorted in ascending order of latency-constraints.

2. Computation resources are allocated according to the policy that aims to minimize the computation latency for each request. This latency can be expressed as the ratio of the computation throughput and the latency requirements.

3. Pending requests are sorted in ascending order of latency-constraints.

4. Regions and cloud servers are allocated with the objective of minimizing the overall FRBC latency.

7.6 Evaluation and results of the data protection model

7.6.1 Testbed

We have implemented a test bed which focuses on the data protection that includes an authorization service based on our FPRBAC model and the MS. We assume that applications are deployed across multiple regions. Hence, data are stored in a distributed data warehouse. A new data is generated and stored at fog nodes in buffers before synchronizing the data to its servers. Similarly, data is retrieved from clouds and stored at fog nodes to serve for requests. Hence, FPRBAC service can operate functions as a security guard in the entire data protection model of Fog computing. Requests are submitted by users and also other fog nodes. Our fog nodes are implemented on a Dell PowerEdge R730 Intel Xeon E5-2660 v3 2.6GHz 16 GB and

an IBM Blade Center HS20 including 8 Blades, an IBM DS400, a FC switch, and 3 Ethernet switches. We also use two Raspberry Pi 3 as the gateway located at each fog network. The LRD are deployed at fog nodes to reduce latency in serving queries from the MS. In fact, we also implemented the LRD at the cloud for synchronizing among LRDs at fog nodes. Specifically, the Amazon EC2 (EC2, 2014) was used to provide resources and the LRD as global location database. EC2 instances were provisioned in US regions. All LRD are run on MySQL. Fog nodes were built based on the fog project package (fogproject, 2016) and deployed on two fog systems. Packages of our model were deployed on these fog instances and requests were able to access data via an RMI interface. GCM (GCM, 2014) was devised to notify users when there are any operations on the data.

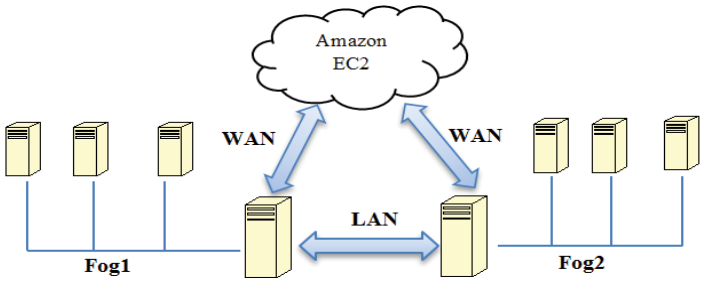


Figure 7.5 Testbed setup

7.6.2 Performance evaluation

This section presents the performance evaluation of the proposed model. We investigate different aspects as follows: 1) processing time evaluation; 2) data protection evaluations; 3) mobility evaluations.

Suppose that there are N requests for fog resources. For each request $r \in N$, the processing time consists of elements as follows: 1) the trust establishment time from two regions, denoted as tt; 2) the lookup service time from client to fog node, denoted as lt; 3) the verification and MS time at Fog node, denoted vt; 4) the response time from fog node to client, denoted rt. Each node of Fog computing network represents a method and an edge represents whether a method is invoked. Assume that the execution cost of a method m in a fog node is m_f and on cloud server m_c. So, the duration of executing a request on a fog node is:

$$m_f = \sum_{\forall i \in N} tt_i + lt_i + vt_i + rt_i \quad (7.23)$$

If two regions have established the trust relationship, tt is set equal to 0. Regarding the execution

time of a request at cloud server in (Dang et al., 2015), the cost of executing a request is:

$$m_c = \sum_{\forall\, i \in N} t_{lookup\ service_i} + t_{verification\ and\ data\ mobility_i} + t_{Data\ operation_i} \quad (7.24)$$

7.7 Response time performance - comparing Fog and Cloud processing times.

Table 7.3 illustrates the execution time of services for a request. It can be seen that the response time of fog node is much less than that of cloud instance, approximately ½ execution time. Notably, the proposed model introduced small latency in the response time, approximately a quarter of cloud's response time. Therefore, the proposed model finds it possible to integrate fog applications to protect data while still satisfying latency requirements.

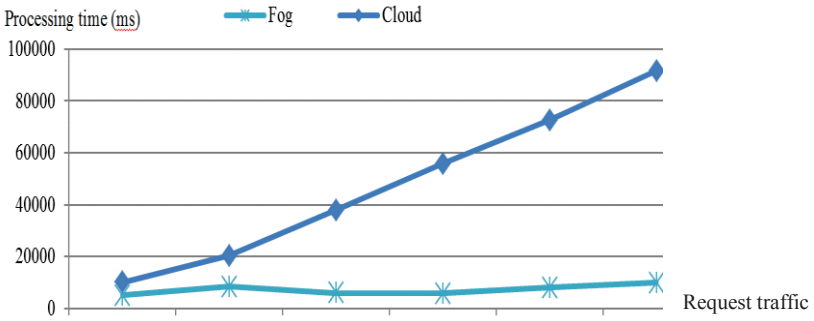


Figure7.6 Time cost of verification and MS for different requests

Table 7.3 Execution time of components

Execution components	**Fog**	**Cloud**
Lookup service	350 ms	651.8 ms
Verification and MS	160 ms	500 ms
Overall response time for a request	635 ms	3748 ms

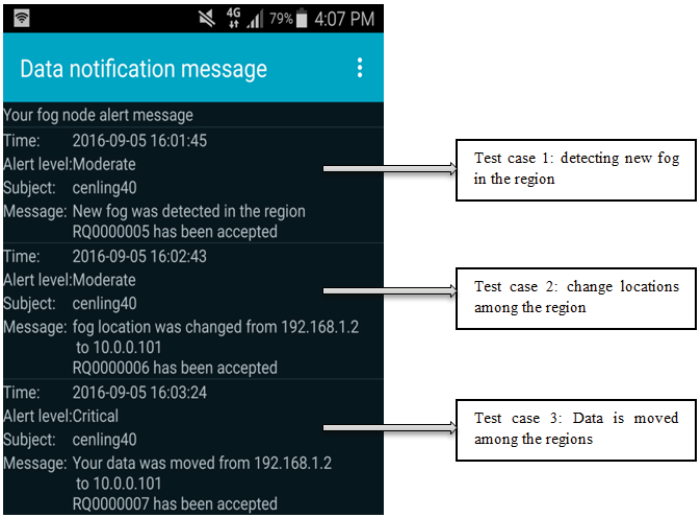


Figure 7.7 FPRBAC violation notification view in Samsung Galaxy Note 4

Figure 7.6 illustrates the execution time of lookup and verification and MS for a different number of requests. It can be seen that along with the increasing of the number of requests, the time cost of verification and MS increases obviously for our model. However, comparing to the overheads at the cloud, our model only experienced slight latency due to processing many verification requests. Notably, with large IoTs population, excessive traffic from the edges of the network to the Cloud affect latency severely.

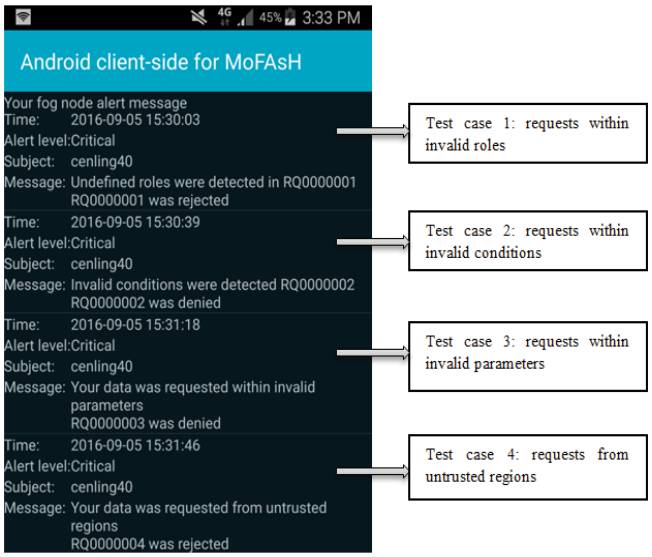


Figure 7.8 FPRBAC violation notification view in Samsung Galaxy Note 4

7.7.1 Data protection performance - detecting violations of access control policies

In this section, we conduct data violations against access control policies of our proposed model

and trust-based region verification to evaluate the proposed model reacting correctly by the following test cases: 1) requests try to violate the role based access control and bypass verification procedure within undefined roles; 2) clients request to access the data within insufficient conditions; 3) requests with invalid parameters attempt to gain access to the data; 4) users from untrusted regions attempt to retrieve the data at the new region. Figure 7.7 presents notification messages related to violations detected by the FPRBAC.

Direct access and Intrusion attacks. A request without any parameter is processed as an intrusion attack. Hence, the FPRBAC locks the source of the request and invokes the MS to notify data owner or administrator. In fact, a request with correct parameters is still required to go through the verification process where FPRBAC validates each parameter. Moreover, if the verification process exceeds the configured time threshold, the FPRBAC would reject the request and terminate the process.

Attack the role-based access control within undefined roles. Since FPRBAC has predefined roles associated wit data operations, a request is firstly verified whether it consists of roles matching with the list of roles in the system. Any inconsistent outcome created in the process would be actively detected when the FPRBAC is running and cause its termination. Retrieving these values and tampering with them to eliminate their functions are technically difficult.

Requests to access the data without valid conditions. At this checkpoint, roles and data operations are input and verified with conditions. If any inconsistent constraint occurred, requests are denied and a notification message is sent to the data owner.

Requests coming from untrusted regions. A request is only verified at FPRBAC if it comes from a trusted region to the fog system. Thus, the trust relationships need to be established among regions in order to process the request.

7.7.2 MS operations

Figure 7.8 shows notification messages associated with different location changes. We established three test cases: the first one is triggering the MS when there is a request to join in the current region; the second one is changing locations among regions, and the third one is triggering the MS when moving the data. All three test cases triggered the MS and then the notification message is sent to users' mobile devices immediately. The MS notifies data owner or administrator when it receives requests related to both fog locations and data locations. Then,

it triggers the notification center to send a message to inform data owner about the status of the request.

7.8 Numerical results of FBRC scheme

Simulation results are presented in this section to validate the task completion time by scheduling tasks to multiple regions and cloud servers. Without loss of generality, the regions and cloud servers throughout are assumed known. To evaluate the efficiency of our proposed scheduling scheme, we simulate requests, system capabilities, and scheduling strategies strictly following the system model defined in Section 7.4. Especially, in order to show the advantage of our proposed task scheduling scheme, we introduce two competitors namely Cloud-based ("Cloud") and Region-based ("Region") task scheduling schemes. The former schedules all computation tasks onto cloud servers until all the tasks are allocated or Cloud servers are fully loaded, while the latter handles all tasks on all regions until all of them are allocated or regions are fully loaded.

We select the parameter settings for the simulation as follows: each cloud server is with a total computation rate of *30* while the computation on each region is set as *10*. The current resources of regions and cloud servers are set randomly in the range of [*0.7,1*] as they are shared by tasks. The transmission latency among regions is randomly set in the range of [*0.01, 0.09*] while the transmission latency between a region and a cloud server are randomly set in the range of [*0.4,0.7*]. We investigate how FBRC performs over a range of parameters..

7.8.1 On effects of task arrival rate

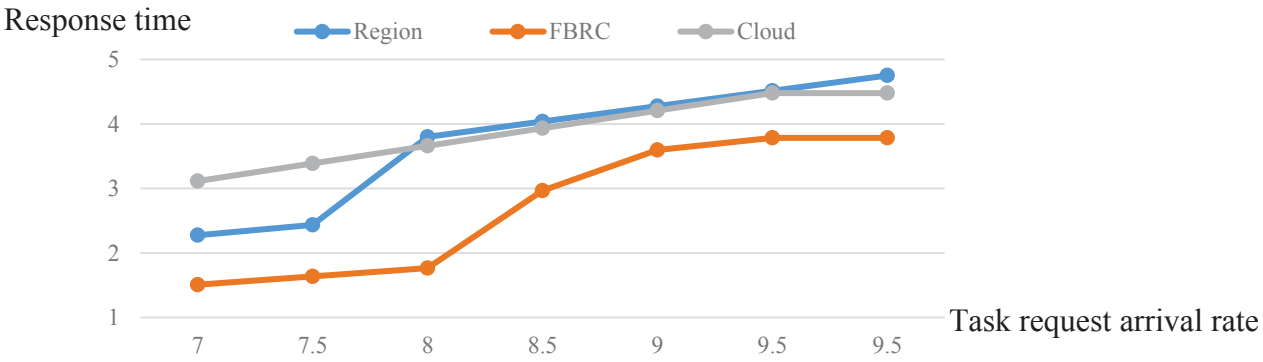


Figure 7.9 Task arrival rate

We first compare the task completion time of Region, FBRC, and Cloud under different task arrival rates from *7* to *10* (Figure 7.9). When task arrival rate increases, more regions will become involved in the process to perform requests. This is because the longer queue delay leads to larger computation time. However, the benefit of our proposed scheme over "Region" and "Cloud" can be observed when task arrival rate increases. Thus, it provides the flexibility in selecting computation resources between regions and Cloud servers.

7.8.2 On effects of computation service rate

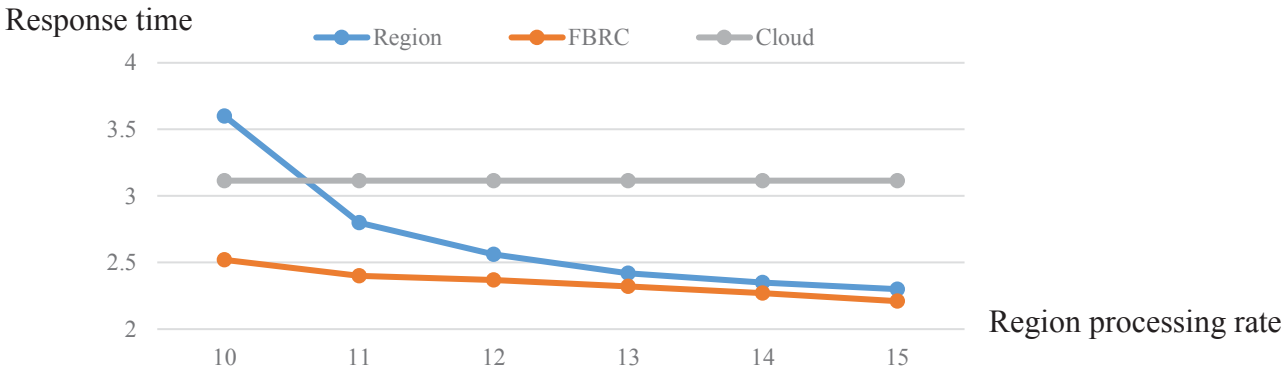


Figure 7.10 Region processing rate

Figure 7.10 and figure 7.11 present the task completion time of region service rate and cloud server rate. To observe the effect on region processing rate, we increase the service rates from *10* to *15*. It can be seen from Figure 7.10 that the completion time of Region and FBRC shows a decreasing trend of region service rate. When the service rates increase, regions process more requests to provide with a faster response. This results in the shorter computation time. The increase of service rate, in fact, leads to a significant impact on the computation time of the Region. Hence, for Region, the completion time decreases significantly. For FBRC, requests shall be processed by computation resources to obtain a faster response. Thus, more resources of regions will be used in processing requests to reduce completion time. For Cloud, as all requests are handled at cloud servers, there are no benefits from increasing the region service rate.

The similar trend is also presented in Figure 7.11. When computation service rate on cloud servers is small, FBRC's response latency is *2.82* while that of Cloud is *3.73* at service rate *18*. It can be explained that FBRC assigns more task to regions. However, when the service rate of cloud servers increases, the difference in response latency between FBRC and Cloud becomes

small. For example, when service rate is *36*, the gap decreases to *0.11*. Overall, FBRC can always schedule resource optimally to obtain the low response latency.

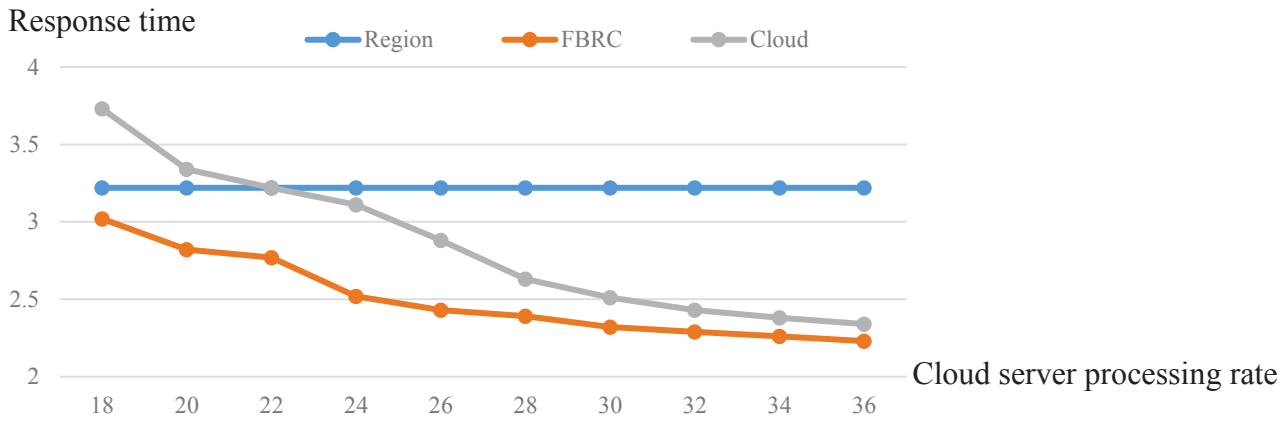


Figure 7.11 Computation cloud server processing rate

7.9 Summary

In this chapter, the data protection and task scheduling mechanisms for Fog computing are proposed. A new concept of "Region" in Fog computing is also introduced to provide nearby access for clients. The data protection model deals with protecting data and handling mobility in fog environment. The model features the RBTA model for trust translation among fog nodes of regions, the FPRBAC for access control at fog nodes, a mobility management service to handle location requests at a region. The task scheduling mechanism for Region-based cloud addresses resource constraints and sensitive latency requirements, whereas cloud resources are utilized appropriately for heavy computation tasks. The scheduling problem was formulated as an Integer program and solved by a heuristic algorithm. The experimental outcomes and numeric results demonstrated the feasibility and efficiency of our proposed models in term of data security and privacy, latency response and resource utilization.

We have presented our proposed schemes and models in addressing security and performance issues in this thesis. The next chapter will conclude our work and contributions in this research and suggest work to be carried out in the future.

CHAPTER 8 CONCLUSION AND FUTURE WORK

In this chapter, we first present the summary of the whole research in the thesis. Next, we outline the main contributions of our research in the field. From the work that has been investigated and carried out across the thesis, we suggest directions that should be studied in the future.

8.1 Conclusion

In this thesis, we identified critical issues in handling data for cloud computing. The issues associated with data mobility, data distribution and data processing have been investigated and addressed by proposing appropriate models and schemes that provide data management efficiently in term of data security, privacy and data processing. Firstly, although the handling of data in a cloud can be delegated by a provider to a sub-provider or another, CSPs do not often deploy the same protection schemes. Secondly, current big data models do not effectively handle files of different sizes and formats. Thirdly, mechanisms that are used to process users' sensitive data are not efficient in terms of system performance and resource utilization. Finally, cloud computing does not handle well local issues as it is located far from clients. As a consequence, CPSs are unable to provide appropriate security mechanisms for users' sensitive data allocated at cloud storage and to allocate fair resource allocation. Users' data, in fact, are not processed by appropriate secured cloud resources and guaranteed to be handled well when responding to low-latency requests. In this thesis, we have proposed innovative models and schemes to deal with identified issues. The data mobility management model protects data when it is moved to new clouds. The data distribution and replication schemes enable data to be distributed, retrieved, and accessed effectively and in a controlled way. In fact, users' sensitive data is also processed in a secure manner at different CSPs. It does not matter if a CSP is a private cloud or a public cloud. Meanwhile, the local issues are handled by the proposed Fog-based Region model.

The research is this thesis opened up a distinct and new type of cloud security service provisioning in the cloud computing. The security service mechanism namely DMaaS and active data protection work together to protect the data being moved among CSPs. The novelty of this research lies in security on data mobility, data distribution by using "collision" to cluster files and the integration of trust on cloud resources for big data processing. By proposing innovative services, schemes, and models, the major contribution of this thesis is that it enables CPSs to deliver trust computing services and resources and results in a widespread adoption of cloud services in high-security demand areas such healthcare. Moreover, the proposed Fog-based Region model also helps the broader adoption of sensitive-latency applications, where computing services and resources are provided to nearby users to provide real-time responses.

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

We comprehensively investigated issues in handling EHR related to security, distribution, and processing. By studying and analyzing related work, we identified research gaps which needed to be fulfilled in addressing these issues.

The proposed data mobility management model provides the DMM and active protection service. The DMM handles physical location changes of user's data in various cloud environments and ensures these locations are registered with LRD and recorded within the data itself. The active protection service deals with extending data protection when the data is moved among clouds. We also presented a comprehensive analysis of data mobility scenarios in various Cloud environments.

Our research also proposed a novel LRD that is capable of serving for tracing and tracking data locations. Furthermore, a new TDFS structure with recordable structure was designed to actively capture locations of requests. More importantly, a proposed establishing supervisor is able to deploy the equivalent data protection scheme at the visited cloud to protect the data.

The Experimental and simulation results demonstrated that the data mobility management model not only handles data protection at the original cloud but also provides the continuity of protection regardless of where the data is going and ensures data owners can track changes of data location and other information if necessary.

We proposed a novel HBFC scheme to distribute, store, and retrieve EHR efficiently in a cloud environment. The proposed scheme utilizes hash functions to cluster files in a controlled way. It makes use of the "Collision" property of the hash function that allows files to be distributed

into defined clusters for quick retrieval. It then utilizes P2P structure for data management. Based on the HBFC scheme, we also proposed the data replication management model that provides high reliability and availability for data in the cloud.

Simulation results demonstrated that the HBFC scheme allows us to cluster files in a controlled way to suit the specifications of a real system. The low computational complexity of hash functions enables the proposed scheme to achieve performance efficiency in searching for requested files.

In this thesis, we thoroughly and systematically investigated issues associated with users' sensitive data for big data applications. Hence, we proposed a novel trust metric to assign to cloud resources that are used in big data processing. We proposed a trust-based scheduling scheme for big data application with MapReduce. The scheme first provides trust assignments for cloud resources. It then schedules these resources optimally to process the data depending on sensitive levels of the data.

The simulation results demonstrated that the proposed scheme benefits CPSs and customers, enabling low cost data processing but achieving high gains in security. The scheduling scheme showed the flexibility of selecting CSPs regardless of whether they are private clouds or public clouds to process data efficiently and in a secure manner.

We comprehensively investigated data security and scheduling resource issues in Fog computing. We introduced a new concept of "Region" that handles performance issues to provide computing resources for nearby clients. The proposed data protection model governs a RBTA for trust translation among fog nodes of regions, a FPRBAC for access control at fog nodes, and a mobility management service to handle changes of users and fog devices' locations. Moreover, we proposed an efficient task scheduling mechanism for FBRC. The scheduling approach handles requests locally not just by a region but by multiple regions or cloud servers when additional resources are needed.

The implementation and numerical results demonstrated that regions allow users with assigned roles to access shared resources and data in a secure manner. When there are changes of data location among regions, data owners are informed in a timely way. Requests are scheduled among resources at regions and cloud servers effectively to achieve better performance and satisfy sensitive-latency requirements of applications.

In this thesis, we addressed specifically security and performance issues on EHR: at rest,

mobility and in processing; a) On data at rest, the ADCu empowers EHR data with capabilities of self-defend and self-protect against intrusions or violations; b) On mobility, we proposed the data mobility management model; c) On data distribution, we proposed the HBFC scheme to efficiently distribute a large-scale of EHR. With a, b, and c, the proposed mechanisms are only applicable to EHRs because of their special features and characteristics. On processing, the proposed schemes and models can be applied on both EHR and big data in general. However, not all mechanisms of big data are applicable to EHR.

8.2 Future work

Even though the data mobility model was implemented on real cloud infrastructures, the experiments were only carried out on small-scale clouds. It is of interest to deploy the model either with a large-scale of data or a large-scale and distributed clouds. Within various data moving scenarios, data locations are still stored at LRD, whereas data owners will be informed in a timely way through the MS.

The proposed schemes and models introduced new metrics, concepts, schemes and architectures in the field of research. Thus, future research can be studied in the following directions:

The experiments on the data mobility model are based on a few tiny cloud servers which have limited computing resources. The LRD is deployed centrally using the original cloud. Hence, increase of overheads results from the implementation of large-scale data as well as intensive requests. We still need to implement the model on large-scale cloud infrastructures which involve not only more powerful virtual cloud servers but also a large number of cloud servers. In the future, we intend to test the proposed model with a large number of requests together with various data moving scenarios. Thus, we will be able to improve our proposed protection mechanism and the performance in terms of scalability and availability.

The data distribution scheme based on hash functions and P2P systems demonstrated the efficiency in data management. In the future, the scheme can be extended to multiple hash spaces by adding more rings or partitioning the current hash space into small spaces. The scheme is able to handle data efficiently when the systems are skewed. It is also an interesting idea to combine hashing and indexing for the distribution scheme. Indexing mechanisms allows for the quickly identifying of clusters storing data, whereas hashing approaches provide local

searches in the clusters.

The research in this thesis focused on distributing and retrieving data from the requester sides which then processed the data. We did not discuss how the request can be handled at the requested nodes where data is located. Hence, it would be interesting to investigate how to send requests to requested nodes which process the data and return results to requesters. Hashing approaches will be used to identify data locations to send requests and receive results.

The trust-based scheduling scheme for big data application currently provides task processing on the trust aspect. In the future, we aim to extend the proposed scheme and further investigate on the energy aspect. The trust-based approach can serve as a basis for scheduling tasks to provide high trust execution and the energy consumptions can be considered as a second constraint in task scheduling. The scheme will achieve both security and energy efficiency requirements for big data applications. We will also investigate the trust-based and deadline-constrained scheme to address both security and performance requirements.

With the increase of the number of network elements and sensitive-delay applications, latency-response has become a critical issue. The proposed Fog-based Region is a promising mechanism to handle latency requirements, which allows computing resources to be allocated closer to clients. The proposed Fog-based Region will be employed as a basic computation mechanism for IoTs and SDN related applications. This will open new approaches for the implementation of sensitive-latency applications in the healthcare area. The current experiments for our proposed Fog-based Region in this thesis were based on a laboratory's mini-private cloud platform and small public cloud instances. In the future, we intend to test our proposed Fog-based Region in actual industry applications and a distributed cloud environment.

REFERENCES

(NECTAR), N. e. C. T. a. R. Available: http://nectar.org.au/ 2015].

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

ADRION, W. R. 1993. Research methodology in software engineering. Summary of the Dagstuhl Workshop on Future Directions in Software Engineering" Ed. Tichy, Habermann, and Prechelt, ACM Software Engineering Notes, SIGSoft. 36-37.

AHMED, S. & ABDULLAH, A. E-healthcare and data management services in a cloud. High Capacity Optical Networks and Enabling Technologies (HONET), 2011, 19-21 Dec. 2011. 248-252.

ALBELOOSHI, B., DAMIANI, E., SALAH, K. & MARTIN, T. 2016. Securing Cryptographic Keys in the Cloud: A Survey. *IEEE Cloud Computing,* 3**,** 42-56.

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

ALTMANN, J. & BEDANE, Z. B. 2009. A P2P File Sharing Network Topology Formation Algorithm Based on Social Network Information. INFOCOM Workshops 2009, IEEE, 19-25 April 2009. 1-6.

ANDROULAKI, E., SORIENTE, C., MALISA, L. & CAPKUN, S. Enforcing Location and Time-Based Access Control on Cloud-Stored Data. IEEE 34th International Conference on Distributed Computing Systems (ICDCS), June 30 2014-July 3 2014. 637-648.

ANGIN, P., BHARGAVA, B., RANCHAL, R., SINGH, N., LINDERMAN, M., OTHMANE, L. B. & LILIEN, L. An Entity-Centric Approach for Privacy and Identity Management in Cloud Computing. Reliable Distributed Systems, 2010 29th IEEE Symposium on, Oct. 31 2010-Nov. 3 2010. 177-183.

APACHE. 2017. *Hadoop: Open source implementation of MapReduce* [Online]. Available: http://hadoop.apache.org/.

ATENIESE, G., PIETRO, R. D., MANCINI, L. V. & TSUDIK, G. 2008. Scalable and efficient provable data possession. *Proceedings of the 4th international conference on Security and privacy in communication netowrks.* Istanbul, Turkey: ACM.

AZURE. 2014. *Microsoft Azure* [Online]. Available: https://azure.microsoft.com 2014].

BAHGA, A. & MADISETTI, V. K. 2013. A Cloud-based Approach for Interoperable Electronic Health Records (EHRs). *Biomedical and Health Informatics, IEEE Journal of,* 17**,** 894-906.

BAMIAH, M., BROHI, S., CHUPRAT, S. & AB MANAN, J. L. A study on significance of adopting cloud computing paradigm in healthcare sector. Cloud Computing Technologies, Applications and Management (ICCCTAM), 2012 International Conference on, 8-10 Dec. 2012. 65-68.

BENSON, K., DOWSLEY, R. & SHACHAM, H. 2011. Do you know where your cloud files are? *Proceedings of the 3rd ACM workshop on Cloud computing security workshop.* Chicago, Illinois, USA: ACM.

BERNSTEIN, D. J. 2015. *D. J. Bernstein* [Online]. Available: http://cr.yp.to/djb.html.

BETGE-BREZETZ, S., KAMGA, G. B., DUPONT, M. P. & GUESMI, A. Privacy Control in Cloud VM File Systems. Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on, 2-5 Dec. 2013. 276-280.

BEUNARDEAU, M., CONNOLLY, A., GERAUD, R. & NACCACHE, D. 2016. Fully Homomorphic

Encryption: Computations with a Blindfold. *IEEE Security & Privacy,* 14**,** 63-67.

BONOMI, F., MILITO, R., NATARAJAN, P. & ZHU, J. 2014. Fog Computing: A Platform for Internet of Things and Analytics. *In:* BESSIS, N. & DOBRE, C. (eds.) *Big Data and Internet of Things: A Roadmap for Smart Environments.* Cham: Springer International Publishing.

BONOMI, F., MILITO, R., ZHU, J. & ADDEPALLI, S. 2012. Fog computing and its role in the internet of things. *Proceedings of the first edition of the MCC workshop on Mobile cloud computing.* Helsinki, Finland: ACM.

BOWERS, K. D., JUELS, A. & OPREA, A. 2009. Proofs of retrievability: theory and implementation. *Proceedings of the 2009 ACM workshop on Cloud computing security.* Chicago, Illinois, USA: ACM.

CAMOUS, F., MCCANN, D. & ROANTREE, M. Capturing Personal Health Data from Wearable Sensors. Applications and the Internet, 2008. SAINT 2008. International Symposium on, July 28 2008-Aug. 1 2008. 153-156.

CARRA, D., LO CIGNO, R. & BIERSACK, E. W. 2008. Stochastic Graph Processes for Performance Evaluation of Content Delivery Applications in Overlay Networks. *Parallel and Distributed Systems, IEEE Transactions on,* 19**,** 247-261.

CASOLA, V., BENEDICTIS, A. D., RAK, M. & VILLANO, U. SLA-Based Secure Cloud Application Development: The SPECS Framework. 2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 21-24 Sept. 2015. 337-344.

CATTELL, R. 2011. Scalable SQL and NoSQL data stores. *SIGMOD Rec.,* 39**,** 12-27.

CHANG, H., KODIALAM, M., KOMPELLA, R. R., LAKSHMAN, T. V., LEE, M. & MUKHERJEE, S. Scheduling in mapreduce-like systems for fast completion time. 2011 Proceedings IEEE INFOCOM, 10-15 April 2011. 3074-3082.

CHEN, C. H., LIN, J. W. & KUO, S. Y. 2015. MapReduce Scheduling for Deadline-Constrained Jobs in Heterogeneous Cloud Computing Systems. *IEEE Transactions on Cloud Computing,* PP**,** 1-1.

CHEN, G., HU, T., JIANG, D., LU, P., TAN, K. L., HOANG TAM, V. & WU, S. BestPeer++: A Peer-to-Peer Based Large-Scale Data Processing Platform. Data Engineering (ICDE), 2012 IEEE 28th International Conference on, 1-5 April 2012. 582-593.

CHEN, L. 2014. *Achieving Trust-oriented Data Protection in the Cloud Environment.* University of Technology, Sydney.

CHEN, L. & HOANG, D. B. Novel Data Protection Model in Healthcare Cloud. High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on, 2-4 Sept. 2011. 550-555.

CHEN, L. & HOANG, D. B. Towards Scalable, Fine-Grained, Intrusion-Tolerant Data Protection Models for Healthcare Cloud. IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 16-18 Nov. 2011. 126-133.

CHEN, L. & HOANG, D. B. Active data-centric framework for data protection in cloud environment. ACIS 2012: Location, location, location: Proceedings of the 23rd Australasian Conference on Information Systems. ACIS, 1-11.

CHEN, L. & HOANG, D. B. Adaptive Data Replicas Management Based on Active Data-centric Framework in Cloud Environment. High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on, 13-15 Nov. 2013. 101-108.

CHEN, L. & HOANG, D. B. Addressing Data and User Mobility Challenges in the Cloud. IEEE Sixth International Conference on Cloud Computing (CLOUD), June 28 2013-July 3 2013. 549-556.

CONG, W., QIAN, W., KUI, R. & WENJING, L. Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing. Proceedings of IEEE INFOCOM, 14-19 March 2010. 1-9.

CONSORTIUM, I., SCHRECKER, S., SOROUSH, H. & MOLINA, J. 2016. *Industrial Internet of Things Volume G4: Security Framework*, CreateSpace Independent Publishing Platform.

DAN LIN & SQUICCIARINI, A. 2010. Data protection models for service provisioning in the cloud.

Proceedings of the 15th ACM symposium on Access control models and technologies. Pittsburgh, Pennsylvania, USA: ACM.

DANG, T. D. & HOANG, D. Data Mobility as a Service. 2016 IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW), 27-30 June 2016. 67-71.

DANG, T. D., HOANG, D. & NANDA, P. Data Mobility Management Model for Active Data Cubes. The 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 20-22 Aug. 2015. 750-757.

DASGUPTA, S., PAPADIMITRIOU, C. & VAZIRANI, U. Algorithms; 2006. McGraw-Hill.

DE RIDDER, M., CONSTANTINESCU, L., LEI, B., YOUN HYUN, J., KUMAR, A., JINMAN, K., FENG, D. D. & FULHAM, M. A web-based medical multimedia visualisation interface for personal health records. Computer-Based Medical Systems (CBMS), 2013 IEEE 26th International Symposium on, 20-22 June 2013. 191-196.

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

DECANDIA, G., HASTORUN, D., JAMPANI, M., KAKULAPATI, G., LAKSHMAN, A., PILCHIN, A., SIVASUBRAMANIAN, S., VOSSHALL, P. & VOGELS, W. 2007. Dynamo: amazon's highly available key-value store. *SIGOPS Oper. Syst. Rev.,* 41**,** 205-220.

DOLEV, D. & YAO, A. C. 1981. On the security of public key protocols. *Proceedings of the 22nd Annual Symposium on Foundations of Computer Science.* IEEE Computer Society.

DOLIN, R. H., ALSCHULER, L., BOYER, S., BEEBE, C. & GROUP, K. E. An update on HL7's XML-based document representation standards. Proceedings of the AMIA Symposium. American Medical Informatics Association, 190.

DONG, B., ZHENG, Q., TIAN, F., CHAO, K.-M., MA, R. & ANANE, R. 2012. An optimized approach for storing and accessing small files on cloud storage. *J. Netw. Comput. Appl.,* 35**,** 1847-1862.

DOUKAS, C., MAGLOGIANNIS, I., KOUFI, V., MALAMATENIOU, F. & VASSILACOPOULOS, G. Enabling data protection through PKI encryption in IoT m-Health devices. Bioinformatics & Bioengineering (BIBE), 2012 IEEE 12th International Conference on, 11-13 Nov. 2012. 25-29.

DSOUZA, C., AHN, G. J. & TAGUINOD, M. Policy-driven security management for fog computing: Preliminary framework and a case study. 2014 IEEE 15th International Conference on Information Reuse and Integration (IRI), 13-15 Aug. 2014. 16-23.

EC2, A. 2014. *Amazon Elastic Compute Cloud* [Online]. Available: http://aws.amazon.com/.

FERRAIOLO, D. F., SANDHU, R., GAVRILA, S., KUHN, D. R. & CHANDRAMOULI, R. 2001. Proposed NIST standard for role-based access control. *ACM Trans. Inf. Syst. Secur.,* 4**,** 224-274.

FIRDHOUS, M., GHAZALI, O. & HASSAN, S. A trust computing mechanism for cloud computing with multilevel thresholding. 6th IEEE International Conference on Industrial and Information Systems (ICIIS), 16-19 Aug. 2011. 457-461.

FOGPROJECT. 2016. *fogproject* [Online]. Available: https://fogproject.org/.

FOSTER, I., YONG, Z., RAICU, I. & SHIYONG, L. Cloud Computing and Grid Computing 360-Degree Compared. GCE '08 Grid Computing Environments Workshop, 12-16 Nov. 2008. 1-10.

GAO, Z., DESALVO, N., KHOA, P. D., KIM, S. H., XU, L., RO, W. W., VERMA, R. M. & SHI, W. Integrity Protection for Big Data Processing with Dynamic Redundancy Computation. Autonomic Computing (ICAC), 2015 IEEE International Conference on, 7-10 July 2015. 159-160.

GARSON, K. & ADAMS, C. 2008. Security and privacy system architecture for an e-hospital environment. *Proceedings of the 7th symposium on Identity and trust on the Internet.* Gaithersburg, Maryland: ACM.

GCM. 2014. *Google Cloud Messaging for Android* [Online]. Available: https://developer.android.com/google/gcm/index.html.

GHEMAWAT, S., GOBIOFF, H. & LEUNG, S.-T. 2003. The Google file system. *SIGOPS Oper. Syst. Rev.,* 37**,** 29-43.

GIUSEPPE DECANDIA, DENIZ HASTORUN, MADAN JAMPANI, GUNAVARDHAN KAKULAPATI, AVINASH LAKSHMAN, ALEX PILCHIN, SWAMINATHAN SIVASUBRAMANIAN, PETER VOSSHALL & VOGELS, W. 2007. Dynamo: amazon's highly available key-value store. *SIGOPS Oper. Syst. Rev.,* 41**,** 205-220.

GOH, E.-J., SHACHAM, H., MODADUGU, N. & BONEH, D. 2003. SiRiUS: Securing Remote Untrusted Storage. *NDSS,* 3**,** 131-145.

GOYAL, V., PANDEY, O., SAHAI, A. & WATERS, B. 2006. Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM conference on Computer and communications security.* Alexandria, Virginia, USA: ACM.

HÄYRINEN, K., SARANTO, K. & NYKÄNEN, P. 2008. Definition, structure, content, use and impacts of electronic health records: A review of the research literature. *International Journal of Medical Informatics,* 77**,** 291-304.

HIMSS. 2017. *Healthcare Information and Management Systems Society* [Online]. Available: http://www.himss.org/.

HUANG, C., ZHU, S. & WU, D. Towards Trusted Services: Result Verification Schemes for MapReduce. Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on, 13-16 May 2012. 41-48.

IAKOVIDIS, I. 1998. Towards personal health record: current situation, obstacles and trends in implementation of electronic healthcare record in Europe. *International Journal of Medical Informatics,* 52**,** 105-115.

JAE-WOO, L. Mobility Management Using Frequently Visited Location Database. Multimedia and Ubiquitous Engineering, 2007. MUE '07. International Conference on, 26-28 April 2007. 159-163.

JCHORD. 2016. *JChord* [Online]. Available: https://code.google.com/archive/p/joonion-jchord/downloads.

JIE, H., SHARAF, M. & CHIN-TSER, H. A Hierarchical Framework for Secure and Scalable EHR Sharing and Access Control in Multi-cloud. Parallel Processing Workshops (ICPPW), 2012 41st International Conference on, 10-13 Sept. 2012. 279-287.

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

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

JUNG, B. DICOM-X - seamless integration of medical images into the EHR. Computer-Based Medical Systems, 2005. Proceedings. 18th IEEE Symposium on, 23-24 June 2005. 203-207.

JUNQING, S., SUNDARA-RAJAN, K., LINDSEY, L., MAMISHEV, A., JOHNSON, E., TEREDESAI, A. & KRISTAL, A. A pervasive Dietary Data Recording System. IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 21-25 March 2011. 307-309.

KAANICHE, N., LAURENT, M. & BARBORI, M. E. CloudaSec: A novel public-key based framework to handle data sharing security in clouds. 2014 11th International Conference on Security and Cryptography (SECRYPT), 28-30 Aug. 2014. 1-14.

KALLAHALLA, M., RIEDEL, E., SWAMINATHAN, R., WANG, Q. & FU, K. 2003. Plutus: Scalable Secure File Sharing on Untrusted Storage. *Proceedings of the 2nd USENIX Conference on File and Storage Technologies.* San Francisco, CA: USENIX Association.

KAMARA, S. & LAUTER, K. 2010a. Cryptographic cloud storage. *Proceedings of the 14th international conference on Financial cryptograpy and data security.* Tenerife, Canary Islands, Spain: Springer-Verlag.

KAMARA, S. & LAUTER, K. 2010b. 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.

KATHRYN GARSON & ADAMS, C. 2008. Security and privacy system architecture for an e-hospital environment. *Proceedings of the 7th symposium on Identity and trust on the Internet.* Gaithersburg, Maryland, USA: ACM.

KHAN, W. A., HUSSAIN, M., AFZAL, M., AMIN, M. B. & SUNGYOUNG, L. Healthcare standards based sensory data exchange for Home Healthcare Monitoring System. Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE, Aug. 28 2012-Sept. 1 2012. 1274-1277.

KO, R. K. L., JAGADPRAMANA, P. & LEE, B. S. Flogger: A File-Centric Logger for Monitoring File Access and Transfers within Cloud Computing Environments. 2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, 16-18 Nov. 2011. 765-771.

KO, R. K. L., KIRCHBERG, M. & LEE, B. S. From system-centric to data-centric logging - Accountability, trust & security in cloud computing. 2011 Defense Science Research Conference and Expo (DSR), 3-5 Aug. 2011. 1-4.

KO, R. K. L., LEE, B. S. & PEARSON, S. 2011c. Towards Achieving Accountability, Auditability and Trust in Cloud Computing. *In:* ABRAHAM, A., MAURI, J. L., BUFORD, J. F., SUZUKI, J. & THAMPI, S. M. (eds.) *Advances in Computing and Communications: First International Conference, ACC 2011, Kochi, India, July 22-24, 2011, Proceedings, Part IV.* Berlin, Heidelberg: Springer Berlin Heidelberg.

KO, R. K. L., RUSSELLO, G., NELSON, R., PANG, S., CHEANG, A., DOBBIE, G., SARRAFZADEH, A., CHAISIRI, S., ASGHAR, M. R. & HOLMES, G. 2015. STRATUS: Towards Returning Data Control to Cloud Users. *In:* WANG, G., ZOMAYA, A., MARTINEZ, G. & LI, K. (eds.) *Algorithms and Architectures for Parallel Processing: ICA3PP International Workshops and Symposiums, Zhangjiajie, China, November 18-20, 2015, Proceedings.* Cham: Springer International Publishing.

KRUTZ, R. L. & VINES, R. D. 2010. *Cloud Security: A Comprehensive Guide to Secure Cloud Computing*, Wiley Publishing.

KUO, J. J., YANG, H. H. & TSAI, M. J. Optimal approximation algorithm of virtual machine placement for data latency minimization in cloud systems. IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, April 27 2014-May 2 2014. 1303-1311.

LAKSHMAN, A. & MALIK, P. 2010. Cassandra: a decentralized structured storage system. *SIGOPS Oper. Syst. Rev.,* 44**,** 35-40.

LE, H., HOANG, D. & SIMMONDS, A. PARM: a physically-aware reference model for overlay internetworking. 20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06), 18-20 April 2006. 6 pp.

LEE, K., KIM, D., HA, D., RAJPUT, U. & OH, H. On security and privacy issues of fog computing supported Internet of Things environment. 2015 6th International Conference on the Network of the Future (NOF), Sept. 30 2015-Oct. 2 2015. 1-3.

LEI, C., JI-JIANG, Y. & QING, W. Privacy-Preserving Data Publishing for Free Text Chinese Electronic Medical Records. Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual, 16-20 July 2012. 567-572.

LENK, A., KLEMS, M., NIMIS, J., TAI, S. & SANDHOLM, T. 2009. What's inside the Cloud? An architectural map of the Cloud landscape. *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing.* IEEE Computer Society.

LEWIS, F. 2013. *Industry view: Big data and health* [Online]. Available: http://www.ehi.co.uk/resources/industry-view/126 [Accessed 14/05/2014 2014].

LEWKO, A. & WATERS, B. 2011. Decentralizing attribute-based encryption. *Proceedings of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology.* Tallinn, Estonia: Springer-Verlag.

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. *In:* JAJODIA, S. & ZHOU, J. (eds.) *Security and Privacy in Communication Networks.* Springer Berlin

Heidelberg.

LI, X. & DU, J. 2013. Adaptive and attribute-based trust model for service level agreement guarantee in cloud computing. *IET Information Security,* 7**,** 39-50.

LIU, C., CHEN, J., YANG, L. T., ZHANG, X., YANG, C., RANJAN, R. & RAO, K. 2014. Authorized Public Auditing of Dynamic Big Data Storage on Cloud with Efficient Verifiable Fine-Grained Updates. *IEEE Transactions on Parallel and Distributed Systems,* 25**,** 2234-2244.

LIU, G., SHEN, H. & WARD, L. 2015. An Efficient and Trustworthy P2P and Social Network Integrated File Sharing System. *IEEE Transactions on Computers,* 64**,** 54-70.

LIU, W. & PARK, E. K. Big Data as an e-Health Service. 2014 International Conference on Computing, Networking and Communications (ICNC), 3-6 Feb. 2014. 982-988.

LIU, Z., JIANG, Z. L., WANG, X., YIU, S. M., ZHANG, C. & ZHAO, X. Dynamic Attribute-Based Access Control in Cloud Storage Systems. 2016 IEEE Trustcom/BigDataSE/ISPA, 23-26 Aug. 2016. 129-137.

LJ, B., A, G. & H, S. 2000. Describing Electronic Health Records Using XML Schema. *XML Asia Pacific*.

LUNA, J., TAHA, A., TRAPERO, R. & SURI, N. 2015. Quantitative Reasoning About Cloud Security Using Service Level Agreements. *IEEE Transactions on Cloud Computing,* PP**,** 1-1.

MAPREDUCESIMULATOR. 2016. *MapReduceSimulator* [Online]. Available: https://github.com/clsx524/MapReduceSimulator.

MASHAYEKHY, L., NEJAD, M. M., GROSU, D., ZHANG, Q. & SHI, W. 2015. Energy-Aware Scheduling of MapReduce Jobs for Big Data Applications. *IEEE Transactions on Parallel and Distributed Systems,* 26**,** 2720-2733.

MASSONET, P., NAQVI, S., PONSARD, C., LATANICKI, J., ROCHWERGER, B. & VILLARI, M. A Monitoring and Audit Logging Architecture for Data Location Compliance in Federated Cloud Infrastructures. Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on, 16-20 May 2011. 1510-1517.

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

MIN, Y. & YUANYUAN, Y. 2010. An Efficient Hybrid Peer-to-Peer System for Distributed Data Sharing. *Computers, IEEE Transactions on,* 59**,** 1158-1171.

MIRZAEI, N. 2009. Cloud Computing.

MODIC, J., TRAPERO, R., TAHA, A., LUNA, J., STOPAR, M. & SURI, N. 2016. Novel efficient techniques for real-time cloud security assessment. *Computers & Security,* 62**,** 1-18.

MOHER, T. & SCHNEIDER, G. M. 1982. Methodology and experimental research in software engineering. *International Journal of Man-Machine Studies,* 16**,** 65-87.

MOURADIAN, C., NABOULSI, D., YANGUI, S., GLITHO, R. H., MORROW, M. J. & POLAKOS, P. A. 2017. A Comprehensive Survey on Fog Computing: State-of-the-art and Research Challenges. *IEEE Communications Surveys & Tutorials,* PP**,** 1-1.

NARAYAN, S., GAGN\, M., \#233 & SAFAVI-NAINI, R. 2010. Privacy preserving EHR system using attribute-based infrastructure. *Proceedings of the 2010 ACM workshop on Cloud computing security workshop.* Chicago, Illinois, USA: ACM.

NARAYANAN, H. A. J. & GIINE, M. H. Ensuring access control in cloud provisioned healthcare systems. IEEE Consumer Communications and Networking Conference (CCNC), 9-12 Jan. 2011. 247-251.

NESHATPOUR, K., MALIK, M., GHODRAT, M. A., SASAN, A. & HOMAYOUN, H. Energy-efficient acceleration of big data analytics applications using FPGAs. 2015 IEEE International Conference on Big Data (Big Data), Oct. 29 2015-Nov. 1 2015. 115-123.

NGUYEN, T., HOANG, D. & SENEVIRATNE, A. Challenge-response trust assessment model for personal space IoT. 2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), 14-18 March 2016. 1-6.

NI, Q., BERTINO, E., LOBO, J. & CALO, S. B. 2009. Privacy-Aware Role-Based Access Control. *IEEE*

Security & Privacy, 7**,** 35-43.

NOMAN, A. & ADAMS, C. DLAS: Data Location Assurance Service for cloud computing environments. Tenth Annual International Conference on Privacy, Security and Trust (PST), 16-18 July 2012. 225-228.

OSANAIYE, O., CHEN, S., YAN, Z., LU, R., CHOO, K. & DLODLO, M. 2017. From cloud to fog computing: A review and a conceptual live VM migration framework. *IEEE Access,* PP**,** 1-1.

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

PANKAJ GOYAL & MIKKILINENI, R. Policy-Based Event-Driven Services-Oriented Architecture for Cloud Services Operation; Management. IEEE International Conference on Cloud Computing, 2009 (CLOUD '09), 21-25 Sept. 2009. 135-138.

PEI, X., WANG, Y., YAO, W., LIN, J. & PENG, R. Security Enhanced Attribute Based Signcryption for Private Data Sharing in Cloud. 2016 IEEE Trustcom/BigDataSE/ISPA, 23-26 Aug. 2016. 737-743.

PETERSON, Z. N. J., GONDREE, M. & BEVERLY, R. 2011. A position paper on data sovereignty: the importance of geolocating data in the cloud. *Proceedings of the 3rd USENIX conference on Hot topics in cloud computing.* Portland, OR: USENIX Association.

POPA, D., BOUDAOUD, K., BORDA, M. & CREMENE, M. Mobile cloud applications and traceability. RoEduNet International Conference 12th Edition Networking in Education and Research, 26-28 Sept. 2013. 1-4.

PU, L., CHEN, X., XU, J. & FU, X. 2016. D2D Fogging: An Energy-Efficient and Incentive-Aware Task Offloading Framework via Network-Assisted D2D Collaboration. *IEEE Journal on Selected Areas in Communications,* PP**,** 1-1.

RANCHAL, R., BHARGAVA, B., OTHMANE, L. B., LILIEN, L., ANYA, K., MYONG, K. & LINDERMAN, M. Protection of Identity Information in Cloud Computing without Trusted Third Party. Reliable Distributed Systems, 2010 29th IEEE Symposium on, Oct. 31 2010-Nov. 3 2010. 368-372.

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

RISSON, J. & MOORS, T. 2006. Survey of research towards robust peer-to-peer networks: search methods. *Comput. Netw.,* 50**,** 3485-3521.

ROBERTS, R. 1999. R. S. Dick, E. B. Steen and D. E. Dether (EDS), The Computer-based patient record: an essential technology for health care. Revised edition. Washington DC: Institute of Medicine, National Academy Press, 1997. ISBN 0-309-05532-6, 234 pages. £28.95. *The International Journal of Health Planning and Management,* 14**,** 74-75.

RUAN, A. & MARTIN, A. TMR: Towards a Trusted MapReduce Infrastructure. 2012 IEEE Eighth World Congress on Services, 24-29 June 2012. 141-148.

SAHAI, A. & WATERS, B. 2005. Fuzzy Identity-Based Encryption. *In:* CRAMER, R. (ed.) *Advances in Cryptology – EUROCRYPT 2005.* 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.

SCHUBERT, L. & JEFFERY, K. 2012. Advances in clouds. *Report of the Cloud Computing Expert Working Group. European Commission*.

SHACHAM, H. & WATERS, B. 2008. Compact Proofs of Retrievability. *Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology.* Melbourne, Australia: Springer-Verlag.

SHUCHENG, Y., CONG, W., KUI, R. & WENJING, L. Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing. Proceedings of IEEE INFOCOM, 14-19 March 2010. 1-9.

SINHA, P. & ZOLTNERS, A. A. 1979. The Multiple-Choice Knapsack Problem. *Operations Research,* 27**,** 503-515.

SONG, J., YAO, J. & WU, C. Cloud computing and its key techniques. Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, 12-14 Aug. 2011. 320-324.

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

STOICA, I., MORRIS, R., KARGER, D., KAASHOEK, M. F. & BALAKRISHNAN, H. 2001. Chord: A scalable peer-to-peer lookup service for internet applications. *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications.* San Diego, California, USA: ACM.

STOJMENOVIC, I. & WEN, S. The Fog computing paradigm: Scenarios and security issues. 2014 Federated Conference on Computer Science and Information Systems, 7-10 Sept. 2014. 1-8.

STORER, M. W., GREENAN, K. M., MILLER, E. L. & VORUGANTI, K. 2007. POTSHARDS: secure long-term storage without encryption. *2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference.* Santa Clara, CA: USENIX Association.

SUN, W., YU, S., LOU, W., HOU, Y. T. & LI, H. 2016. Protecting Your Right: Verifiable Attribute-Based Keyword Search with Fine-Grained Owner-Enforced Search Authorization in the Cloud. *IEEE Transactions on Parallel and Distributed Systems,* 27**,** 1187-1198.

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

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

TAHA, A., TRAPERO, R., LUNA, J. & SURI, N. AHP-Based Quantitative Approach for Assessing and Comparing Cloud Security. 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, 24-26 Sept. 2014. 284-291.

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.

TAN, Y. S., KO, R. K. L., JAGADPRAMANA, P., SUEN, C. H., KIRCHBERG, M., LIM, T. H., LEE, B. S., SINGLA, A., MERMOUD, K., KELLER, D. & DUC, H. Tracking of Data Leaving the Cloud. 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, 25-27 June 2012. 137-144.

TANCER, J. & VARDE, A. S. Cloud technology and EHR data management. IEEE 6th International Conference on Information and Automation for Sustainability (ICIAfS), 27-29 Sept. 2012. 112-117.

TANTISIRIROJ, W., SON, S. W., PATIL, S., LANG, S. J., GIBSON, G. & ROSS, R. B. 2011. On the duality of data-intensive file system design: reconciling HDFS and PVFS. *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis.* Seattle, Washington: ACM.

ULLAH, K. W. & AHMED, A. S. Demo Paper: Automatic Provisioning, Deploy and Monitoring of Virtual Machines Based on Security Service Level Agreement in the Cloud. Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on, 26-29 May 2014. 536-537.

ULUSOY, H., KANTARCIOGLU, M. & PATTUK, E. TrustMR: Computation integrity assurance system for MapReduce. Big Data (Big Data), 2015 IEEE International Conference on, Oct. 29 2015-Nov. 1 2015. 441-450.

VAQUERO, L. M. & RODERO-MERINO, L. 2014. Finding your Way in the Fog: Towards a Comprehensive Definition of Fog Computing. *SIGCOMM Comput. Commun. Rev.,* 44**,** 27-32.

VARADHARAJAN, V. & TUPAKULA, U. 2014. Security as a Service Model for Cloud Environment. *IEEE Transactions on Network and Service Management,* 11**,** 60-75.

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. *In:* FURNELL, S., LAMBRINOUDAKIS, C. & PERNUL, G. (eds.) *Trust, Privacy and Security in Digital Business.* Springer Berlin Heidelberg.

VUCETIC, M., UZELAC, A. & GLIGORIC, N. E-Health Transformation Model in Serbia: Design, Architecture and Developing. International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 10-12 Oct. 2011. 566-573.

WANG, Q., WANG, C., REN, K., LOU, W. & LI, J. 2011a. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems,* 22**,** 847-859.

WANG, S., AGRAWAL, D. & ABBADI, A. E. 2011b. A comprehensive framework for secure query processing on relational data in the cloud. *Proceedings of the 8th VLDB international conference on Secure data management.* Seattle, WA: Springer-Verlag.

WANG, Y., LI, X., JIN, Q. & MA, J. AB-Chord: An Efficient Approach for Resource Location in Structured P2P Networks. 2012 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing, 4-7 Sept. 2012. 278-284.

WANG, Y. & SHI, W. 2014. Budget-Driven Scheduling Algorithms for Batches of MapReduce Jobs in Heterogeneous Clouds. *IEEE Transactions on Cloud Computing,* 2**,** 306-319.

WANG, Y., UEHARA, T. & SASAKI, R. Fog Computing: Issues and Challenges in Security and Forensics. 2015 IEEE 39th Annual Computer Software and Applications Conference (COMPSAC), 1-5 July 2015. 53-59.

WANG, Z., HUANG, D., ZHU, Y., LI, B. & CHUNG, C. J. 2015b. Efficient Attribute-Based Comparable Data Access Control. *IEEE Transactions on Computers,* 64**,** 3430-3443.

WEI, J., WANG, S., ZHANG, L., ZHOU, A., SUN, Q., SHI, R. & YANG, F. Minimizing Data Transmission Latency by Bipartite Graph in MapReduce. 2015 IEEE International Conference on Cluster Computing, 8-11 Sept. 2015. 521-522.

WEI, W., DU, J., YU, T. & GU, X. SecureMR: A Service Integrity Assurance Framework for MapReduce. Computer Security Applications Conference, 2009. ACSAC '09. Annual, 7-11 Dec. 2009. 73-82.

WHITE, T. 2012. *Hadoop: The Definitive Guide Third Edition Edition* Inc. O'Reilly Media.

WIKI, H. 2017. Apache Hadoop.

WOHLMACHER, P. & PHAROW, P. Applications in health care using public-key certificates and attribute certificates. Computer Security Applications, 2000. ACSAC '00. 16th Annual Conference, Dec 2000. 128-137.

XU, L., KHOA, P. D., KIM, S. H., RO, W. W. & SHI, W. Another Look at Secure Big Data Processing: Formal Framework and a Potential Approach. Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on, June 27 2015-July 2 2015. 548-555.

XU, X. & ZHAO, X. A Framework for Privacy-Aware Computing on Hybrid Clouds with Mixed-Sensitivity Data. High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on, 24-26 Aug. 2015. 1344-1349.

XU, Y., CHEN, J. & PENG, H. Research of Electronic Patient Record Based on XML. International Conference on Management of e-Commerce and e-Government, 2009 (ICMECG '09), 16-19 Sept. 2009. 219-222.

XUAN-QUI, P. & EUI-NAM, H. Towards task scheduling in a cloud-fog computing system. 2016 18th

Asia-Pacific Network Operations and Management Symposium (APNOMS), 5-7 Oct. 2016. 1-4.

YANG, K., JIA, X. & REN, K. 2015. Secure and Verifiable Policy Update Outsourcing for Big Data Access Control in the Cloud. *IEEE Transactions on Parallel and Distributed Systems,* 26**,** 3461-3470.

YI, S., HAO, Z., QIN, Z. & LI, Q. Fog Computing: Platform and Applications. 2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb), 12-13 Nov. 2015. 73-78.

YI, S., LI, C. & LI, Q. 2015b. A Survey of Fog Computing: Concepts, Applications and Issues. *Proceedings of the 2015 Workshop on Mobile Big Data.* Hangzhou, China: ACM.

YU, W. D. & CHEKHANOVSKIY, M. A. An Electronic Health Record Content Protection System Using SmartCard and PMR. e-Health Networking, Application and Services, 2007 9th International Conference on, 19-22 June 2007. 11-18.

YUQI, M., CUIBO, Y., TAO, M., CHUNHONG, Z., WEI, Z. & XIAOHUA, Z. Dynamic Load Balancing with Multiple Hash Functions in Structured P2P Systems. Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on, 24-26 Sept. 2009. 1-4.

ZENG, D., GU, L., GUO, S., CHENG, Z. & YU, S. 2016. Joint Optimization of Task Scheduling and Image Placement in Fog Computing Supported Software-Defined Embedded System. *IEEE Transactions on Computers,* 65**,** 3702-3712.

ZHANG, C., CHANG, E. C. & YAP, R. H. C. Tagged-MapReduce: A General Framework for Secure Computing with Mixed-Sensitivity Data on Hybrid Clouds. Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on, 26-29 May 2014. 31-40.

ZHANG, O. Q., KIRCHBERG, M., KO, R. K. L. & LEE, B. S. How to Track Your Data: The Case for Cloud Computing Provenance. 2011 IEEE Third International Conference on Cloud Computing Technology and Science, Nov. 29 2011-Dec. 1 2011. 446-453.

ZHANG, Q., ZHANI, M. F., YANG, Y., BOUTABA, R. & WONG, B. 2015. PRISM: Fine-Grained Resource-Aware Scheduling for MapReduce. *IEEE Transactions on Cloud Computing,* 3**,** 182-194.