Surya Nepal
Mukaddim Pathan *Editors*

Security, Privacy and Trust in Cloud Systems

Springer

Security, Privacy and Trust in Cloud Systems

Surya Nepal · Mukaddim Pathan
Editors

Security, Privacy and Trust in Cloud Systems

Springer

Editors
Surya Nepal
CSIRO ICT Centre
Marsfield, NSW
Australia

Mukaddim Pathan
Telstra Corporation Limited
Melbourne, VIC
Australia

ISBN 978-3-642-38585-8 ISBN 978-3-642-38586-5 (eBook)
DOI 10.1007/978-3-642-38586-5
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013947067

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To Amrita, Ashraya, and Shruti, for their support and understanding during the book preparation

Surya

To Ziyuan, for her warmth and support that gave me strength towards completion of the book

Mukaddim

Preface

Cloud computing has emerged as a new paradigm for on-demand delivery of computing resources to consumers as utilities. It offers unlimited computing resources to its users in a pay-as-you-go model with a higher level of quality of service such as availability and reliability in a substantially reduced infrastructure cost. With such an offering, it is not surprising that businesses are considering moving their IT infrastructure to cloud. However, it has been widely reported in the surveys of CTOs and CIOs that they have a number of reservations about adapting cloud computing for their businesses, and the security, privacy and trust of cloud systems is at the top of their list. The recent news reported in media about the leakage of customers' personal data have exacerbated their concerns even more. With the emergence of social media, such events are spreading faster than ever before and the impact of breach of privacy of their customers could be catastrophic to businesses. Businesses have serious concerns on moving their services and data to a cloud environment. These concerns need to be addressed to realize the vision of delivering IT services as utilities.

The vision of delivering unlimited computing resources, e.g., compute, network, and storage, as utilities, as promised by the cloud computing paradigm, has made some of the tasks—that were impossible to achieve a few years back for small and medium size businesses—possible. Businesses can run sophisticated data analytics tools without investing a big amount on IT infrastructure. This has been one of the driving forces behind the emergence of the new research area, called "Big Data". One of the key challenges in big data is transforming the raw data available to a business into business value and strategic advantage. Better management and analysis of data will become the next frontier of innovation, competition, and productivity, and cloud has a big role to play in this area. For example, according to a McKinsey Global Institute study, a retailer exploiting the full potential of big data could increases its operating margin by more than 60 %. Efficient and effective use of big data could save more than $300 billion for US government in the healthcare sector alone. Therefore, there is a need for effective and efficient management and analysis of big data. Cloud computing has emerged as a choice of technology platform for big data. However, the lack of security and privacy of data in the cloud has been a major hurdle for businesses to utilize the full potential of cloud to unlock the business intelligence residing in their data.

In order to take the full advantage of enormous amount of business data using cloud, the issues related to security, privacy, and trust of data services need a careful attention. The foremost concern for businesses is that they have to relinquish the full control of their data to the cloud service providers without knowing whether there are adequate measures in place to protect their data. They also need to be aware of legal implications to their data. As the cloud enable migration of data across different jurisdictions, which laws are applicable to the data becomes an important factor to be considered while moving data to the cloud. As pointed out by a cloud service provider in a recent conference, the cloud computing inadvertently provided a playground for lawyers. Therefore, it is important to address legal aspects of data protection in the cloud.

Cloud computing introduces challenges to traditional approaches to protecting data including authentication and authorization. There is a need to develop a new way of authenticating users for cloud data services and defining access control. The implications of cloud computing paradigm to identity management and user authentication need to be further analyzed. Related to identity management is the issue for intercloud data migration. Unless there is a way of achieving seamless transition of data migration from one cloud provider to another, just like changing utility providers today, the security, privacy, and trust issues will continue to have implication beyond a single cloud provider.

Cryptographic approaches have been used to protect data where the data is encrypted both in motion and at rest so that they are never revealed to anyone other than data owners themselves. In such an approach, the data is encrypted before storing to cloud storage services and is never decrypted, while residing in the cloud. The data is retrieved into the trusted local environment before decryption. But the cloud introduces new challenges due to the cost of moving and processing big data. This means we need to look at the mechanisms of processing encrypted data in the cloud without compromising confidentiality. This demands privacy preserving analysis of big unstructured data as well as privacy preserving queries over relational databases. The privacy preserving querying and analysis of data enables to process data in the encrypted forms. A number of researchers have looked at the new form of encryption techniques, called homomorphic encryption. Developing effective and efficient fully homomorphic encryption techniques still remains as a challenging problem. In the coming years, we expect to see a reasonable progress made in this direction.

In the past few years, outsourcing firms have been increasingly used by businesses to provide their services to customers in cost-effective ways. The core strategy is to outsource certain aspects of a business process to skilled, but cost-effective, external service providers. The cloud computing paradigm needs to support this business model to be adapted successfully by enterprises in practice. Outsourcing requires multiple organizations working together to achieve a goal. Competing organizations may use the same outsourcing firms to perform a certain process within their businesses (such as billing). The cloud platform should

support the sharing of data and processes across different organizations while preserving the privacy of both data and processes. Not all processes can be outsourced. Some of the processes are going to be performed within organizations to preserve the competitive advantages of enterprises. How to support the sharing of data in cloud across collaborating organizations in such a way that competitive advantage of businesses and privacy of the data can be preserved. Meeting these two conflicting requirements is a challenge in itself.

Recent reports on cloud data services have indicated that data owners would like to know what is happening with their data. Who have accessed it? When it was accessed? Where it is stored? When the movement of the data occurred? How often the data is backed up? How many copies of the data are kept? The metadata about the data becomes as important as the data itself and sometime the size of the metadata becomes larger than the original data. A cloud data service should be able to answer all these questions with a clear separation of duties. This means the data management and activity logging components should work independently so that the data owner can trust the integrity of logged data. The answers to these questions can be found in the data accountability. How to standardize the data accountability service and implement it is as an integral part of cloud data service is an interesting and challenging problem.

Cyber attacks have been on the rise in recent times. The effect of cyber attacks in cloud is severe. For example, the denial of service attacks on cloud data services may not only disrupt the services and keep the genuine customers out of enterprise services, but also increase the costs due to the underlying pay-as-you-go model. Cloud service providers should be able to provide a "credit card" like security measure to their customers and should be able to refund all costs incurred through cyber attacks. However, there is no way cloud service providers can vouch that a service request is genuine or the result of a cyber attack. Thus, the cloud service providers should not only be able to detect and prevent the cyber attacks on the services deployed on their clouds, but also should establish clear guidelines on how to resolve the disputes arising from such attacks. It is thus clear that some of the challenges related to cloud security, privacy, and trust go beyond the technological solutions. We need to look at the social and legal aspects of the cloud data services.

Another interesting debate around cyber attacks is whether the cloud data service is more attractive to cyber attacks than an individual enterprise data service. Some believe that cloud data services are more attractive for attackers as they know they can unlock a large number of valuable information if the attacks on cloud services are successful. They thus believe that the data become prone to more attacks when it is kept in the cloud. An alternative thought is that cloud service providers can probably have a large number of security experts working on preventing cyber attacks on enterprise data than any single enterprise could afford at any time. They believe that the cloud providers are better equipped to protect

enterprise data than enterprises themselves. The reality may lie in between these two opposite views. Time can only tell which view is right!

In recent times, we have seen an increasing number of cloud service providers that operate within a single jurisdiction or across multiple jurisdictions. The choice of providers is good for consumers, but the emergence of a large number of cloud service providers poses a number of challenges from the point of view of trust. How do you know which provider is best for you? Reputation based on past experience has been used as a mechanism of addressing the issue of trust. Although this approach has a foundation on economics, marketing, social, and behavior sciences, we believe that we need to look at the holistic solutions that take into account of the technological and social aspects of trust.

In the past few years, there have been an increasing number of efforts toward developing cloud standards by national and international standard bodies. Such efforts could go a long way to address some of the concerns about security, privacy, and trust in cloud systems. However, the success relies on the adaptation of such standards in practice.

In this book, we have outlined the problems in developing secure, private, and trusted cloud systems from different points of views. Researchers, students, and practitioners need to understand the complexity of developing such systems from the point of views of standards, technologies, tools, economics, and social and behavioral sciences. As the cloud computing is a new and evolving paradigm, the solutions are being researched and still emerging. Therefore, this book is intended to pose key research challenges and some emerging solutions along with future trends.

Overview and Scope of the Book

This book, entitled "Security, Privacy and Trust in Cloud Systems" presents cloud security fundamentals and related technologies to-date, with a comprehensive coverage of evolution, current landscape, and future roadmap. It provides a smooth organization with introductory, advanced, and specialist content, i.e., from basics of security, privacy, and trust in cloud systems, to advanced cryptographic techniques, case studies covering both social and technological aspects, and advanced platforms. The book builds on academic and industrial research and developments, and case studies that are being carried out at many different institutions around the world. In addition, the book identifies potential research directions and technologies that drive future innovations. We expect the book to serve as a valuable reference for larger audience such as systems architects, practitioners, product developers, researchers, and graduate level students.

Organization

This book will enable readers to understand the basics, identify the underlying technology, summarize their knowledge on concepts, ideas, principles, and various paradigms which span on Cloud security, privacy, and trust domains. The book is organized into three parts, namely, Part I: "Cloud Security"; Part II: "Cloud Privacy and Trust"; and Part III: "Case Studies: Cloud Security, Privacy, and Trust". Specifically, the topics of the book are the following:

- Cloud security fundamentals
- Secure information sharing and data protection in the cloud
- Cloud security architecture and protocol
- Autonomic security in cloud systems
- Cryptography and crypto-protocols for cloud systems
- QoS-based trust model and QoS monitoring mechanism
- Enterprise cloud security case study
- Open research issues in cloud security and future roadmap

Part I of the book focuses on the basic ideas, techniques, and current practices related to "Cloud Security". "Cloud Security: State of the Art", by Soares et al., presents a comprehensive analysis of the state of the art on cloud security issues. In addition to presenting the key concepts on cloud security, this chapter discusses the most prominent security issues tackled in literature, surveying vulnerabilities, gaps, threats, attacks, and risks in cloud environment. Thilakanathan et al., in "Secure Data Sharing in the Cloud", provide a review on methods of achieving secure and efficient data sharing in the cloud. The presented research outcome is particularly useful for secure sharing of real-world critical data from the business, government and/or medical domains. In "Adaptive Security Management in SaaS Applications", Almorsy et al. discuss on a security management framework to deliver autonomic security where the security level, enforced on the cloud platform and cloud services, automatically adapt to match the current security risks and threats. Addressing the limitations of using the virtualization technology in cloud systems, Caron et al. in "Smart Resource Allocation to Improve Cloud Security" present a resource allocation technique to improve cloud security. They introduce a way for users to express security requirements and demonstrate how a cloud service provider can address those requirements. Building on cryptographic mechanisms to guarantee security properties such as data confidentiality and integrity, "Mandatory Access Protection within Cloud Systems" by Bousquet et al. describes mandatory access protection in cloud systems.

Part II of this book highlights technologies to ensure "Cloud Privacy and Trust". Tormo et al. in "Identity Management in Cloud Systems" present, analyze, and compare current identity management standards, technologies, and solutions from the cloud perspective, taking into account their features and requirements. They provide a set of recommendations to be taken into consideration when

designing and deploying any identity-based service in a cloud environment. It is followed by a "Data Accountability in Cloud Systems" on data accountability, by Ko, reviewing definitions, existing techniques and standards in the area of data accountability in cloud systems. Based on MapReduce, "Privacy Preservation over Big Data in Cloud Systems" by Zhang et al. discusses on data privacy preservation and data quality in the cloud under given privacy requirements. This chapter demonstrates a prototype privacy-preserving framework to anonymize large-scale data sets in the cloud. In "Securing Outsourced Databases in the Cloud", Liu talks about privacy of database services in cloud systems. He presents an indexing scheme and an associated encryption scheme to encrypt databases and query encrypted databases in the cloud. This part of the book is ended with "Trust Model for Cloud Systems with Self Variance Evaluation", by Wang et al., presenting reputation-based trust models for cloud systems. They introduce a general trust model to get a more comprehensive and robust reputation evaluation.

Part III, the final part of the book, consists of a handful of representative "Case Studies on Cloud Security, Privacy, and Trust". In "Cryptographic Role-Based Access Control for Secure Cloud Data Storage Systems", Zhou et al. describe access control models and the use of cryptographic techniques for secure cloud data storage. In their case study, authors cover a scheme which integrates cryptographic techniques with role-based access control and show how the scheme can be used to secure data storage in the cloud. "Accountability-Based Compliance Control of Collaborative Business Processes in Cloud Systems" by Yao et al. presents a case study on accountability-based compliance control of collaborative business process in cloud systems. Authors base their case study on Amazon EC2 using a loan application business process. A case study on 'Reputation as a Service' is presented next. In this chapter, Itani et al. demonstrate a secure and accountable reputation system for ranking cloud service providers. In "Combating Cyber Attacks in Cloud Systems Using Machine Learning", Khorshed et al. present a machine-learning approach to combat cyber attacks in cloud systems. The final chapter of the book, by Kertesz and Varadi, cover the legal aspects of data protection in cloud systems. They examine use cases and assess them against evaluation criteria derived from the relevant cloud computing law for the data processing of end-user details and materials, including roles and responsibilities necessary for legal compliance.

Acknowledgments

The book came into light due to the direct and indirect involvement of many researchers, academics, and industry practitioners. We acknowledge and thank the contributing authors, research institutions, and companies whose papers, reports, articles, notes, Web sites, study materials have been referred to in this book. We offer our special appreciation to Springer and its publishing editor, Dr. Christoph Baumann, for helping us to bring this book out in a quick time.

Prior technical sources are acknowledged citing them at appropriate places in the book. In case of any errors, we would like to receive feedback so that it could be taken into consideration in the next edition.

We hope that this book will serve as a valuable text for students, especially at graduate level and a reference for researchers and practitioners working in the Cloud security, privacy, and trust domains.

Surya Nepal
Mukaddim Pathan

Contents

Contributors

A. B. M. Shawkat Ali Faculty of Arts, Business, Informatics and Education, Central Queensland University, Bruce Highway, North Rockhampton, QLD, 4702, Australia

Mohamed Almorsy Faculty of Information and Communication Technologies, Swinburne University of Technology, John Street, Hawthorn, VIC, 3122, Australia

Mathieu Blanc CEA, DAM, DIF, Arpajon 91297, France, e-mail: first.last@cea.fr

Aline Bousquet Laboratoire d'Informatique Fondamentale d'Orléans, ENSI de Bourges, 88 bd Lahitolle, Bourges, 18020, France, e-mail: first.last@ensi-bourges.fr

Jeremy Briffaut Laboratoire d'Informatique Fondamentale d'Orléans, ENSI de Bourges, 88 bd Lahitolle, Bourges 18020, France, e-mail: first.last@ensi-bourges.fr

Rafael A. Calvo Department of Electrical Engineering, University of Sydney, Sydney, NSW 2006, Australia

Eddy Caron LIP Laboratory, UMR CNRS - ENS Lyon - INRIA - UCB, Université de Lyon, Lyon, 5668, France, e-mail: eddy.caron@ens-lyon.fr

Ali Chehab Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon, e-mail: chehab@aub.edu.lb

Jinjun Chen Faculty of Engineering and IT, University of Technology Sydney, Sydney, Australia, e-mail: jinjun.chen@gmail.com

Shiping Chen CSIRO ICT Center, Cnr Vimiera and Pembroke Rodas, Marsfield, NSW, 2122, Australia, e-mail: Shiping.Chen@csiro.au

Laurent Clevy Alcatel Lucent Bell Labs, Route de Villejust, Nozay 91620, France, e-mail: first.last@alcatel-lucent.com

Frederic Desprez INRIA Grenoble Rhône-Alpes ZIRST Montbonnot, 655 avenue de l'Europe, Saint Ismier Cedex 38334, France, e-mail: frederic.desprez@inria.fr

Diogo A. B. Fernandes Department of Computer science, Instituto de Telecomunicações, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal, e-mail: dfernandes@penhas.di.ubi.pt

Mário M. Freire Department of Computer science, Instituto de Telecomunicações, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal, e-mail: mario@di.ubi.pt

Cesar Ghali Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon, e-mail: csg04@aub.edu.lb

João V. Gomes Department of Computer science, Instituto de Telecomunicações, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal, e-mail: jgomes@penhas.di.ubi.pt

Damien Gros CEA, DAM, DIF, Arpajon 91297, France, e-mail: first.last@cea.fr

John Grundy Faculty of Information and Communication Technologies, Swinburne University of Technology, John Street, Hawthorn, VIC, 3122, Australia

Michael Hitchens Information and Networked Systems Security Research, Department of Computing, Macquarie University, Macquarie Park, NSW, Australia

Xiaofeng Hu School of Computer, National University of Defense Technology, Changsha, 410073, Hunan, China

Amani Ibrahim Faculty of Information and Communication Technologies, Swinburne University of Technology, John Street, Hawthorn, VIC, 3122, Australia

Pedro R. M. Inácio Department of Computer science, Instituto de Telecomunicações, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal, e-mail: inacio@di.ubi.pt

Wassim Itani Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon, e-mail: wgi01@aub.edu.lb

Ayman Kayssi Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon, e-mail: ayman@aub.edu.lb

Attila Kertesza MTA SZTAKI Computer and Automation Research Institute, Budapest, P.O. Box 63, 1518, Hungary, e-mail: kertesz.attila@sztaki.mta.hu

Md Tanzim Khorshed Faculty of Arts, Business, Informatics and Education, Central Queensland University, Bruce Highway, North Rockhampton, QLD, 4702, Australia

Arnaud Lefray LIP Laboratory, UMR CNRS - ENS Lyon - INRIA - UCB, Université de Lyon, Lyon, 5668, France, e-mail: first.last@ens-lyon.fr

David Levy School of Electrical and Information Engineering, University of Sydney, Sydney, NSW 2006, Australia, e-mail: dlevy@ee.usyd.edu.au

Chang Liu Faculty of Engineering and IT, University of Technology Sydney, Sydney, Australia, e-mail: changliu.it@gmail.com

Dongxi Liu CSIRO ICT Center, Cnr Vimiera and Pembroke Rodas, Marsfield, NSW, 2122, Australia, e-mail: dongxi.liu@csiro.au

Félix Gómez Mármol NEC Laboratories Europe, Kurfürsten-Anlage, 36, 69115, Heidelberg, Germany, e-mail: felix.gomez-marmol@neclab.eu

Surya Nepal CSIRO ICT Center, Cnr Vimiera and Pembroke Rodas, Marsfield, NSW, 2122, Australia, e-mail: Surya.Nepal@csiro.au

Gregorio Martínez Pérez Departamento de Ingeniería, de la Información y las Comunicaciones, Facultad de Informática, Universi-dad de Murcia, 30100, Murcia, Spain, e-mail: gregorio@um.es

Jonathan Rouzaud-Cornabas LIP Laboratory, UMR CNRS - ENS Lyon - INRIA - UCB, Université de Lyon, Lyon, 5668, France, e-mail: jonathan.rouzaud-cornabas@ens-lyon.fr

K. L. Ko Ryan Department of Computer Science, The University of Waikato, Hamilton, New Zealand, e-mail: ryan@waikato.ac.nz

Liliana F. B. Soares Department of Computer science, Instituto de Telecomunicações, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal, e-mail: lsoares@penhas.di.ubi.pt

Jinshu Su School of Computer, National University of Defense Technology, Changsha, 410073 Hunan, China

Danan Thilakanathan Department of Electrical Engineering, The University of Sydney, Sydney, NSW 2006, Australia

Christian Toinard Laboratoire d'Informatique Fondamentale d'Orléans, ENSI de Bourges, 88 bd Lahitolle, Bourges 18020, France, e-mail: first.last@ensi-bourges.fr

Ginés Dólera Tormo NEC Laboratories Europe, Kurfürsten-Anlage, 36, 69115, Heidelberg, Germany, e-mail: gines.dolera@neclab.eu

Vijay Varadharajan Information and Networked Systems Security Research, Department of Computing, Macquarie University, Macquarie Park, NSW, Australia, e-mail: vijay@science.mq.edu.au

Szilvia Varadib Department of International and European Law, University of Szeged, Rakoczi ter 1, Szeged, 6722, Hungary, e-mail: varadiszilvia@juris.u-szeged.hu

Benjamin Venelle Alcatel Lucent Bell Labs, Route de Villejust, Nozay 91620, France, e-mail: first.last@alcatel-lucent.com

Xiaofeng Wang School of Computer, National University of Defense Technology, Changsha, 410073, Hunan, China, e-mail: xf_wang@nudt.edu.cn

Saleh A. Wasimi Faculty of Arts, Business, Informatics and Education, Central Queensland University, Bruce Highway, North Rockhampton, QLD, 4702, Australia

Chunqing Wu School of Computer, National University of Defense Technology, Changsha, 410073, Hunan, China

Chi Yang Faculty of Engineering and IT, University of Technology Sydney, Sydney, Australia, e-mail: chiyangit@gmail.com

Jinhui Yao School of Electrical and Information Engineering, University of Sydney, Sydney, NSW, 2006, Australia, e-mail: jinhui.yao@gmail.com

Xuyun Zhang Faculty of Engineering and IT, University of Technology Sydney, Sydney, Australia, e-mail: xyzhanggz@gmail.com

Huan Zhou School of Computer, National University of Defense Technology, Changsha, 410073, Hunan, China

Lan Zhou Information and Networked Systems Security Research, Department of Computing, Macquarie University, Macquarie Park, NSW, Australia

Part I
Cloud Security

Cloud Security: State of the Art

Liliana F. B. Soares, Diogo A. B. Fernandes, João V. Gomes, Mário M. Freire and Pedro R. M. Inácio

1 Introduction

Throughout the end of the first half and during the second half of the past century, advances in technology allowed scientists to develop computer systems. In the beginning, mostly between the forties and the sixties, single computers would fill large rooms with electronics that would consume as much power as several hundreds of modern desktop computers. However, a diversity of revolutionary approaches were invented throughout the time, gradually replacing those large, and expensive, computer rooms with smaller, more powerful computers, being able to hold many of them. This allowed computer systems and networks to emerge, like standard Ethernet networks that still persist today. Distributed systems, one of those approaches, arose as a means to aggregate computational assets with the main goal of supporting highly intensive and hardware-demanding tasks, which can consume several processing resources simultaneously and last for a long time. Examples of such tasks include molecular modeling, scientific simulations and weather forecasts.

L. F. B. Soares (✉) · D. A. B. Fernandes · J. V. Gomes ·
M. M. Freire · P. R. M. Inácio
Instituto de Telecomunicações, Department of Computer Science,
University of Beira Interior Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal
e-mail: lsoares@penhas.di.ubi.pt

D. A. B. Fernandes
e-mail: dfernandes@penhas.di.ubi.pt

J. V. Gomes
e-mail: jgomes@di.ubi.pt

M. M. Freire
e-mail: mario@di.ubi.pt

P. R. M. Inácio
e-mail: inacio@di.ubi.pt

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_1,

1.1 Computer Clusters, Grids and Clouds

Two of the most typical distributed systems are computer clusters and computer grids. As the name suggests, computer clusters consist of coupled hardware resources that can function standalone. These resources are put on a centralized point, usually a large, cooled, well protected room, leading to expensive approaches. Clusters enhance computational, memory and storage capabilities of computer systems, therefore requiring specially designed middleware to support communications between each hardware node. In addition, this type of architecture for computer systems requires bandwidth-wise network links to support large amounts of intracommunications network traffic. However, nowadays, open-source software enables the use of processing units and memory from desktop computers to dispatch distributed, potentially parallel tasks by nominating a master node that manages the cluster. This approach can reduce implementation costs to companies rather than spending big on supercomputers or expensive clusters. Grid systems, on the other hand, are not as coupled as clusters and dwell over large scattered and heterogeneous networks in a decentralized fashion. The most prominent example is the Internet, over which grids are deployed as overlay networks. In fact, the most well-known form of grids is the one of using typical desktop and home computers as end slave computation nodes. This approach, however, brings obstacles, such as increased management complexity, task assignment, and results collecting and gathering.

Based on the cluster and grid computing paradigms [35, 90], cloud computing has emerged in the last few years as the latest approach with the purpose of computing as utility. In fact, grid computing provides the backbone and supporting infrastructure for clouds [35]. It was around 2008 that the cloud computing subject started gaining serious interest among the industry and the academia. Additionally, to provide a better perspective, the *utility computing* term was first invoked as early as 1965 [23]. It refers to computational resources efficiently wrapped as services, being more a business model rather than a computing paradigm today, and matching the cloud business model. Analogously to clusters, clouds are composed of umpteen servers placed in large, cooled, well protected rooms under the same subnet. Facilities that host clouds are nowadays called data centers, which require being physically and logically segregated from malicious intrusions because clouds usually hold large amounts of sensitive data belonging to customers. The main innovative side of clouds is how Information Technologies (IT) are put together along with virtualization techniques, providing web service-based and on-demand capabilities accessible over the Internet. To this end, a pay-per-use business model is implemented, meaning that computational, storage or networking resources rented by customers are strictly billed according to their needs, that is, time of usage, assets required, load and security measures. Cloud systems are both centralized and decentralized, allowing public access to their resources via web technologies. Hence, a centralized and distributed resource handling approach is applied, providing multi-tenant and Service-Oriented Architecture (SOA) capabilities, similarly to grids.

1.2 Cloud Security

The National Institute of Standards and Technology (NIST) view of cloud is summarized in version 3 of the security guidance entitled as *Security Guidance for Critical Areas of Focus in Cloud Computing* [24], published by the Cloud Security Alliance (CSA), an organization aiming at promoting the use of best practices in cloud security. In the document, the cloud deployment models, the cloud service delivery models or service models, and the five essential characteristics of clouds are described. The cloud deployment models include public, private, hybrid, and community clouds, and Virtual Private Clouds (VPCs). The service models are divided into Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). Finally, the characteristics are broad network access, rapid elasticity, measured service, on-demand self-service, and resource pooling.

The NIST [74] mentions security, interoperability and portability as major barriers for a broader adoption of cloud solutions. There are just a few standards supporting clouds, which translates into the *lock-in* issue faced by customers. In other words, when a customer decides for a certain cloud provider, the data stored on the cloud cannot yet migrate to clouds of other providers. Nonetheless, *interclouds* [11, 88], a term referring to a network of clouds, a place of cloud computing, interoperability, ubiquitous and utility computing, and data storage, would overcome this issue and free data movement among clouds belonging to different providers. Clouds increased the complexity of many previous security issues and introduced new ones, being yet a risky business. To demonstrate how security is one of the most mind changing factor (if not the most important), in 2009, the International Data Corporation (IDC), a market research and analysis firm, has harvested opinions among company Chief Information Officers (CIOs) on the most concerning cloud obstacles. The survey [46] was concluded with the security topic ranking first with 87.5 % of the votes, 12.9 % more than the study on the previous year [47], in which security also led with 74.6 %. The results of the 2009 study are illustrated in Fig. 1. This perspective concerning cloud security is shared with the *Top predictions for IT Organizations and Users for 2012 and Beyond* report [38], property of Gartner, a technology research and advisory company. Because of security, people hesitate to fully move their business into clouds, slowing down their propagation, as the research and the industry are focused on patching security issues, rather than exploring their potentialities. In addition, *"my sensitive corporate data will never be in the cloud"* is a statement that has been heard multiple times [3], further pointing out how critical security is. Because clouds outsource businesses of many customers, which includes potentially sensitive data and applications, they pose as attractive attack targets for cybernetic pirates or *malicious insiders*. Thus, there is much at stake in the cloud business, being *data disclosure*, *data loss* and *financial loss* major risk scenarios. Clouds offer many appealing features and advantages, but until some of its risks are better understood, major players might hold back [106]. This means that cloud systems are a risky business, not only to customers, but also to the providers investments.

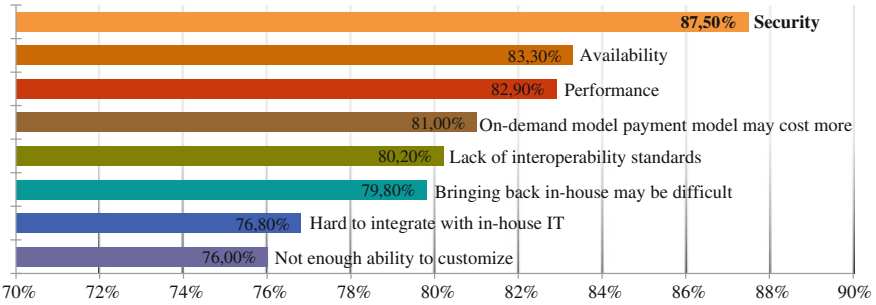


Fig. 1 Challenges and issues of the cloud model according to IDC and corresponding results from the cloud user survey (adapted from [46])

1.3 Organization

The previous paragraphs enlightened on the differences between the cloud, the cluster and the grid computing paradigms, highlighting the most prominent characteristics of these distributed systems. Essentially, the importance of the cloud security topic is highlighted in the discussion, underlining how critical it is to address security issues. To this end, this chapter discusses the most prominent security issues tackled in the literature, surveying vulnerabilities, gaps, threats, attacks, and risks on cloud environments. Such terms are emphasized throughout the text as to better distinguish each issue. Additionally, concepts of both cloud and cloud security subjects are described in order to facilitate the understanding of this chapter. Foremost, the chapter presents a comprehensive analysis of the state-of-the-art on cloud security issues.

The remaining of this chapter is organized as follows. Section 2 delivers an insight on the works that are most similar to the one presented herein. Section 3 overviews some general features of clouds and key concepts of cloud security. Subsequently, in Sect. 4, a discussion of the published literature on security issues in cloud environments is presented. A synthesis of the chapter containing a timeframe overview of what was discussed is included in Sect. 5. The chapter ends with the main conclusions in Sect. 6.

2 Related Work

Cloud security has been in vogue on the literature and industry for a while now. Various international conferences have focused on this subject alone, such as the Association for Computer Machinery (ACM) *Workshop on Cloud ComputingSecurity*, the *International Conference on Cloud Security Management* and the only European conference on the subject, *SecureCloud*, which already numbers up to three editions. As a result, several scientific contributions have been published, not only

in conferences proceedings, but also in international journals and magazines. Thus, there are a few works surveying this area of knowledge that are worthy to describe herein.

The study in [115] surveyed security and privacy concerns of cloud providers. Firstly, the security topic was discussed while having in mind availability, confidentiality, data integrity, control and audit properties, concluding that these do not meet current concerns. Secondly, the privacy topic was discussed with focus on out-of-date privacy acts that fail to protect information from being disclosed to the government and third-parties. In addition, the multi-location issue of clouds is also included in the study, stating that knowing in which country the data will be kept is a prerequisite for customers, in order to find by which laws the data is governed. It was claimed that new strategies should be put forward to achieve the five aforementioned properties and that privacy acts should be changed accordingly.

Again, in [116], the confidentiality, privacy, integrity and availability aspects in clouds were placed under observation. Various issues were discussed so as to present a synthesis of security requirements and threats with respect to the service models. The study ended with the proposal of a trusted third-party solution to eradicate security threats of confidentiality, integrity, authenticity and availability. The solution combined Public Key Infrastructures (PKIs), the Lightweight Directory Access Protocol (LDAP) and Single Sign-On (SSO) with a top-down fashion of the trust tree. The study was concluded with the premise that cloud benefits will outnumber its shortcomings.

Another survey targeting security issues on the cloud service models was presented in [95]. Each model was singularly studied, pointing out some of the most significant security vulnerabilities, threats and risks. It should be noted that the SaaS model was the one with the majority of the issues. An overview of current solutions discussed in the literature is presented afterwards. Yet again, the study was concluded saying that proper solutions should be designed and deployed to allow the cloud industry expand further.

The security and privacy topics were again discussed in [111]. A comprehensive and technical review of security issues was included in the study, in which confidentiality, integrity, availability, accountability and privacy-preservability were identified as the most significant attributes. To each property, a few security issues are described, followed by the corresponding defense solutions. In the end, it was claimed that the study might help shaping the future research directions in the security and privacy contexts in terms of clouds.

In [88], various security challenges were enumerated as key topics in cloud security. Those challenges related with resource allocation, system monitoring and logging, computer forensics, virtualization, multi-tenancy, authentication and authorization, availability, and cloud standards. The study particularly focused afterwards on introducing the Service Level Agreements (SLAs), trust, and accountability topics with regard to cloud security. Issues and solutions were dually discussed throughout the study.

The previous works defined the basis of this chapter by providing materials to review the state-of-the-art on the subject. Nonetheless, the review presented in

this chapter contains a wider analysis when compared to those studies, allowing to construct a broader taxonomy for cloud security issues, leaving aside a deeper analysis of solutions for such issues. As commonly seen in other works, including the ones above, the chapter also discusses basic cloud and cloud security concepts in order to ease its understanding.

3 Security-Related Concepts in Cloud Environments

This section defines and describes some basic concepts on cloud and cloud computing, together with key notions on cloud security. The discussion complements some ideas already included in the Introduction section. Thus, it prepares the reader for the remaining part of the chapter.

3.1 Cloud Service Models

The increasing connection demands of the population have triggered the development of Web 2.0 and a new class of services. Cloud systems have adopted a standard three-model architecture, each one containing fundamental parts to support the cloud unique operation. The architecture is composed of IaaS, PaaS and SaaS, sorted upwardly.

The bottom model, IaaS, revolutionized how businessmen invest in IT infrastructures. Instead of spending large amounts of budget in hardware and technical crews to assemble and manage the materials, IaaS providers offer reliable virtual servers on the minute. Amazon Web Services (AWS) is a real example of such providers. A pay-per-use approach is employed in this model, meaning that customers only pay for what they require. Additionally, it abstracts businesses from the scalability, management and provisioning of the infrastructure, allowing them to focus on promoting their applications. The IaaS model provides basic security, including perimeter firewall and load balancing, as long as the VMM is not compromised. The provider should, at least, ensure security up to the VMM, which includes environmental, physical and virtualization security. IaaS ultimately suffers from the *data locality* and *co-location* issues.

The middleware model, PaaS, delivers abilities for customers to develop their own cloud applications by providing platforms and frameworks. Consequently, this model becomes more extensible than SaaS by providing a set of customer-ready features, therefore administrating greater flexibility in security. Thus, *unsafe* Integrated Development Environments (IDEs) and Application Programming Interfaces (APIs) may constitute vulnerability points. Furthermore, because the underlying elements of SOA applications are obscured by the architecture, cybernetic pirates are most likely to attack visible code written by users. A set of coding metrics should be put forward to evaluate the quality of code written by those users.

The top model, SaaS, allows applications to be remotely deployed and hosted on clouds, delivering on-demand capabilities in the form of services that can be accessed via the Internet. This model improves operational efficiency and also reduces costs to customers, similarly to IaaS. It is rapidly becoming prevalent in the cloud business as it is rapidly meeting the requirements of IT companies. However, several security issues are raised by this model, mostly related with data storage, thus making customers uncomfortable in adopting SaaS solutions. Cloud providers must assure data isolation, confidentiality and privacy, meaning that users should not access nor understand data from other users. Nonetheless, from the customer viewpoint, it is hard to tell whether or not the data is well secured, and that applications are available at all times. Furthermore, it is harder to preserve or enhance security that was formerly provided by previous systems.

Although the three service models make up the foundations for the cloud operation, the IT industry is assisting to a mutation; it is converging to Anything-as-a-Service (XaaS). Because clouds are virtually infinite and can, therefore, support anything, or everything, in the form of services, the XaaS approach is able to deliver a wide range of services, including large resources and specific requirements.

3.2 Data Center Facilities Security

As previously said, clouds are computer systems put on specially designed rooms to hold a massive number of servers and network links. Cooling is an active research area in which many approaches are proposed and implemented with the purpose of producing efficient facilities. Protection is other topic of concern when mentioning such facilities. Rooms in such infrastructures hold many expensive network, computation and storage devices, and private data, therefore requiring proper security. In fact, entrepreneurs build data centers while having in mind many geological and environmental aspects, such as location, temperature, humidity, and earthquakes. Other political, governmental, and energy-saving aspects are also taken into consideration. For instance, grid redundancy [22] is a technique used to assure power continuity to devices, by tolerating loss of any power supply or a single alternating current power source. The goal is to provide the most possible reliable facilities to achieve high availability [19], reaching 99.99 % uptime in many cases, and being fully fault-tolerant. Hence, many data centers achieve the tier 4 level, which is the highest level defining the quality of data centers, being the lowest tier 1.

Physical security is established on-site in the data center facilities. If this prerequisite would not be fulfilled, other security measures would be unnecessary. For example, a security center managing video cameras, security sensors, personnel entrances and access to resources may be the most adopted approach. All this to prevent break-ins and other physical violations. Nonetheless, physical access to the rooms holding equipments should be restricted and only exclusive personnel with security clearances should go in to perform managing operations.

Flooding attacks, *hardware interruption*, *theft*, *modification*, *infrastructure misuse* and *natural disasters* are amongst the main issues of data center facilities [116].

Clouds contain service-driven networks, Storage Area Networks (SANs), and computational and storage-related hardware, which should be dully protected by resorting to firewalls and Intrusion Detection Systems (IDSes), like in standard networks. This approach would enable the analysis of network traffic and the detection or prevention of malicious intrusion attempts [62, 67]. Various IDS solutions have been provided [27, 61, 84]. It is recommended to deploy both IDS and Intrusion Prevention System (IPS) in clouds in order to achieve the desired security level [70]. Honeypots should also be considered, so as to divert attackers attentions [93]. Nonetheless, it should be paid some attention to the trade-off between security and performance [78], because too many security deployments may cause disruptions. Amazon Elastic Compute Cloud (EC2), for example, provides a firewall solution to each customer. By default, it is configured in deny mode, and customers must configure ports to allow incoming traffic for hosted services. The firewall has the ability of restricting traffic by protocol, port, and Internet Protocol (IP) address [1, 8].

3.3 Cloud Stakeholders

The players intervening in the cloud business define how the infrastructure is used and managed. In a simplified way, clouds have virtualized platforms that abstract the underlying hardware, and have services running on top of those platforms. Cloud providers own data center facilities and, therefore, have the responsibility of managing the facilities and the hardware resources in them. Service providers are another, but optional, stakeholder that can rent cloud resources to a cloud provider. In turn, service providers can deliver computational, storage and networking capabilities in the form of services to cloud customers. At all times, SLAs are negotiated in order to define the terms of service and what the cloud customer requires. Ideally, the optimal SLA should cover all critical security requirements. Traditionally, however, the extent of SLAs implemented in the industry does not fully include confidentiality and integrity aspects [88], mainly due to the challenges related with storage outsourcing. End users, which are also part of the model, are the ones that ultimately enjoy the services. This model is schematized in Fig. 2, where two distinct service providers

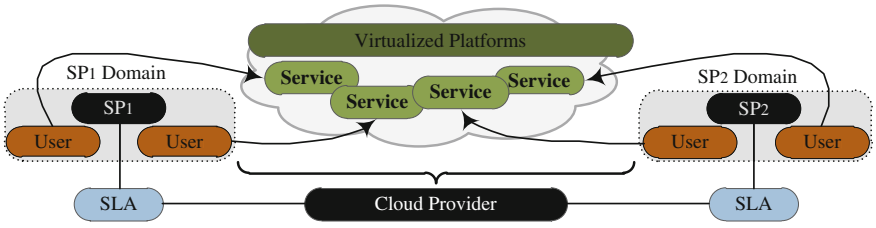


Fig. 2 Cloud stakeholders model adapted from [64, 115]. SP stands for service provider

hosting their services on the same cloud are illustrated. A noteworthy aspect is that, while cloud customers are responsible for application-level security, providers are responsible for physical and logical security. Intermediate layers of the cloud stack are shared between one another. Cloud customers may outsource their responsibility in security aspects to third-parties, who sell specialized security services.

3.4 Important Concepts in Cloud Security

As clouds rely on virtualization techniques, it is important to identify and describe which elements provide the backbone for virtualization. Thanks to it, a multi-tenant ability is implemented in clouds, meaning that users access applications specially designed to run on cloud platforms. Therefore, it is also important to discuss cloud software with focus on security. Moreover, clouds hold massive amounts of data from cloud customers, which is the main reason why data outsourcing and data storage are critical concepts to discuss. Consequently, standardization is also an issue relevant to include in cloud storage discussions. Finally, trust is briefly discussed from the outsourcing business model standpoint. These concepts are analyzed below, providing the means to clarify and identify the source of some cloud vulnerabilities and threats.

3.4.1 Virtualization Elements

Virtualization itself, or Virtual Machine (VM), is the process of abstracting computer applications, services and Operating Systems (OSes) from the hardware on which they run [93]. Virtualization technologies are placed within the IaaS model. Virtualized OSes are called guest OSes or just guests. The benefits of virtualization include costs and downtime reduction, ease of management and administration, and scalability [14]. Notwithstanding, it brought many new problems intrinsic to its nature, which researchers and entrepreneurs have tried to patch. A VM image is a pre-built copy of the memory and storage contents of a particular VM. VM images can be easily cloned or moved to another location while keeping the integrity of its contents. This allows clouds to deliver highly available services, that is, it keeps VMs running on other physical resources if the previous resources were compromised or allocated for other operations or VMs. Hence, it is perceivable that VMs require a middleware layer to support such operations, which is done by the help of Virtual Machine Monitors (VMMs), usually called hypervisors, and cloud computing OSes. Examples of popular hypervisors are VMware Player, VirtualBox and Xen. Cloud computing OSes are similar to traditional OSes, but provide an additional set of virtualization functionalities, such as allocation and deallocation of VMs, dispatching and migration of processes, and setup interprocess communications, in order to support autonomous scaling and opportunistic deployment of SaaS applications [81]. This would all be great if no security issues arose. However, virtualization brings

abstraction in *data locality*, which means that cloud users cannot pin-point the exact physical location of their data, as VMs can be moved from one machine to another autonomously by the underlying layers. Furthermore, data leakage by exploring VM or hypervisor vulnerabilities is the main virtualization risk.

3.4.2 Multi-Tenancy

Multi-tenancy is a virtualization feature that, apart from clouds, it is also present in grid systems. It consists of multiple users, called tenants, sharing a common platform or a single instance of an application. In a public cloud scenario, various customers may share the same VMMs and physical resources, but each one accesses its own VM with inherent SLAs, different security configurations and billing policies.

3.4.3 Cloud Software

Although virtualization brings many new issues, issues already prevalent in software developing are transported to clouds. These type of issues scatter over the PaaS and SaaS models, bringing up vulnerabilities in APIs and IDEs, and web technologies, respectively. For instance, bad programming approaches in deploying cloud applications or common Cross-Site Scripting (XSS) attacks can exploit vulnerabilities in these models. Therefore, each service model contains its own problems, raising concerns in the cloud business model.

3.4.4 Data Outsourcing

Outsourcing is the process of contracting services from third-party entities. In the context of cloud systems, data outsourcing consists in renting storage services off of cloud providers to store data from the customer on-premises. This approach brings both capital expenditure (CapEx) and operational expenditure (OpEx) to customers. However, more importantly, brings physical separation between the customers and their data, an issue called *loss of control* [111]. This has been one of the main motivations feeding customers contingency about moving their businesses into clouds. To overcome this, providers must be trustworthy and guarantee secure computing and data storage.

3.4.5 Data Storage Security and Standardization

Mechanisms to ensure information security must be applied to data stored in cloud systems. Cryptography approaches must be employed to ensure classical security properties, that is, confidentiality and privacy, integrity, authentication and availability. To this end, cloud providers should provide trusted encryption schemas

and application-level protocols, as well as authentication mechanisms and integrity checking techniques to ensure that data was not tampered with. This implies the use of secure communication protocols and standards. Nonetheless, the latest, standardization, poses as one of the cloud main barriers, complicating interoperability and the development of *interclouds*. Furthermore, applying classical security techniques may be impractical in cloud systems due to the great amount of data stored in their servers. Thinking on hashing entire data sets provides a good example on the issue, since it would require abnormal computational and communication overhead. New mechanisms are nonetheless expected to be developed, such as homomorphic encryption [92] that enables processing encrypted data without being decrypted, ideal for public clouds. Data backups and restores are also essential for the correct functioning of clouds. To this end, providers usually supply geographic redundancy to data, meaning that data is copied to different geographical locations, usually to another data center of the same cloud provider.

3.4.6 Trust

Trust is a critical barrier that must be surpassed in the cloud business model. Firstly, cloud customers must trust in the cloud systems of providers that are going to store their data. Secondly, providers must trust customers with access to the services, that is, access to clouds, which translates into one of the cloud main security issues. Malicious users can conceal attacks by perpetuating them as apparently legitimate users, like the *co-location* attack. Consequently, SLAs must be well detailed in order to legally cover all possible atypical scenarios in case of unexpected consequences of misusing the cloud infrastructure for both the client or for third-parties. Another important aspect in the trust topic is the pro-activity of cloud users in terms of security. Consider that users use low secure passwords to authenticate in the cloud via web, such as the passwords most used throughout the cybernetic world as shown by SplashData, a company dedicated to address password concerns in IT, in the Worst Passwords of 2012 [94]. The study was compiled by using millions of passwords posted online by hackers and it was concluded that the word *password* is the most common password. Several distinct characters should be used in order to assemble enough entropy. Moreover, most employees share their passwords with a coworker, a friend, or a friend of a coworker even after receiving password training [101]. This is a major problem, not only in clouds, but to all Internet systems, as unnoticed intrusions can happen.

3.5 General and Cloud-Specific Issues

A *security issue* is a general term that includes several problems. Vulnerabilities, gaps, threats, attacks, and risks are adequate sub-terms that derive from the issue term. It is important to distinguish the difference in these commonly used terms in

the security field [26]. *Threat* is a situation that may exploit a *vulnerability*, which is a flaw or weakness of a system, as in *gap*. *Attack* is the act of exploiting a threat, while *risk* is the likelihood of a threat agent taking advantage of a vulnerability and corresponding business impact. Moreover, people on the cloud security field tend to misunderstand the difference between general and cloud-specific issues, as that difference is not crisp enough [41]. Cloud computing heavily builds on capabilities available through several core technologies, which include web applications, the cloud service models, virtualization IaaS offerings, and cryptography mechanisms. A cloud-specific security issue must be intrinsic to or prevalent in a core technology. It has to have its root cause in one of the five essential characteristics proposed by the NIST, it is provoked when innovations make tried-and-tested security controls difficult or impossible to implement, or it is prevalent in established state-of-the-art cloud offerings. Only cloud-specific issues are included in the chapter.

3.6 Categorization of Cloud Security Issues

Researchers tend to define their own taxonomy on cloud security issues as there is not yet a *de facto* standard to follow. The study described in [92] mentions four categories, namely *cloud infrastructure*, *platform and hosted code*, *data*, *access*, and *compliance*. Another study [59] organized security issues into a model with three main sections, named security categories, security dimensions and security in the service models. Security categories span from *cloud customers* to *providers*, which can also be complemented with government entities [15], while security dimensions include *domains*, *risks* and *threats*. For instance, the *isolation failure* threat is due to virtualization vulnerabilities, therefore being placed on the IaaS model, posing as an issue to both customers and providers. In this chapter, the assessment of existing security issues in cloud security is done with basis on the previously discussed studies. This chapter considers *software-related*; *data storage and computation*; *virtualization*; *networking, web and hardware resources*; *access*; and *trust* as the most direct, in terms of threat identification, and embracing security issues set of categories in the cloud context.

4 Main Cloud Security Issues

Due to the growth of the cloud in the industry, entrepreneurs have decided to adopt cloud services, in spite of being aware of its security issues. Thus, clouds attract attention from a potential dangerous community and other parties aiming to exploit security vulnerabilities, and to perhaps publicly disclose private information. Malicious activities are motivated by a wide panoply of reasons, namely personal benefit, glory or vendetta. Therefore, it is important to address such security issues, which

are thoroughly reviewed in this section. This review is supported by a comprehensive study of the state-of-the-art on the subject.

4.1 Security Issues Identified by Organizations

The cloud security topic concerns not only the research community, but also the industry. Various documents have been published with the intent of aiding the development of trustworthy cloud systems. Nonetheless, customers should get a security assessment from third-parties before committing to a cloud provider. The following works, described in chronological order, are considered pioneering works on the subject [59].

The Gartner published the security document entitled *Assessing the Security Risks of Cloud Computing* [37]. In this document, seven security risks are identified as critical aspects to be considered by cloud customers before committing to a provider. They are privileged user access, regulatory compliance, data location, data segregation, recovery, investigative support, and long-term viability.

The European Network and Information Security Agency (ENISA), an organization responsible for responding to cyber security issues in the European Union, provided the document entitled *Cloud Computing:Benefits, Risks and Recommendations* [33]. Eight cloud specific risks are considered as top risks also from the customer viewpoint. They are loss of governance, *lock-in*, *isolation failure*, compliance risks, management interface compromise, data protection, insecure of incomplete data deletion, and malicious insider.

The CSA published version 1 of the document *Top Threatsto Cloud Computing* [25]. Besides describing each top threat, real life examples and remediation directions are provided. In addition, to each threat, references to the domains of the following document are included, along with the service models affected, which are summarized in Table 1 except for the domains. As can be concluded from the analysis of the table, all threats affect the three service models, except for two threats, illustrating

Table 1 Top threats to cloud computing as described in CSA [25], plus the domains in which they are included and the service models they affect. A check mark ✓ means the threat affects the underlying model. A cross × means otherwise

Threat #	Name	IaaS	PaaS	SaaS
1	Abuse and Nefarious use of cloud computing	✓	✓	×
2	Insecure interfaces and APIs	✓	✓	✓
3	Malicious insiders	✓	✓	✓
4	Shared technology issues	✓	×	×
5	Data loss or leakage	✓	✓	✓
6	Account of service Hijacking	✓	✓	✓
7	Unknown risk profile	✓	✓	✓

Table 2 Summary of the main characteristics of the cloud deployment models, regarding Ownership (Organization (O), Third-Party (TP), or Both (B)), Management (O, TP, or B), Location (Off-site, On-site, or B), Cost (Low, Medium, or High), and Security (Low, Medium, or High)

Model	Ownership	Management	Location	Cost	Security
Public	TP	TP	Off-site	Low	Low
Private and community	O or TP	O or TP	On-site	High	High
Hybrid	O and TP	O and TP	On-site and Off-site	Medium	Medium
VPC	B	B	B	B	High

how important it is to address security issues. Threat #4 only affects IaaS because it is where shared virtualization resources are located.

The CSA also published version 3 of the report *Security Guidance for Critical Areas of Focus in Cloud Computing* [24]. Fourteen domains are identified in report, namely cloud computing architectural framework (domain #1), governance and enterprise risk management (domain #2), legal and electronic discovery (domain #3), compliance and audit (domain #4), information lifecycle management (domain #5), portability and interoperability (domain #6), traditional security, business continuity and disaster recovery (domain #7), data center operations (domain #8), incident response (domain #9), application security (domain #10), encryption and key management (domain #11), identity and access management (domain #12), virtualization (domain #13), and security as a service (domain #14).

4.2 Deployment Models and Service Delivery Models Security

The cloud provides different deployment models that have their own characteristics, advantages and disadvantages. Table 2 summarizes the characteristics of the public, private, community, hybrid, and VPC deployment models. While public clouds are cheaper, located on-site, owned and managed by third-party entities, private clouds are more expensive, as they require specialized technicians and hardware, are located on-site, usually behind corporate firewalls, and can be owned and managed either by the organizations itself or a third-party. Private clouds are therefore more secure than public clouds. Community clouds are a particular case of private approaches, has they are setup to support a common interest between several distinct owners. Hybrid clouds mix up both public and private models. Moreover, VPCs are also a particular case of private models, resembling Virtual Private Networks (VPNs) in the sense of a network that is built on top of another network, removing the concerns related with operating in shared or public environments.

A cloud customer should deliberate the security state of each model before committing to a specific one, in order to conduct a strategic evaluation and be aware of current issues. More specifically, an assessment from the business perceptive should be performed in terms of security requirements. To this end, Table 3 intersects the

Table 3 Service models versus the public, private and hybrid deployment models with respect to security requirements as described by [83]

Security requirements	Cloud deployment models								
	Public cloud			Private cloud			Hybrid cloud		
Identification & Authentication	✓	∗	✓	✓	∗	✓	∗	∗	✓
Authorization	✓	✓	✓	∗	∗	✓	∗	∗	✓
Confidentiality	∗	∗	✓	∗	✓	✓	∗	∗	✓
Integrity	✓	∗	✓	∗	✓	✓	✓	✓	✓
Non-repudiation	∗	∗	✓	∗	∗	✓	∗	∗	∗
Availability	✓	✓	∗	✓	✓	✓	∗	∗	∗
	IaaS	PaaS	SaaS	IaaS	PaaS	SaaS	IaaS	PaaS	SaaS
	Cloud service models								

A check mark (✓) means an obligatory requirement in that specific service model and in the underlying deployment model, while an asterisk (∗) means optional

cloud deployment and service models with respect to six security requirements [83]. As can be concluded from the analysis of the table, authorization requirements on the three service models are mandatory in public clouds, so as to prevent unauthorized access to assets. The hybrid model is less demanding in terms of security requirements than both public and private models, as it is more secure. Amongst the three deployment models, integrity properties are much desired, pointing out the interest in checking data correctness and if it was tampered with or corrupted. Moreover, SaaS is the service model with more security requirements throughout all underlying deployment models. The study in [95] corroborates this trend, showing that a great deal of security issues is included in the SaaS, when compared to the remaining models. Particular issues arise in public clouds as specific pieces of data may be amongst other types of data potentially unrelated due to the SaaS model. The next sub-subsections discuss the cloud service models with more detail.

Subsequent subsections perform an extensive review of the state-of-the-art security issues in cloud environments, looking into vulnerabilities, gaps, threats, attacks, risks, and general problems that affect the cloud business. Naturally, those issues can relate to specific deployment or service models, thereupon complementing previous discussions. The review is conducted with basis on the taxonomy defined in Sect. 3.6.

4.3 Software-Related Security Issues

Software security is and has been a vital part of computer systems. Normally, large applications with thousands of lines of code, or even millions, are written by several different people with distinct programming skills and ideals. Managing such applications is a hard task and gives rise to software holes that can be exploited by malicious users. Additionally, programmers find it more attractive to provide functional, fancy software, sometimes caring more about the interface and functionality, than security

and reliability. Moreover, open-source software is free and its code is exposed to analysis, easing the search for bugs to be explored. Software security is a system-wide issue that takes into account both security mechanisms and design with security purposes, and ends up in robust programming approaches to harden software and hamper attacks [12]. Hence, software security should be part of a full design and development lifecycle, and should start being considered in all software engineering phases [68, 77]. In the cloud context, software security is intrinsic to the PaaS and SaaS models. The following subsections cover security issues related with *multi-tenancy*, *platforms and frameworks*, *user frontend*, and *control*.

4.3.1 Multi-Tenancy

The multi-tenancy feature stems from the virtualization layer and allows storing information from multiple customers at the same physical hardware. This scenario promotes attackers to exploit this configuration in the form of *co-location*, *co-residence*, or *co-tenancy attacks*, wherein an attacker is able to disguise as a regular and legitimate customer to infiltrate neighbor VMs belonging to other real legitimate customers. Several aftermaths are possible, including compromising integrity, confidentiality and privacy. Hacking trough *loopholes or injecting client code* into the SaaS model are two possible ways to achieve that purpose [95]. Another series of issues are implied by multi-tenancy, such as *data breach* [85, 105], *computation breach* [105], *data loss* and *data leakage* [12]. The *object reusability* issue is mentioned in [116]. It can be a result of *data remanence* in multi-tenant scenarios. To tackle these issues, data must be well segregated, meaning that SaaS systems have to assure clear boundaries between customers data.

4.3.2 Platforms and Frameworks

Given that clouds follow a SOA approach, the problem of data integrity gets magnified when compared to other distributed systems [95]. Moreover, web services normally rely on eXtensible Markup Language (XML), SOAP and REpresentational State Transfer (REST), and APIs. Most vendors deliver their APIs without transactions support, which further complicates the management of data integrity across multiple SaaS applications. Additionally, the PaaS model supplies platform and frameworks to develop cloud applications. During the applications development lifecycle, programmers should deploy rigorous security measures focusing on authentication, access control and encryption [12]. Nonetheless, *unsafe* APIs and IDEs (may allow hosting *botnets* and *trojan horses*) with *insecure systems calls or deficient memory isolation* [72] may comprise points of entrance to attackers [113].

4.3.3 User Frontend

Besides the particular issues derived from insecure platforms and frameworks, other problems arise from the user frontend to services. In other words, because clouds provide on-demand self-service capabilities, services subscribed by customers require a management interface with fine-grained configurations of those services [41]. Usually, those interfaces are accessible via the Internet, therefore being at peril because of web related security issues [102]. Moreover, there is a higher probability of exploiting those administrative interfaces in clouds than in other systems where management functionalities are only accessible to a few administrators.

4.3.4 Control

Control refers to regulating the use of a system composed of an infrastructure, data, applications and services [115]. While a customer may be concerned about *malicious usage* and *malicious computation*, a provider may be worried about monitoring and logging events within the cloud systems, because standards and control mechanisms are scarce [41]. Log files record all tenant events. Thus, *insufficient logging and monitoring capabilities* may hamper or prevent pruning for a single tenant.

Software spans across almost all elements of a system. This subsection discussed specific security issues of this topic. Nonetheless, the following subsections present a more detailed description of issues that may recall software-related components.

4.4 Data Storage and Computational Security Issues

Data storage security also comprises an important aspect of Quality of Service (QoS) [108]. Clouds, however, complicated QoS requirements and raised new issues. As customers do not have their data locally stored, techniques and mechanisms to efficiently and reliably check the data are required. Auditing approaches are adequate for such task, but it would not be fair to let the provider or the customer execute the auditing, because neither of them can guarantee to provide unbiased and honest results [107]. In addition, customers may not have the time, willingness, resources, or feasibility to monitor their data. In such case, they may delegate such responsibility to an optional trusted third-party auditor. Storage outsourcing hardens the task of efficiently and securely verify that a server, or a group of servers, is faithfully storing outsourced data [4]. Following is a discussion of security issues related with *abstraction*, *integrity*, *confidentiality and privacy*, *availability*, *sanitization*, and *cryptography*.

4.4.1 Abstraction

Clouds raised the *loss of control* issue [111], in which the customer is physically separated from the servers storing or computing the data, having no possible control over them whatsoever. In addition to being physically away, the data is somewhere within the server pool, at an unknown location, disabling the possibility of pinpointing the storage partition, network port, and switches involved in the handling of the data [92]. This issue is due to virtualization abstracting the VM location. Additionally, it is hard to isolate particular resources that have been compromised. *Multi-location* [115] is a term used to refer data being held in multiple locations. In the case of a service provider, as a means to provide high availability, data is backed up to other clouds of the same provider, usually in other data centers. However, big players, like Google and Amazon, have data centers all over the world, rising compliance and legal issues as data travels across borders, which are further discussed in Sect. 4.8. Nonetheless, a trustworthy SaaS model must be capable of offering *data locality* reliability.

4.4.2 Integrity

Integrity refers not only to data integrity but also to computational integrity. Data integrity points out that data should be honestly and correctly stored, detecting corruptness, modification or unauthorized deletion. Computational integrity stands for authentic computation of programs and reliable results output without being distorted by *malicious agents*, and *downtimes* and *slowdowns*, respectively. Attacks threatening these properties are usually known by *data loss*, *data manipulation* and *dishonest computation* [111]. *Administrative errors* in data backups, restoration, migration, or other operational tasks may lead to data loss, while *malfunctioning* Central Processing Units (CPUs), *transmissions buses*, *vulnerable code*, *misconfigured policies*, or rootkit attacks may lead to *dishonest computation*. For example, MapReduce, a computing framework for processing large datasets in distributed systems, may be dishonestly executed by misconfigured or malicious machines, resulting in inaccurate computational results. Additionally, finding out which machines are compromised is a difficult task. On the other hand, Atomicity, Consistency, Isolation and Durability (ACID) properties are nowadays assured and preserved in standalone database systems; but distributed systems, such as clouds, created a challenge in transactions between data sources, which must be handled in a fail-safe manner [95].

4.4.3 Confidentiality and Privacy

Confidentiality is fundamentally related with privacy, both emphasizing that confidential or private resources are not to be accessed nor seen by unauthorized parties. Confidentiality refers to data confidentiality and computational confidentiality. In the cloud context, however, questions related with these properties arise when an

individual, a businessman, or a government agency shares information in the cloud [95]. Nonetheless, the risks vary significantly with the terms of service and privacy policies established in SLAs. Laws may oblige cloud providers to look for criminal activity evidence or other government security matters in data contents. As previously said, assessing if the confidentiality agreement is being fulfilled is hard for both the customer and third-party hired auditors because the cloud provider may not be willing to allow examining metadata files. *Malicious insiders*, governed by personal motivations such as low salaries, and by exploring defective personnel management or insufficient security clearances in the data center, may access and *disclose data* private to customers. A single incident can expose a huge amount of information. Pooling and elasticity of cloud systems determine that resources allocated to one user will be reallocated to a different user a later time. Thus, it might be possible for a user to read data written by previous users in terms of memory and storage, an issue named *data recovery* [41]. A similar problem occurs due to VM mobility, further discussed in the next subsection.

4.4.4 Availability

Cloud services need to be guaranteed that they are up and running around the clock and accessible on-demand. IaaS physical and virtual resources, like databases and processing servers, need to be available to support data fetch operations and computational tasks of programs. To this end, a multi-tier architecture is deployed in cloud systems [95], supported by a load-balanced farm of application instances running on many servers. This approach enables resiliency against Denial of Service (DoS) attacks by building software and hardware failure measures in all tiers. *Weak authentication mechanisms* and *weak session management* are possible exploitable threats to provoke a DoS in data and computational availability. For instance, a supposedly legitimate user can have servers processing highly demanding tasks, occupying resources that might be denied to other users.

4.4.5 Sanitization

Data deletion has been a concern in distributed systems, to which monitoring, marking and tracking mechanisms have been employed for data discovery [71]. In clouds, data sanitization is of extreme importance, useful to properly dispose of sensitive data belonging to customers. In fact, data centers, like the ones belonging to Google, have destruction policies that physically wreck hard drives, even though media sanitization is hard or impossible due to resource pooling and elasticity [41]. Nonetheless, de *deficient destruction policies* may result in *data loss* [12] and *data disclosure* [16]. Examples include discarding disks without being wiped out [2] or the impossibility of destruction because disks might still be being used by other tenants [41].

4.4.6 Cryptography

The main issues concerning application of cryptographic techniques are the use of *insecure or obsolete cryptography mechanisms* and *poor key management* [41]. One particular issue stems from the abstraction layer imposed by virtualization between VMs and hardware, which can weaken Pseudo-Random Number Generators (PRNGs), resulting in the production of poor cryptographic material. The usage of not up-to-date and faulty cryptographic algorithms might expose pertinent and encrypted data [113]. Furthermore, given that nowadays most computers have multi-core CPUs with high clock rates, *brute force* attacks are easier to perform. Hence, programmers should pay particular attention to the cryptographic mechanisms embedded on their applications.

4.5 Virtualization Security Issues

Virtualization technology is placed in the SaaS model of the cloud stack. It is one of the main innovative sides of clouds. While constituting a defense and fault-tolerant mechanism, virtualization also poses many security issues, as not all virtualized environments are *bug-free* [2]. A multi-tenant approach seems promising to the cloud providers perspective, but increases the co-location attack surface as VM-to-VM attacks become more probable [9]. Regardless of the fact that virtualization security has been the subject of research even before the emergence of clouds, achieving logical and virtual isolation has not yet been completely met. The following discussion addresses security issues related with *managing images*, *monitoring virtual machines*, *networking*, *integrity*, *confidentiality and privacy*, and *availability*.

4.5.1 Managing Images

VMMs allow VMs to be easily turned on, off, or suspended by saving its current state in images. At the next boot, the state is loaded and applications can run or rerun as normally. VMs images contain information of files, processes, and memory blocks of the guest OS, therefore being potentially large-sized. This may pose performance challenges to cryptographic techniques [104]. Images are usually kept offline at an image repository. Even in offline state, they are vulnerable to *theft* and *code injection* [72]. One possible workaround is to concatenate various images, given that is harder to copy larger files than smaller ones. The administrator of an image repository risks *hosting and distributing malicious images*. Security properties of dormant images are not constant and degrade over time because an unknown threat may appear after the time of publishing images [110]. Moreover, images should converge to a steady state by performing scans for worms and other viruses, otherwise *infected* VMs can sporadically disseminate malware, an issue named *transience* [36]. This also applies to software licenses, security patches and updates, wherein administrators

tend to overlook long-lived inactive images because of high *maintenance costs*. Other issue is known as VM *sprawl* [65], referring to the possibility of having the number of VMs continuously growing while most of them are idle or never back from sleep, in turn wasting resources and complicating VMs management. A cloud provider risks leaking data due to unwittingly making files public. A cloud user risks running *vulnerable*, *malicious*, *out-of-date or unlicensed images* stored at an insecure, wrongly administrated repository [110]. The danger of compromising images lies in bypassing firewalls or IDSes by running an apparently legitimate VM, and place it in the cloud. VMs encapsulate software dependencies, allowing easy propagation of *trojan horses*.

4.5.2 Monitoring Virtual Machines

VMMs do not offer perfect isolation, inspection and interposition [95]. For example, Virtual PC and Virtual Server from Microsoft may allow *running code* on the host system or on another VM, or *elevating privileges*. Because of an input validation error, *Xen* allowed the *execution of commands* in Domain0 by a root user of a guest OS. Such problems are due to VMM vulnerabilities. VM *escape* refers to the case of gaining access to the VMM through a VM [50], therein being capable of attacking other VMs monitored by the same VMM. Well-known VM *escape* attacks include SubVirt [60], BLUEPILL [89] and Direct Kernel Structure Manipulation (DKSM) [7]. *Zero-day* vulnerabilities consist of vulnerabilities that are exploited before the developers know about them [70]. In the virtualization context, one could be capable of gaining access to VMMs or VMs. HyperVM was once exploited without the knowledge of the provider, resulting in the destruction of many web sites [40]. Furthermore, monitoring all VMs in a data center may massively increase the *computational overhead* due to the wide range of OSes that can be deployed in seconds, an issue named VM diversity [36, 103]. The ease of cloning and distributing VMs throughout cloud servers can also *propagate errors* and make arise to other vulnerabilities. Thus, more work is needed to enhance behavioral and introspection VM techniques while having in mind operational cost.

4.5.3 Networking

IaaS provides the means to resource pooling, therefore enabling customers to share physical resources and, most likely, networking equipments. Vulnerabilities originated in communication protocols also affect clouds, such as in Domain Name Service (DNS) servers [41, 72]. In virtualized networks, *administrative access* and *tailoring* is limited, potentially leaving uncovered holes [41]. *Incorrect network virtualization* may allow user code to access sensitive portions of the underlying infrastructure, disclosing sensitive knowledge of the real network or resources from other users [2]. *Packet sniffing* and *spoofing* also applies here. Virtualization software, such as virtual switches, may contain vulnerabilities that enable *network-based*

VM attacks [72]. Another important aspect in virtualized networking is the traffic produced, which is twofold. It occurs not only in real, physical, standard Ethernet networks, but also within virtualized environments. It can be defying to control both kinds of traffic because tried-and-tested network-level security might not work in the virtualization layer [41]. For instance, Virtual Local Area Networks (VLANs) and firewalls prove *less effective* when moved to such environments [104]. Moreover, securing dynamic establishment of *virtualized communication channels* is aggravated when VMs dwell across various IaaS platforms [80]. VM *mobility* [36, 103] is an issue arising from the resource pooling feature. VMs are likely to be copied or moved to other servers via network links, enabling quick deployment, but also quick *spread of vulnerable configurations* [50] and *image theft*. VM *moving*, also called *live migration* [114], is susceptible to many attacks. *Man-in-the-middle* attacks [75], Time of Check to Time of Use (TOCTTOU) vulnerabilities and *replay* attacks [114] are examples of such attacks. Copying images is also known as VM *cloning* or *template image cloning* [31, 41]. An image can be manipulated to provide backdoor access in the future, when instanced. Also, a template image might retain the original owner data which, when copied, may leak sensitive information like secret keys and cryptographic salt values. Moreover, various copies of the same VM may exist, and an attacker may access one and read its contents unnoticed, while trying to break the administrator password. Computer forensics techniques can be applied to obtain complete history of the VM, including usernames, passwords, applications and services, Internet browsing history, and IP addresses.

4.5.4 Integrity, Confidentiality and Privacy

Previously described issues can have impact on integrity, confidentiality and privacy properties. Nonetheless, more specific issues that compromise those properties are described herein. VM *hopping* is a term referring to maliciously gaining access to another VM belonging to a different cloud user [50], which can happen due to VMM *isolation failure*, as discussed earlier. Also known as cross-VM attacks [111], they require that two VMs are running on the same physical host and the knowledge of the IP address of the victim. According to [85], such requirements are easily met, highlighting the crucial importance of addressing these issues. Moreover, a single VMM is most likely to place many VMs co-resident on the same machine. In a successful attack, it is possible to monitor resources, modify configurations and files. Amazon EC2 was successfully exploited in 2009 [85] and 2011 [13] through *cross*-VM *side-channel* and *covert-channel* attacks. *Side-channel* attacks passively observe data flowing on the server, while *covert-channels* monitor and send data [17]. Variations of these attacks are being researched, such as using CPU load as a *covert-channel* [76], and L2 *cache covert-channel* to leak small useful information, such as private keys. *Timing side-channels* are regarded as an insidious security challenge because they are hard to control, enable stealing data, only the cloud provider can detect them, and can undermine efficiency [5]. Virtualization technologies may use a memory deduplication technique that allows reducing physical memory usage.

Attacks to this technique have been described in the literature [96, 97], and have the purpose of detecting applications or files on a co-residing VM. Furthermore, a series of attacks conducted by a *malicious insider* have been demonstrated in [87] and used to get plaintext passwords in memory dumps of a VM, extract a private key out of a key pair using memory snapshots, execute arbitrary commands in a backup copy of a VM by exploiting Domain0, and compromise data by exploiting VM relocation. The aforementioned attacks exploit the very nature of multi-tenancy, making it possible to access data belonging to other tenants. Thus, integrity, confidentiality and privacy properties are compromised by such attacks. Availability is also compromised because the attacker can stop services or ruin boot configurations so that VMs need fixing. Virtualization issues are key issues in clouds.

4.5.5 Availability

To compromise availability in virtualized environments, it is possible to take one VM, or more, under the control of an attacker, occupy all available resources so that the VMM cannot handle more VMs [103]. In such case, support to other VMs would be denied. However, a threshold for resource allocation should be deployed to mitigate this issue.

4.6 Networking, Web and Hardware Resources Security Issues

Cloud infrastructures are not only composed by the hardware where the data is stored and processed, but also by the path connecting to the point it gets transmitted. In a typical cloud scenario, data is split into a vast number of packets that are transmitted from source to destination through umpteen number of third-party infrastructure devices and links [85, 113]. The Internet is the most prominent example to use here. It is already known to suffer from *man-in-the-middle*, IP *spoofing*, *port scanning*, *packet sniffing*, *botnets*, *phishing*, and *spam* attacks. Consequently, the cloud inherits such issues from the Internet, even though *botnets* in clouds are easier to shutdown than traditional ones [17]. So, even if large amounts of security measures are put in the service models, the data is still transmitted through common Internet technology. Additionally, technologies to access the cloud vary from service enabled fat clients to web browser-based thin clients [55], being the latest the most common nowadays [45]. In fact, SaaS applications are required to be accessed over the web, by which a browser is most suitable. Thus, clouds also inherit web security issues, such as the ones mentioned in *The Ten Most Critical Web Application Security Risks* [77], a document containing the top ten web security issues, which was elaborated by Open Web Application Security Project (OWASP), a non-profit organization dedicated to the widespread of good application security practices. For example, *injection* tops the list, while XSS stands second. Web services play an important role in clouds, therefore deserving a special attention in this subsection. Below, a discussion of security issues

related with *communication protocols and standards*, *integrity*, *confidentiality and privacy*, *availability*, and *accountability* is thus included.

4.6.1 Communication Protocols and Standards

By design, the HyperText Transport Protocol (HTTP) is stateless and does not guarantee delivery nor supports transactions. To address this, web applications usually implement session handling techniques, many times being vulnerable to *session riding or session hijacking* [41]. As mentioned before, Dynamic Host Configuration Protocol (DHCP), DNS, and IP are among known vulnerable protocols that might enable *network-based cross-tenant attacks* in IaaS. Distinct technologies are used to access cloud resources [70], such as HTTP and HTTP Secure (HTTPS) for SaaS applications; SOAP, REST and Remote Procedure Call (RPC) technologies for PaaS web services and APIs; and remote connections, VPNs and File Transfer Protocol (FTP) for IaaS storage and computational services. Thus, web security and known protocol-related security issues are also relevant to clouds.

4.6.2 Integrity

Database systems require transaction support in order to guarantee data integrity. As HTTP fails to do so, *transaction management* comprises an obstacle that should be handled at the software API to enhance transactions in SaaS applications. An attack named *metadata spoofing* consists in reengineering metadata descriptions of, for example, Web Service Definition Language (WSDL) documents by establishing a *man-in-the-middle* [51]. Using such attack, it is possible to specify operations different from the ones in the document, allowing to create user logins, for instance. Besides compromising integrity, it also compromises authentication, posing as a different threat. However, if sound techniques are used, the attack is easily detected. Nonetheless, one has to take into consideration that, in clouds, WSDL documents are more dynamically accessed than on other systems, drastically raising the potential spread of malicious files and, thereupon, the probability of a successful attack.

4.6.3 Confidentiality and Privacy

In the network and web context, compromising confidentiality and privacy may imply compromising authentication mechanisms firstly. Network penetration and packet analysis; session management weaknesses; and incorrect Secure Sockets Layer (SSL) configurations can lead to *active session hijacking* and *credentials access* [95]. Concerns regarding web services existed before the emergence of clouds. For example, *wrapping* attacks [55, 69, 82], also known as *rewriting* attacks, consist in rewriting SOAP messages by adding a wrapper and forged XML field to access the target web resource. A valid SOAP envelope signature of the original message is maintained,

to validate the execution of the modified request. In 2009, Amazon EC2 was found to be vulnerable to a variation of a *wrapping attack* [42], which allowed performing an arbitrary number of EC2 operations. Moreover, an attack named *SOAPAction spoofing*, which consists in modifying HTTP headers after intersecting them to also invoke other operations, was perpetuated in a .NET web service in 2009, as well as an XML *injection attack* [51]. Another attack, entitled WSDL *scanning* attack, has been addressed in various studies [30, 51]. It consists in discovering and fingerprinting web services to find omitted, confidential operations, supposedly available only to administrators. *Data leakage* is a possible aftermath of all these attacks. Note that the aforementioned attacks raise confidential and privacy concerns, as executing arbitrary operations may output sensitive information or allow the attackers to access unauthorized resources.

4.6.4 Availability

Data centers require bandwidth-wise networking links to support large amounts of network traffic. However, in 2007, Cisco stated [21] that large server cluster designs are usually under-provisioned in terms of network capacity with a factor of 2.5:1 up to 8:1, meaning that the network capacity of data centers is less than the aggregate capacity of the hosts inside the same subnet. This *bandwidth under-provisioning* [111] vulnerability worsens the impact of *flooding* attacks, raising availability and QoS concerns. DoS attacks by *flooding* are some of the most concerning in computer systems. For example, a botnet can be used to send millions of Transmission Control Protocol (TCP) SYN messages to the target server in a small period of time, creating a Distributed DoS (DDoS) which can overload the server processing all the requests. DoS attacks are often termed as direct or indirect attacks. A direct DoS attack implies pre-determining the target service, which potentially gives in to the overload. A possible side effect of direct approaches is that adjacent services running on the same hosting machine can also be denied, creating collateral damage—an indirect DoS attack. A condition known as *race in power* [52] is the worst case scenario of DoS attacks. As cloud systems may aid overloaded machines by relocating services to other machines, the workload can be propagated throughout the system. At some point, the system aiding may also host the *flooding*, placing both cloud systems off against each other, both aiding one another with resources until the point where one, finally, gives in and reaches a full loss of availability state. A new form of particular cloud DoS has been discovered in 2010 [63]. The objective is to starve an exploitable bottleneck uplink found in the topology. To perform such attack, it is required to gain access to enough hosts within the target subnet and to produce, preferably, User Datagram Protocol (UDP) traffic upwardly through the uplink, in order to consume its bandwidth, having the side effect of starving other TCP connections, who back off during congestion. *Resource exhaustion* [51], a particular case of DoS, is characterized by a large attack surface. The *oversize payload* attack on web services consists in increasing memory usage when Document Object Model (DOM) parsing is employed to transform XML documents into memory objects. A

raise of memory consumption with a factor of 2:30 has been observed for common web service frameworks, such as the Axis web service, which resulted in an out-of-memory exception. Another attack named coercive parsing exploits namespace vulnerabilities in XML parsing with the purpose of overusing the CPU. Axis2 web service was exploited, causing CPU usage of 100 %. Moreover, the *obfuscation* attack aims at overloading the CPU and increasing memory usage. With the same objectives, the *oversized cryptography* attack exploits buffer vulnerabilities and encrypted key chains in web services. Other related issues include WS-Addressing spoofing attack [53], in the Business Process Execution Language (BPEL) [51] and flooding by using XML messages [18].

4.6.5 Accountability

Clouds follow a pay-as-you-go business model. In terms of networking, bandwidth rented by customers is also billed according to their needs and usage. *Flooding* also impacts accountability, raising costs for cloud customers. SLAs must, therefore, cover determination of liability in case of abnormal large bills, meaning that the responsible party must be determined [111]. Fraudulent Resource Consumption (FRC) [48, 49], a more subtle and evasive attack than DoS, has the goal of exploring the cloud pricing model to cause *financial loss* to the victim. It consists in sending requests to consume bandwidth continuously and for a long period of time, but not intensively enough to cause DoS. It is hard to analyze and classify traffic belonging to a Fraudulent Resource Consumption (FRC) attack, which is representative of an Economic Denial of Sustainability (EDoS). Also, in order to achieve maximum profitability, cloud providers choose to multiplex applications of different customers in the resources pool, in order to achieve high utilization. However, this may cause incorrect resource consumption metering, resulting in additional costs for the customers and leading to *inaccurate billing* [111].

4.7 Access Security Issues

Access to cloud resources concerns the security policies deployed by an organization to its employees, according to their role in the company. For instance, personnel with lower security clearance should not have access to certain datasets of higher security clearance. Thus, SaaS applications should be capable of being customized and configurable to incorporate specific access security policies. To this end, authorization and authentication mechanisms are put in place, which commonly resort to a combination of username and password, called credentials, which are private to each user. A discussion of security issues related with *data centralization*, *credentials*, *authentication*, *authorization*, *identity management*, and *anonymization* is included subsequently.

4.7.1 Data Centralization

Data centers condense in a single centralized point massive amounts of data and resources. They pose as appealing attack points, which may be even a more severe issue if *malicious insiders* and *malicious outsiders* infiltrate the facility and the cloud. The vulnerability might lie on the deployed security procedures for physical and logical access controls. Unpleasant or former employees, customers, hobbyist hackers, espionage agents, or other cybernetic intruders portray a possible malicious community, ready to exploit the right opportunity. Outside threats pose greater impact on clouds, not only in terms of system damage, but also to the provider reputation and business, due to the long term loss of leaving customers [9]. Nevertheless, monitoring of privileged administrators should be carried out, because they can turn into malicious sysadmins—administrators with *malicious intents* [57]. Henceforth, deploying firewalls, Access Control Lists (ACLs) and IDSes or IPSes is mandatory.

4.7.2 Credentials

Usually, LDAP or Active Directory (AD) servers are used to manage credentials in larger systems. In the cloud computing paradigm, those servers can be placed inside the company network, behind a firewall, or outsourced to the cloud provider system. The option mentioned in last increases IT *management overhead* if multiple SaaS services are rented by the customer, because one has to add, modify, disable, or remove accounts, as personnel leaves or enters the company. Furthermore, the loss of control issue applies here also, as the customer is deprived of configurations and security of LDAP or AD servers [92]. In the past, weak password-recovery mechanisms resulted in *weak credential-reset* vulnerabilities when access responsibilities are outsourced [41]. Credentials have long been an issue of remote access mechanisms. If they are stolen by means of *phishing*, *fraud*, *keyloggers*, *buffer overflows*, or other software security holes, *service hijacking* becomes the most probable menace [9]. In such case, monitoring or manipulating data and transactions is possible, along with performing *malicious redirects* or launching DoS attacks by using the compromised account as an attacking concealed base. *Replay sessions* are most likely to happen. Moreover, it is possible to perform User to Root (U2R), in which the attacker gains root level access to VMs or hosts after gaining access to a valid user account [70]. Either way, *service disruptions* can cause a business halt or lose valuable customers, ultimately leading to *financial loss*.

4.7.3 Authentication

A centralized service system approach is advantageous in SaaS authentication because of centered monitoring, making software piracy more difficult [19]. Remote authentication mechanisms rely mostly on regular accounts, nevertheless being susceptible to a plethora of attacks [39]. *Dictionary* and *brute-force* attacks are amongst

the most prominent examples. Various approaches for authentication exist, such as simple text passwords, third-party authentication, graphical passwords, biometric scans, and 3D password objects [28]. Simple text passwords are usually static, long-living approaches. Nevertheless, *archaic static password*—one-tier login—is simply not good enough, constituting actually one of the biggest security risks [44]. For small cloud deployments, third-party authentication is not usually preferred. While graphical password schemes require a long time, biometric technologies, like fingerprinting, palm printing, and iris or retina recognition, may violate the user personal space. Moreover, they require physical presence, thereupon being not applicable in remote systems, but suitable for physical data center security. Finally, 3D passwords do not support multi-level authentication. Still on authentication topic, it is important to mention that customers are most likely to subscribe multiple services to a provider, resulting in several login requirements. Besides Hart [44] claiming one-tier login not being enough and while Tripathi [102] also acknowledges the difficulty in deploy strong user-level authentication, multi-level authentication mechanisms for several services is a hard task to achieve, because their management and reliability are hard to deploy. SSO is perhaps the most used technique. Google, for instance, was once vulnerable in their SSO implementation that allowed users to switch between services, like Gmail and Calendar, without re-authenticating [66]. The Security Assertion Markup Language (SAML) would be used to carry *impersonation* attacks and access a target Google account. Many authentication mechanisms have a threshold for authentication attempts to fight *brute-force* attacks. However, an availability issue arises in the form of DoS via *account lockout*, meaning that an attacker can repeatedly, and in quick succession, try to authenticate with a valid username until the threshold is surpassed [41].

4.7.4 Authorization

Due to the growth of cloud storage systems, services performing mashups of data are likely to be common in the future [19]. It was previously said that centralized access control would be advantageous but, in this scenario, it may not be, however, possible or desirable. The development of data mashups have security implications in terms of *data leakage* and on the number of resources a user retrieves data from. The latter places access authorization requirements for reasons of usability. For instance, Facebook does not typically verify third-party appliances, which use data uploaded to its servers. *Malicious applications* can, therefore, perform malicious activities. Other social sites are also endangered [109]. *Insufficient or faulty authorization checks* are possible attack vectors, like an *insecure direct object reference*, also called Uniform Resource Locator (URL)-*guessing* attacks [41], an issue with rank four in the top ten web application security issues of OWASP. Service management interfaces are also prone to offering *coarse authorization control models*, making impossible to implement duty separation capabilities.

4.7.5 Identity Management

Identity Management (idM) is a broad administrative area that deals with identifying entities and cloud objects, controlling access to resources according to pre-established policies [72]. Three idM perspectives are often considered [95]. The first perspective, pure identity, manages identities with no regards to access or entitlements. The second perspective, log-on, uses traditional approaches by using physical tokens, such as smartcards. The third perspective, service paradigms, delivers online, on-demand, presence-based services with respect to roles, appropriate to cloud services. Three idM models were also identified in [95]. The first, independent idM stack, is maintained at the provider end, keeping usernames, passwords, and all related information per SaaS application. This model should be highly configurable to comply with the customer security policies. The second, synchronized credentials, consist in replicating account information stored at the customer end in the provider, giving access control abilities to the provider. It is in this model that an issue known as *account information leakage* is mentioned. The third, federated idM, provides the means for linking account information storage across multiple idM sources, being SSO one good example of such mechanisms. Authentication occurs at the customer side, while users identity and other attributes are propagated on-demand through federation to the provider. *Trust* and *validation* issues appear in this model. Apart from that, large PaaS and SaaS platforms have complex hierarchies and fine-grained access policies, therefore raising logistic and transport issues in synchronizing data [92]. Moreover, using different identity tokens and identity negotiation protocols in general idM might present interoperability drawbacks also [98].

4.7.6 Anonymization

Some systems implement anonymous access to enhance security, aiming to prevent disclosure of the identity information of users. Notwithstanding, full anonymization brings the *hidden identity of adversaries* issue [111]. *Malicious users* can jeopardize security if they acquire access to an anonymized account, hiding exploitation tracks and becoming undetectable.

4.8 Trust Security Issues

Outsourcing services brings several trust issues. Firstly, cloud customers must heavily trust providers with their data, losing control over it. Secondly, providers must trust and provide access for customers to access cloud resources. Trust is not only related with the relationship between cloud stakeholders, but also with the assets in the cloud, relying on remote data storage mechanisms, computational algorithms, hardware, virtualization techniques, and web-based access [56]. Nonetheless, sometimes trust cannot be established or, even if it can, it may not be enough to make the

customer comfortable. Hence, additional means are expected to be developed in order to increase security confidence in the cloud business. In fact, trust is also important in other distributed systems, such as Peer-to-Peer (P2P) and sensor networks [34]. This section contains a discussion on security issues related with *reputation*, *compliance and legal issues*, *auditability*, *computer forensics*, *confidentiality and privacy*, and *anonymization*.

4.8.1 Reputation

The cloud operation mandates sharing assets among several users. Hence, activities and behaviors of cloud stakeholders affect each others reputation, an issue known as *reputation isolation* [33, 71] or *fate-sharing* [17, 86]. For example, a user may subvert a system, in turn disrupting services to other users. Moreover, all of them benefit from the security expertise concentration that cloud computing offers, depending on the signed SLAs, consequently sharing the same infrastructure and fate. In 2009, Amazon EC2 was subverted by spammers who caused blacklisting of many internal IP addresses, which resulted in major service disruptions [17]. A second noteworthy incident occurred in the same year, in which federal agents seized data centers suspicious of facilitating cyber-crime. Many cloud customers, namely companies, without knowledge of the suspicions, had business disruptions, halts, or even complete closures. Therefore, hardware confiscation, a result from applying law-enforcement, is an issue named *e-discovery*, by which data *disclosure* and *data loss* are major risks [39].

4.8.2 Compliance and Legal Issues

The most clear legal issue stems from the abstraction and multi-location properties of clouds. Several cloud providers have data centers spread all over the world for providing geographic redundancy. However, many countries do not allow data to leave country boundaries. If such happens, to which country jurisdiction does the data falls under when an incident takes place? Or who has jurisdiction over data as it flows across borders? Moreover, can government agencies access the information as it changes jurisdiction? Also, can a provider deliver trustable reports of the customers data health? If served a subpoena or other legal action under a limited time-frame, how can a customer compel the provider, if even possible, to gather potential required evidences within the time-frame? According to Refs. [19, 58, 71], such questions have fuzzy, unclear answers. Additionally, *dishonest computation*, *accidental resource allocation*, *availability issues*, and *data loss* constitute possible violations to SLAs. In any case, compliance issues are at cause. Thus, the risk of investing in certification, such as industry standards or regulatory requirements, is high to customers due to failure of providing compliance evidence by providers [33]. Anyhow, public clouds imply that certain kinds of compliance cannot be achieved, like the security requirement Payment Card Industry (PCI) Data Security Standard

(DSS) [79]. Other perspective on the compliance and legal topic is related with the possibility of different alignment interests between cloud stakeholders [19]. *Limited usability*, *implied*, and *obliged contractual* or *unclear terms* can pose issues in the customers service usage context. After the SLA is closed, the customer remains at the mercy of the provider. Consequently, customers may, or may not, trust particular providers with basis on the SLAs they offer. SLAs are normally not consistent amongst different providers, leading to a higher uncertainty in identifying the most trustworthy providers [43]. In some cases, providers might use subcontractors, to which costumers have even less control, influence, compliance certainty, and trust [19], raising a problem known as the *transitive nature* issue. In such case, who is to blame when an incident takes place? The provider or the subcontractor? Problems like this have happened in the past, resulting in *data loss*. More on the law side, issues arise when providers must obey government regulations for disclosing data of lawful interception, which may break the trust chain created with customers and originate conflicting points. For example, the USA PATRIOT Act (UPA) conflicts with the Personal Information Protection and Electronic Documents Act (PIPEDA) and Data Protection Directive in Canada and Europe, respectively. Law acts have nonetheless been published to protect individual privacy and business secrets; but some might not be in accordance with the newly cloud requirements. The Electronic Communications Privacy Act (ECPA) of 1986 and UPA of 2001 are examples of acts that fail to protect data being disclosed to government entities. The Fair Credit Reporting Act (FCRA) of 1970, the Cable Communications Act (CCA) of 1984, the Video Privacy Protection Act (VPPA) of 1988, the Health Insurance Portability and Accountability Act (HIPAA) of 1996, and the Gramm-Leach-Bliley Act (GLBA) of 1999 are other examples of acts that fail to protect data being disclosed to private entities. Thus, such out-of-date acts are inapplicable to cloud scenarios, as of 2010 [115].

4.8.3 Auditability

The answer to the question of the *provability of data deletion* is very relevant to a company retention policy [19]. Moreover, how can a customer be assured that data was really deleted or that security mechanisms are really being applied? Is trusting the provided reports, if any, enough? Auditability enables assessing the security requirements by putting an additional layer above virtualized guest OSes [2]. To this end, methodologies are necessary to analyze service conditions, monitor intrusions, accesses, record logs with detailed descriptions of what happens, and other events [39]. Nevertheless, cloud providers may not be willing to allow conducting audit operations [33]. Mutual auditability may provide a collaborative approach to be sure of each others *trustworthiness*, improving incident response and recovery. Customers can delegate auditing to third-party entities, leaving those responsibilities in the hands of specialized personnel. Auditability hastens blame attribution in case of search or seizure incidents, which can be vital to the cloud stakeholders, so that law enforcement agencies do not overreach when carrying out their duties during

data collecting procedures in data centers [17]. In fact, data might be forced to be kept within jurisdictional bounds so as to be valid in court if necessary. Nonetheless, businesses may not often like the fact that agencies might try to get their data via the *court system*, overlooking some laws and potentially peeking into the company secrets [115]. In any case, auditability methods are hampered by the *abstraction* issue (data is not physically on a single or fixed set of servers), and some of those methods are not privacy-preserving capable, pointing out research interests in this area [112].

4.8.4 Computer Forensics

Computer forensics is a particular form of auditing that has emerged in recent years to fight against cyber-crime. The main goal is to determine digital evidence by means of analysis techniques [99]. But clouds push and spread data further back into the network and servers, rather than purely being on a physical computing device. Therefore, investigative services and mechanisms also face the *abstraction* issue. According to [44], it is possible to comply with forensics with adequate access to the data, but more difficult in clouds. Private clouds are surely easier to analyze than public ones, because servers, applications, databases and other resources are more easily enumerated [99]. From the user perspective, forensics raise concerns in terms of *data seizing* and *data disclosure*, compromising confidentiality and privacy, while for the party conducting the forensics activities, the cloud stack exhibits several challenges. Key evidence may reside in web browsers history and caches, and other artifacts [20], which are remote to the cloud and hard to get, posing difficulties in *collection*, *collation* and *verification* tasks. Finding out what a user did from the beginning to the end of a service disruption is hard [99]. Virtualized platforms may also give rise to *unsound forensic data* and encryption schemes employed by customers and providers, privacy protecting acts, and time-consuming procedures to gain legal authority, also make of cloud forensics a difficult task to achieve. Moreover, the *lack of validation for disk images* in a scenario demoted of cryptography techniques may pose as a potential problem. *Evidence acquisition* is, therefore, a forefront security issue in cloud forensics [32, 99]. Computer forensics requires solid trust layers, not only at the provider and customer sides, but also in the court. A jury or a judge of a legal action ultimately has to decide whether or not the evidence presented is believable, reliable and trustworthy enough.

4.8.5 Confidentiality and Privacy

From the customer standpoint, *malicious sysadmins* constitute one of the major threats in clouds because of their privileged access, raising trust barriers [91]. A sysadmin is able to install all sorts of software and access VMs. For example, XenAccess allows running a user level process in Domain0 that directly accesses VMs memory contents at runtime. Other more sophisticated security attacks, such as cold boot attacks and *hardware tampering*, can be carried out for the same outcome. A

malicious sysadmin can further remove security-specific kernel modules, like firewalls and anti-viruses, making cloud systems vulnerable [117].

4.8.6 Anonymization

Anonymization can cut semantic links of the data to their owners, to overcome trust issues, while preserving the provider capability of charging resource usage in a proper and reliable manner [54]. Enterprises have actually felt increased pressure to anonymize their data until proper, trustable privacy mechanisms are in place [19]. For example, in the search marketplace, Google modifies the last IP address byte of logged searches after 9 months, and deletes it after 18 months. It also anonymizes cookie information, a process called generalization, which can guarantee reasonable privacy protection [100]. Anonymized data is retained for internal purposes. Even though anonymization is a difficult task, efforts have been carried out to address it in clouds [10, 54]. Nonetheless, *de-anonymization* attacks have been designed too. A family of attacks targeting social networks includes the active, the passive, and the semi-passive attacks, which *breach edge privacy* of a group of individuals [6]. Another study proposed an algorithm with 12 % *de-anonymization* error rate on an online photo-sharing website by only using network topology information [73]. *De-anonymization* attacks on dynamic social networks have also been studied, which used correlation techniques [29] to derive information about users. A curious *de-anonymization* attack targeting health records resulted in identifying the governor, at the time, of the Massachusetts state in the USA. It was proved that *innocuous and neutral data injection*, such as gender and birth-date, on anonymized data can lead to *information leakage* [92].

4.9 Other Security Issues

This subsection discusses security issues that either are not classified according to the taxonomy defined in Sect. 3.6 or that may fall in various categories. A discussion of security issues related with *virtualization and web*, *governance*, and *unknown threats* is included below.

4.9.1 Virtualization and Web

This specific topic includes only the *cloud malware injection* attack [55]. It consists in injecting malicious services or VMs into the cloud, serving any particular purpose of the attacker. The attack is initiated by injecting a service into the SaaS or PaaS models; or a VM into the IaaS model. Secondly, the cloud system must be tricked to treat the service or VM validly. Ultimately, authentic user requests are redirected

to the malicious instance and the code written by the attacker is executed, which can compromise the overall security of the system.

4.9.2 Governance

Governance issues consist in losing administrative, operational and security controls. In cloud environments, the *lock-in* issue might end up in a governance problem. As discussed earlier in Sect. 3.4, interoperability between clouds still faces security and standardization issues, namely when it comes to protocols, data formats and APIs. As a result, customers might become hostages of the provider of their choosing with regard to their data, becoming vulnerable to a few issues, namely the impossibility of data migration, price increases, reliability and security problems, service termination, or even providers going out of business [2, 39]. Standardized APIs would allow customers to deploy SaaS applications and have copies of the same data across multiple providers, mitigating the problem of a cloud provider having all copies in case of going out of business. Having perfect interoperability could lead to a *race-to-the-bottom* condition in terms of cloud pricing, resulting in flattening the profit for providers [2]. Nonetheless, it is normally argued that, on the one hand, customers may not adopt low-cost services because QoS and security properties do matter for them and, on the other, new possibilities to integrate hybrid clouds both on-premises and on-premises would arise. Moreover, besides the *data locality* issue, customers also face losing control over redundancy, file systems, other configurations related with storage and computation and, more importantly, security, namely at the physical level in the IaaS model [19].

4.9.3 Unknown Threats

The CSA finds *unknown risk profile* as one of the top threats to cloud computing [25], resembling *zero-day* vulnerabilities, earlier discussed in Sect. 4.5. Many companies might overlook security issues if the outcome benefits outweigh the risks taken. In this scenario, unknown risks arise when security matters are posted in background or in hold, or are low prioritized. Furthermore, the CSA finds information about who is sharing a cloud infrastructure to be pertinent to assess security risks, in addition to network intrusion logs, redirection attempts, and other types of logs. This is a prime example of why security within cloud environments must be upheld as high priority, for there is always the possibility of an unknown threat and risk [113].

5 Summary of Cloud Security Issues

The review of the literature presented in the previous sections shows that security concerns are focused on the new characteristics of cloud environments. Although

their unique and innovative nature brings new challenges to the IT industry and academia, some already prevalent issues in IT also affect clouds. Because clouds are accessed via the Internet, all the known issues from the latter are inherited by clouds. Some may be amplified if the cloud business model and pricing model are explored. Furthermore, virtualization, which is not a technology that emerged with clouds but an independent component that preceded clouds, got boosted up with the fuzz around cloud computing, being now one of the main cloud characteristics and of extreme importance. However, it enlarges the VM-to-VM attack surface because of VMM vulnerabilities, commonly known as *cross*-VM attacks. Several studies focus on this topic alone, some showing the malicious exploitability of popular cloud solutions and, henceforth, raising alarms for this topic. The fact of sharing a network medium and physical devices might be a major decision factor for cloud customers not adopting cloud solutions. As a consequence, the provider-customer trust bridge might get smaller if auditability is not possible to conduct. Further, the possibility of computer forensics seizing a data center to carry out their duties and the existence of out-of-date acts also contribute to the discussion of cloud security. Outsourcing IT duties to third-party cloud providers is, therefore, an issue yet to overcome.

Figure 3 contains a timeline summarizing most of the studies described in this chapter by indicating in what years they were published and the underlying issue category. In addition, the studies provided by organizations that are considered pioneering in the field are also included. By observing the figure, it is perceivable that both the industry and academia started to study the cloud computing environments and its security state soon enough. Its analysis also shows that the research in 2011 was proliferating, with many studies on the virtualization, networking, web, and hardware resources, access and trust categories. The trust studies, however, are mostly related with de-anonymization, pointing out research directions for applying anonymization in clouds. A deeper analysis uncovers an increase in the scope of issues discussed throughout the studies. In other words, in the initial years, studies focused on single issues. From then on, academia started working on understanding the wider security scope of clouds by studying more issues on single studies. For instance, the study in [19] from 2009 is included in three categories, while [41]

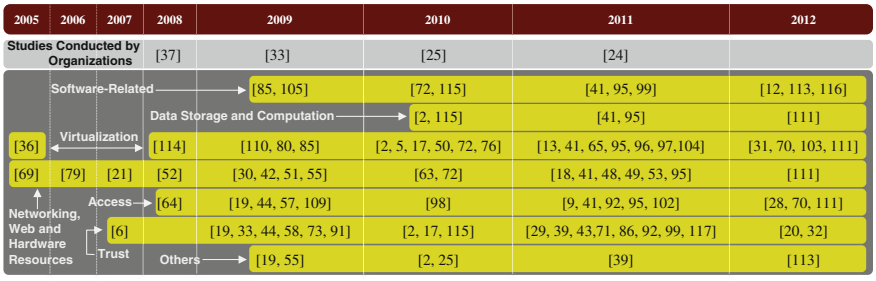


Fig. 3 Timeline exhibiting the date of publication the several studies on the cloud security subject, structured according to the categories used along the chapter

is included in five categories in 2011. Although this conclusion stands true for the sample of studies presented in the chapter, it is believed that other studies might also study the wide security scope of cloud systems.

6 Conclusions

In the last few years, the emergence of cloud and cloud computing related technologies changed the distributed systems paradigms, mainly due to the proliferation of virtualization. Clouds integrate virtualization techniques with modern IT, to allow building an elastic pool of resources that can be seamlessly delivered on-demand to costumers. These resources are normally managed remotely through self-serviced administration interfaces and billed using a pay-per-use model. These appealing features attract attentions from all over the industry and academia. A potentially very dangerous community is also drawn in, which is intrigued by the innovative operation of clouds and the data they hold for storage or processing purposes, posing as a threat to the overall business model. Enterprises changed how their IT investment is done, allowing cloud providers to boost underlying markets by creating a panoply of cloud solutions. As a consequence, data centers have been increasingly becoming more popular, spreading out the buzzword of *cloud computing* to the edges of the IT world. In order to make a completely secure investment out of clouds, it is of utmost importance and absolutely required to address the identified security issues, so as to allow a greater number of cloud native applications and products to be developed. Nonetheless, cloud technology and some of its security issues are relatively new, making some of the already existing solutions directly inapplicable.

Throughout this chapter, an extensive survey on cloud security issues was presented, discussing issues of several domains in the cloud. The literature on the subject was analyzed in the chapter, in which a review of each study was conducted to determine their goal and select the contents required to cover all topics. In addition, basic concepts of the cloud security were also included in the chapter in order to provide the fundamentals to understand the chapter. The studies analyzed emphasize the severity of the security issues related with virtualization techniques. The VM-to-VM attack surface was dramatically increased with the emergence of clouds, embracing now a large number of prominent security issues of cloud environments.

The analysis of the literature reflects a clear interest of researchers to address the cloud and cloud computing security issues. Many studies point out how critical it is to put effort in devising new security measures in order allow the development and adoption of cloud related technologies. In the meanwhile, cloud users might not be able to take full advantage of cloud applications in a fail-safe computing environment. History has proved that security should be a forefront priority and that the awareness on this area is partially motivated by issues and events faced along the way, which seems to apply in this case also.

References

1. Amazon. Amazon Web Services: Overview of Security Processes. http://s3.amazonaws.com/aws_blog/AWS_Security_Whitepaper_2008_09.pdf. White Paper. 2012
2. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M (2010) A view of cloud computing. Commun ACM 53(2010):50–58
3. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Zaharia M (2009) Above the clouds: a berkeley view of cloud computing. In: Technical report #UCB/EECS-2009-28. Electrical Engineering and Computer Sciences University of California.
4. Ateniese G, Di Pietro R, Mancini LV, Tsudik G (2008) Scalable and efficient provable data possession. In: Proceedings of the 4th international conference on security and privacy in communication networks (Istanbul, Turkey, 2008), 9:1–9:10.
5. Aviram A, Hu S, Ford B, Gummadi R (2010) Determinating timing channels in compute clouds. Proceedings of the ACM workshop on cloud computing security, Chicago, IL, USA, In, pp 103–108
6. Backstrom L, Dwork C, Kleinberg J (2007) Wherefore art thou R3579X?: anonymized social networks, hidden patterns, and structural steganography. In: Proceedings of the 16th international conference on world wide web, Banff, Alberta, Canada, pp 181–190.
7. Bahram S, Jiang X, Wang Z, Grace M, Li J, Srinivasan D, Rhee J, Xu D (2010) DKSM: subverting virtual machine introspection for fun and profit. In: 29th IEEE symposium on reliable distributed systems, New Delhi, India, pp 82–91.
8. Begum S, Khan MK (2011) Potential of cloud computing architecture. International conference on information and communication technologies, Karachi, Pakistan, In, pp 1–5
9. Behl A (2011) Emerging security challenges in cloud computing: An insight to cloud security challenges and their mitigation. World congress on information and communication technologies, Mumbai, India, In, pp 217–222
10. Bentounsi M, Benbernou S, Atallah MJ (2012) Privacy-preserving business process outsourcing. In: IEEE 19th international conference on web services, Honolulu, HI, USA, pp 662–663.
11. Bernstein D, Vij D (2010) Intercloud security considerations. In: IEEE 2nd international conference on cloud computing technology and science, Indianapolis, IN, USA, pp 537–544.
12. Boampong PA, Wahsheh LA (2012) Different facets of security in the cloud. In: Proceedings of the 15th communications and networking simulation symposium, Orlando, FL, USA, pp 5:1–5:7.
13. Bugiel S, Nürnberger S, Pöppelmann T, Sadeghi A-R, Schneider T (2011) AmazonIA: when elastiaddress snaps back. In: Proceedings of the 18th ACM conference on computer and communications security, Chicago, IL, USA, pp 389–400.
14. Carroll M, Kotzé P, Van der Merwe A (2011) Secure virtualization–benefits, risks and controls. CLOSER, Noordwijkerhout, Netherlands
15. Che J, Duan Y, Zhang T, Fan J (2011) Study on the security models and strategies of cloud computing. Procedia Eng 23(2011):586–593
16. Chen D, Zhao H (2012) Data security and privacy protection issues in cloud computing. International conference on computer science and electronics engineering, Hangzhou, China, In, pp 647–651
17. Chen Y, Paxson V, Katz RH (2010) What's new about cloud computing security? In: Technical report #UCB/EECS-2010-5. University of California, Berkeley, EECS Department
18. Chonka A, Xiang Y, Zhou W, Bonti A (2011) Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks. J Netw Compu Appl 34(2011):1097–1107
19. Chow R, Golle P, Jakobsson M, Shi E, Staddon J, Masuoka R, Molina J (2009) Controlling data in the cloud: Outsourcing computation without outsourcing control. In: Proceedings of the ACM workshop on cloud computing security. Chicago, IL, USA 2009:85–90
20. Chung H, Park J, Lee S, Kang C (2012) Digital forensic investigation of cloud storage services. Digital Investigation.

21. Cisco (2007) Cisco data center infrastructure 2.5 design guide. http://www.cisco.com/en/US/solutions/collateral/ns340/ns517/ns224/ns944/white_paper_c11-680202.pdf White Paper
22. Cisco (2011) Data center power and cooling. http://www.cisco.com/univercd/cc/td/doc/solution/dcidg21.pdf White Paper
23. Corbató FJ, Vyssotsky VA (1965) Introduction and overview of the multics system. In: Proceedings of the fall joint computer conference (Las Vegas, NV, USA, 1965), pp 185–196.
24. CSA (2011) Security guidance for critical areas of focus in cloud computing v3.0. https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf White Paper
25. CSA (2010) Top threats to cloud computing. https://cloudsecurityalliance.org/research/top-threats/. White paper
26. Dahbur K, Mohammad B, Tarakji AB (2011) A survey of risks, threats and vulnerabilities in cloud computing. In: Proceedings of the international conference on intelligent semantic web-services and applications, Amman, Jordan, pp 12:1–12:6.
27. Dhage SN, Meshram BB, Rawat R, Padawe S, Paingaokar M, Misra A (2011) Intrusion detection system in cloud computing environment. Proceedings of the international conference and workshop on emerging trends in technology, Mumbai, Maharashtra, India, In, pp 235–239
28. Dinesha HA, Agrawal VK (2012) Multi-level authentication technique for accessing cloud services. International conference on computing, communication and applications, Dindigul, Tamilnadu, India, In, pp 1–4
29. Ding X, Zhang L, Wan Z, Gu M (2011) De-anonymizing dynamic social networks. IEEE global telecommunications conference, Houston, USA, In, pp 1–6
30. Doroodchi M, Iranmehr A, Pouriyeh SA (2009) An investigation on integrating XML-based security into web services. In: 5th IEEE GCC conference exhibition, Kuwait City, Kuwait, pp 1–5.
31. Duncan AJ, Creese S, Goldsmith M (2012) Insider attacks in cloud computing. In: IEEE 11th international conference on trust, security and privacy in computing and communications, Liverpool, United Kingdom, pp 857–862.
32. Dykstra J, Sherman AT (2012) Acquiring forensic evidence from infrastructure-as-a-service cloud computing: exploring and evaluating tools, trust, and techniques. Digital Inv 9:S90–S98
33. ENISA (2009) Cloud computing: benefits, risks and recommendations for info-marion security. http://www.enisa.europa.eu/activities/risk-management/files/deliverables/cloud-computing-risk-assessment. White Paper
34. Firdhous M, Ghazali O, Hassan S (2011) A trust computing mechanism for cloud computing with multilevel thresholding. In: 6th IEEE international conference on industrial and information systems (Kandy, Sri Lanka, 2011), pp 457–461.
35. Foster I, Zhao Y, Raicu I, Lu S (2008) Cloud computing and grid computing 360-degree compared. Grid computing environments workshop, Austin, TX, USA, pp 1–10
36. Garfinkel T, Rosenblum M (2005) When virtual is harder than real: security challenges in virtual machine based computing environments. In: Proceedings of the 10th conference on hot topics in operating systems (Santa Fe, NM, USA, 2005), pp 20–20.
37. Gartner (2008) Assessing the security risks of cloud computing. http://cloud.ctrls.in/files/assessing-the-security-risks.pdf. White Paper
38. Gartner (2011) Summary report for gartner's top predictions for IT organizations and users, 2012 and beyond: control slips away. http://www.gartner.com/id=1861020. White Paper
39. Gonzalez N, Miers C, Redigolo F, Carvalho T, Simplicio M, Naslund M, Pourzandi M (2011) A quantitative analysis of current security concerns and solutions for cloud computing. In: IEEE 3rd international conference on cloud computing technology and science, Athens, Greece, pp 231–238.
40. Goodin D (2009) Webhost hack wipes out data for 100,000 sites. The Register.
41. Grobauer B, Walloschek T, Stocker E (2011) Understanding cloud computing vulnerabilities. IEEE Secur Privacy 9(2011):50–57
42. Gruschka N, Iacono LL (2009) Vulnerable cloud: SOAP message security validation revisited. IEEE Int Conf Web Services, Los Angeles, USA, pp 625–631

43. Habib SM, Ries S, Muhlhauser M (2011) Towards a trust management system for cloud computing. In: IEEE 10th international conference on trust. Security Privacy Comput Commun 2011:933–939
44. Hart J (2009) Remote working: managing the balancing act between network access and data security. Comput Fraud Security 2009:14–17
45. Hayes B (2008) Cloud computing. Commun ACM 51(2008):9–11
46. IDC (2009) New IDC IT cloud services survey: top benefits and challenges. http://blogs.idc.com/ie/?p=730. White Paper
47. IDC (2008) New IDC IT cloud services survey: top benefits and challenges. http://blogs.idc.com/ie/?p=210. White Paper
48. Idziorek J, Tannian M (2011) Exploiting cloud utility models for profit and ruin. IEEE Int Conf Cloud Comput, Washington, D.C., USA, pp 33–40
49. Idziorek J, Tannian M, Jacobson D (2011) Detecting fraudulent use of cloud resources. In: Proceedings of the 3rd ACM workshop on cloud computing security workshop (Chicago, IL, USA, 2011), pp 61–72.
50. Jasti A, Shah P, Nagaraj R, Pendse R (2010) Security in multi-tenancy cloud. IEEE international carnahan conference on security technology (San Jose, CA, USA, 2010), pp 35–41.
51. Jensen M, Gruschka N, Herkenhöner R (2009) A survey of attacks on web services. Comput Sci Res Dev 24(4):185–197
52. Jensen M, Gruschka N, Luttenberger N (2008) The impact of flooding attacks on network-based services. In: 3rd international conference on availability, reliability and security, Barcelona, Spain, pp 509–513.
53. Jensen M, Meyer C (2011) Expressiveness considerations of XML signatures. In: IEEE 35th annual computer software and applications conference workshop, Seoul, Korea, pp 392–397.
54. Jensen M, Schäge S, Schwenk J (2010) Towards an anonymous access control and accountability scheme for cloud computing. In: IEEE 3rd international conference on cloud computing, Miami, USA, pp 540–541.
55. Jensen M, Schwenk J, Gruschka N, Iacono LL (2009) On technical security issues in cloud computing. IEEE International conference on cloud computing, Bangalore, India, pp 109–116
56. Jin B, Wang Y, Liu Z, Xue J (2011) A trust model based on cloud model and bayesian networks. Proc, Environ Sci 11(Part A):452–459.
57. Kandukuri BR, Paturi VR, Rakshit A (2009) Cloud security issues. IEEE International conference on services computing, Bangalore, India, pp 517–520
58. Kaufman LM (2009) Data security in the world of cloud computing. IEEE Secur Privacy 7(2009):61–64
59. Khorshed MT, Ali ABMS, Wasimi SA (2012) A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing. Future Gen Comput Sys 28(2012):833–851
60. King ST, Chen PM (2006) SubVirt: implementing malware with virtual machines. IEEE Symposium on security and privacy. Oakland, CA, USA, p 327
61. Lee J-H, Park M-W, Eom J-H, Chung T-M (2011) Multi-level intrusion detection system and log management in cloud computing. In: 13th international conference on advanced communication technology, Phoenix Park, South Korea, pp 552–555.
62. Li H-C, Liang P-H, Yang J-M, Chen S-J (2010) Analysis on cloud-based security vulnerability assessment. In: IEEE 7th international conference on e-business engineering, Shanghai, China, pp 490–494.
63. Liu H (2010) A new form of DoS attack in a cloud and its avoidance mechanism. Proceedings of the ACM workshop on cloud computing security workshop, Chicago, USA, In, pp 65–76
64. Lombardi F, Pietro RD (2011) Secure virtualization for cloud computing. J Network Comput Appl 34(2011):1113–1122
65. Luo S, Lin Z, Chen X, Yang Z, Chen J (2011) Virtualization security for cloud computing service. International conference on cloud and service computing, Washington, USA, pp 174–179
66. Mansfield-Devine S (2008) Danger in the clouds. Netw Secur 2008:9–11

67. Mathisen E (2011) Security challenges and Solutions in Cloud Computing. In: Proceedings of the 5th IEEE international conference on digital ecosystems and technologies, Daejeon, South Korea, pp 208–212.
68. McGraw G (2004) Software Security. IEEE Secur Privacy 2(2004):80–83
69. McIntosh M, Austel P (2005) XML signature element wrapping attacks and countermeasures. Proceedings of the workshop on secure web services, Fairfax, USA, In, pp 20–27
70. Modi C, Patel D, Borisaniya B, Patel H, Patel A, Rajarajan M (2012) A survey of intrusion detection techniques in cloud. J Netw Comput Appl.
71. Monfared AT, Jaatun MG (2011) Monitoring intrusions and security breaches in highly distributed cloud environments. In: IEEE 3rd international conference on cloud computing technology and science, Athens, Greece, pp 772–777.
72. Morsy MA, Grundy J, Müller I (2010) An analysis of the cloud computing security problem. Proceedings of Asia pacific software engineering confernce cloud workshop, Sydney, Australia, In, pp 1–6
73. Narayanan A, Shmatikov V (2009) De-anonymizing social networks. In: 30th IEEE symposium on security and privacy, Oakland, USA, pp 173–187.
74. NIST (2012) NIST cloud computing program. http://www.nist.gov/itl/cloud/. White Paper
75. Oberheide J, Cooke E, Jahanian F (2008) Empirical exploitation of live virtual machine migration. Proceedings of the black hat conference, Washington, USA, In
76. Okamura K, Oyama Y (2010) Load-based covert channels between Xen virtual machines. Proceedings of the ACM symposium on applied computing, Sierre, Switzerland, In, pp 173–180
77. OWASP (2010) The then most critical web application security risks. http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202010.pdf. White Paper
78. Patel A, Taghavi M, Bakhtiyari K, Júnior JC (2012) A systematic review. J Netw Comput Appl Intrusion Detec Prev Sys Cloud Comput.
79. PCI (2012) PCI SSC data security standards overview. https://www.pcisecuritystandards.org/security_standards/index.php. White Paper
80. Pfaff B, Pettit J, Koponen T, Amidon K, Casado M, Shenker S (2009) Extending networking into the virtualization layer. In: Proceedings of the 8th ACM workshop on hot topics in Networks.
81. Pianese F, Bosch P, Duminuco A, Janssens N, Stathopoulos T, Steiner M (2010) Toward a cloud operating system. IEEE/IFIP network operations and management symposium workshop, Osaka, Japan, In, pp 335–342
82. Rahaman MA, Schaad A, Rits M (2006) Towards secure SOAP message exchange in a SOA. In: Proceedings of the 3rd ACM workshop on secure web services, Alexandria, USA, pp 77–84.
83. Ramgovind S, Eloff MM, Smith E (2010) The management of security in cloud computing. Information security for South Africa, Johannesburg, South Africa, pp 1–7
84. Riquet D, Grimaud G, Hauspie M (2012) Large-scale coordinated attacks: Impact on the cloud security. In: 6th international conference on innovative mobile and internet services in ubiquitous computing, Palermo, Italy, pp 558–563.
85. Ristenpart T, Tromer E, Shacham H, Savage S (2009) Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proceedings of the 16th ACM conference on computer and communications security, Chicago, USA, pp 199–212.
86. Roberts JC, Al-Hamdani W (2011) Who can you trust in the cloud?: A review of security issues within cloud computing. Proceedings of the information security curriculum development conference, Kennesaw, GA, In, pp 15–19
87. Rocha F, Correia M (2011) Lucy in the sky without diamonds: stealing confidential data in the cloud. In: IEEE/IFIP 41st international conference on dependable systems and networks workshops, pp 129–134.
88. Rong C, Nguyen ST, Jaatun MG (2012) A survey on security challenges in cloud computing. Comput Elect Engi Beyond Lightning.

89. Rutkowska J (2008) Subverting vistaTM Kernel for fun and profit. Black Hat Conv, Washington, D.C., USA
90. Sadashiv N, Kumar SMD (2011) Cluster, grid and cloud computing: a detailed comparison. In: 6th international conference on computer science education, SuperStar Virgo, Singapore, pp 477–482.
91. Santos N, Gummadi KP, Rodrigues R (2009) Towards trusted cloud computing. Proceedings of the conference on hot topics in cloud computing, San Diego, CA, USA, In
92. Sengupta S, Kaulgud V, Sharma VS (2011) Cloud computing security–trends and research directions. IEEE World Congress Services, Washington D.C., 2011, pp 524–531.
93. Sloan K (2009) Security in a virtualised world. Netw Secur 2009(2009):15–18
94. SplashData (2012) Scary logins: worst passwords of 2012 and how to fix them. http://splashdata.com/press/PR121023.htm. White Paper
95. Subashini S, Kavitha V (2011) A survey on security issues in service delivery models of cloud computing. J Netw Comput Appl 34(2011):1–11
96. Suzaki K, Iijima K, Yagi T, Artho C (2011) Memory deduplication as a threat to the guest OS. In: Proceedings of the 4th European workshop on system security, New York, USA, vol 1:1–1:6.
97. Suzaki K, Iijima K, Yagi T, Artho C (2011) Software side channel attack on memory deduplication. 23rd ACM symposium on operating systems principles.
98. Takabi H, Joshi JBD, Ahn G (2010) Security and privacy challenges in cloud computing environments. IEEE Secur Privacy 8(2010):24–31
99. Taylor M, Haggerty J, Gresty D (2011) Lamb D (2011) Forensic investigation of cloud computing systems. Netw Secur 2011:4–10
100. Toubiana V, Nissenbaum H (2011) Analysis of Google logs retention policies. J Priv Confidentiality 3(2011):3–26
101. Townsend M (2009) Managing a security program in a cloud computing environment. Information security curriculum development conference, Kennesaw, GA, USA, pp 128–133
102. Tripathi A, Mishra A (2011) Cloud computing security considerations. IEEE international conference on signal processing, communications and computing, Xi'an, Shaanxi, China, In, pp 1–5
103. Tsai H-Y, Siebenhaar M, Miede A, Huang Y, Steinmetz R (2012) Threat as a service? virtualization's impact on cloud security. IT Professional 14(2012):32–37
104. Vaquero LM, Rodero-Merino L, Morán D (2011) Locking the sky: a survey on IaaS cloud security. Computing 91(2011):93–118
105. Vascellaro JE (2009) Google discloses privacy glitch. http://blogs.wsj.com/digits/2009/03/08/1214/
106. Viega J (2009) Cloud computing and the common man. Computer 42(2009):106–108
107. Wang C, Ren K, Lou W, Li J (2010) Toward publicly auditable secure cloud data storage services. IEEE Network 24(2010):19–24
108. Wang C, Wang Q, Ren K, Lou W (2009) Ensuring data storage security in cloud computing. In: 17th international workshop on quality of service, Charleston, SC, USA, pp 1–9.
109. Ward M (2009) Facebook users suffer viral surge. http://news.bbc.co.uk/2/hi/technology/7918839.stm
110. Wei J, Zhang X, Ammons G, Bala V, Ning P (2009) Managing security of virtual machine images in a cloud environment. Proceedings of the ACM workshop on cloud computing security, New York, USA, In, pp 91–96
111. Xiao Z, Xiao Y (2012) Security and privacy in cloud computing. IEEE Commun Surv Tutorials 2012:1–17
112. Yang K, Jia X (2012) Data storage auditing service in cloud computing: challenges. Methods Opportunities World Wide Web 15(2012):409–428
113. Yu H, Powell N, Stembridge D, Yuan X (2012) Cloud computing and security challenges. In: Proceedings of the 50th annual southeast regional conference, Tuscaloosa, USA, pp 298–302.
114. Zhang F, Huang Y, Wang H, Chen H, Zang B (2008) PALM: security preserving VM live migration for systems with VMM-enforced protection. In: 3rd Asia-Pacific trusted infrastructure technologies conference, Wuhan, China, pp 9–18.

115. Zhou M, Zhang R, Xie W, Qian W, Zhou A (2010) Security and privacy in cloud computing: a survey. In: 6th international conference on semantics knowledge and grid (Ningbo, China, 2010), pp 105–112.
116. Zissis D, Lekkas D (2012) Addressing cloud computing security issues. Future Gener Comput Sys 28(2012):583–592
117. Zou B, Zhang H (2011) Toward enhancing trust in cloud computing environment. In: 2nd international conference on control, instrumentation and automation, Shiraz, Iran, pp 364–366.

Secure Data Sharing in the Cloud

Danan Thilakanathan, Shiping Chen, Surya Nepal and Rafael A. Calvo

1 Introduction

Cloud systems [1, 2] can be used to enable data sharing capabilities and this can provide an abundant of benefits to the user. There is currently a push for IT organisations to increase their data sharing efforts. According to a survey by InformationWeek [3], nearly all organisations shared their data somehow with 74 % sharing their data with customers and 64 % sharing with suppliers. A fourth of the surveyed organisations consider data sharing a top priority. The benefits organisations can gain from data sharing is higher productivity. With multiple users from different organisations contributing to data in the Cloud, the time and cost will be much less compared to having to manually exchange data and hence creating a clutter of redundant and possibly out-of-date documents. With social networking services such as Facebook, the benefits of sharing data are numerous [4] such as the ability to share photos, videos, information and events, creates a sense of enhanced enjoyment in one's life and can enrich the lives of some people as they are amazed at how many people are interested in their life and well-being. For students and group-related projects, there has been a major importance for group collaborative tools [5]. Google Docs provides data sharing capabilities as groups of students or teams working on a project can share documents and can collaborate with each other effectively. This allows higher productivity compared to previous methods of continually sending updated versions of a document to members of the group via email attachments. Also in modern healthcare environments, healthcare providers are willing to store and share electronic medical records via the Cloud and hence remove the geographical dependence between healthcare provider and patient [6]. The sharing of medical data allows

D. Thilakanathan (✉) · R. A. Calvo
Department of Electrical Engineering, The University of Sydney, Sydney, NSW 2006, Australia
e-mail: thilakanathan@hotmail.com

S. Chen · S. Nepal
CSIRO ICT Centre, Cnr Vimiera and Pembroke Rodas, Marsfield, NSW 2122, Australia

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_2,

the remote monitoring and diagnosis of patients without the patient having to leave their house. In one particular scenario, a patient can connect sensors to monitor their ECG to detect any heart problems [7]. They can then run an app on a smartphone device which receives ECG data from the sensors via Bluetooth. The app can then periodically send ECG data to the Cloud. Any authorised doctor or nurse can then get the ECG data via the Cloud without having to visit the patient hence saving costs and time. Therefore, data sharing becomes such a useful feature to implement in Cloud-based environments.

The Cloud however is susceptible to many privacy and security attacks [8, 9]. As highlighted in [10], the biggest obstacle hindering the progress and the wide adoption of the Cloud is the privacy and security issues associated with it. According to a survey carried out by IDC Enterprise Panel [11] in August 2008, Cloud users regarded security as the top challenge with 75 % of surveyed users worried about their critical business and IT systems being vulnerable to attack. Evidently, many privacy and security attacks occur from within the Cloud provider themselves [12] as they usually have direct access to stored data and steal the data to sell to third parties in order to gain profit. There are many examples of this happening in the real world as highlighted in [13]. In today's world, there is a strong need to share information to groups of people around the world. Since the Cloud is riddled with so many privacy issues, many users are still apprehensive about sharing their most critical data with other users.

Some of major requirements of secure data sharing in the Cloud are as follows. Firstly the data owner should be able to specify a group of users that are allowed to view his or her data. Any member within the group should be able to gain access to the data anytime, anywhere without the data owner's intervention. No-one, other than the data owner and the members of the group, should gain access to the data, including the Cloud Service Provider. The data owner should be able to add new users to the group. The data owner should also be able to revoke access rights against any member of the group over his or her shared data. No member of the group should be allowed to revoke rights or join new users to the group.

One trivial solution to achieving secure data sharing in the Cloud is for the data owner to encrypt his data before storing into the Cloud, and hence the data remain information-theoretically secure against the Cloud provider and other malicious users. When the data owner wants to share his data to a group, he sends the key used for data encryption to each member of the group. Any member of the group can then get the encrypted data from the Cloud and decrypt the data using the key and hence does not require the intervention of the data owner. However, the problem with this technique is that it is computationally inefficient and places too much burden on the data owner when considering factors such as user revocation. When the data owner revokes access rights to a member of the group, that member should not be able to gain access to the corresponding data. Since the member still has the data access key, the data owner has to re-encrypt the data with a new key, rendering the revoked member's key useless. When the data is re-encrypted, he must distribute the new key to the remaining users in the group and this is computationally inefficient and places too much burden on the data owner when considering large group sizes

that could be in excess of millions of users. Hence this solution is impractical to be deployed in the real-world for very critical data such as business, government and/or medical related data. In this article, we review existing literature on methods of achieving data sharing in the Cloud that is both secure and efficient.

The rest of this article is organised as follows. Section 2 provides a summary of the existing related literature surveys. Section 3 gives a brief review of some security related definitions and concepts. Section 4 discusses the privacy issues of Cloud computing and what is currently being done to solve those issues. Section 5 provides an in-depth discussion on key management in the Cloud. Section 6 provides a thorough discussion on secure data sharing techniques in the Cloud. Section 7 will review and provide an analysis on the literature. Section 8 concludes the review article.

2 Related Work

This section aims to present a summary of existing review articles related to secure data sharing in the Cloud. The review articles and surveys presented in this section do not focus specifically on secure data sharing in the Cloud, rather the main requirements that will enable it. The study of secure data sharing in the Cloud is fairly new and has become increasingly important with the advancements and growing popularity of the Cloud as well as the growing need to share data between people. We categorise the existing review articles in two aspects: data sharing and Cloud security.

There have been a number of reviews on security and privacy in the Cloud. Xiao and Xiao [14] identifies the five concerns of Cloud computing; confidentiality, integrity, availability, accountability, and privacy and thoroughly reviews the threats to each of the concerns as well as defense strategies. Chen and Zhao [15] outlines the requirements for achieving privacy and security in the Cloud and also briefly outlines the requirements for secure data sharing in the Cloud. Zhou [16] provided a survey on privacy and security in the Cloud focusing on how privacy laws should also take into consideration Cloud computing and what work can be done to prevent privacy and security breaches of one's personal data in the Cloud. Wang et al. [17] explored factors that affect managing information security in Cloud computing. It explains the necessary security needs for enterprises to understand the dynamics of information security in the Cloud. Wang [18] carried out a study on the privacy and security compliance of Software-As-A-Service (SaaS) among enterprises through pilot testing privacy/security compliance. They then carry out analysis work on the measurements to check whether SaaS complies with privacy and security standards. The method does not however take into account other Cloud models such as Platform-As-A-Service (PaaS) and in particular Infrastructure-As-A-Service (IaaS), as needed for data sharing. Oza et al. [19] carried out a survey on a number of users to determine the user experience of Cloud computing and found that the main issue of all users was trust and how to choose between different Cloud Service Providers. This is also highlighted in [12] as it states, "Although researchers have identified numerous security threats to the Cloud, malicious insiders still represent a significant concern." There

are many examples [13] of insider attacks such as Google Docs containing a flaw that inadvertently shared user documents, MediaMax going out of business in 2008 after losing 45 % of stored client data due to administrator error, Salesforce.com leaking a customer list and falling victim to phishing attacks on a number of occasions. It's clear from many of the reviews, that the Cloud is very susceptible to privacy and security attacks and currently there is on-going research that aims to prevent and/or reduce the likelihood of such attacks.

The importance of data sharing and the need to ensure privacy and security is discussed in a number of existing articles. Saradhy and Muralidhar [20] review the impact of the Internet on data sharing across many different organisations such as government agencies and businesses. They classify data sharing into data dissemination, query restriction, and record matching. They also provide a framework for secure and useful sharing of data on the internet. Butler [21] describes the issues of data sharing on the Internet where sharing information can allow users to infer details about users. This is useful as it raises awareness to organisations that the data they choose to share with the public can still raise privacy issues and does not guarantee the confidentiality of its users. Mitchley [22] describes the benefits of data sharing from a banking perspective and highlights the privacy issues still affecting it. Feldman et al. [23] discuss the important benefit of data sharing in terms of public health, in particular for education and professional development. Geoghegan [24] discuss a list of organisations that effectively and secure share information via the Cloud. However, it doesn't discuss the methodologies the organisations use to secure data or the downside of these organisations. There is also literature that focus on one aspect of security as well as data sharing; access control. Access control can be used to authorise a subset of users to view confidential data provided they have the right permission. Sahafizadeh and Parsa [25] survey a number of different access control models and evaluates its effectiveness. The survey however, is limited to only software systems and does not take into consideration Cloud systems.

Table 1 shows a summary of the related work. The table categorises the related work in two aspects; Cloud security and Data sharing. The table depicts whether the related work addresses the threats, defense strategies and requirements related to the Cloud or data sharing. The table also depicts whether the related work addresses the impact of the Cloud and/or data sharing in real-world scenarios.

The aim of this paper is to present a comprehensive review of private and secure data sharing in Cloud computing.

3 Privacy Issues in the Cloud

3.1 Privacy Issues

Privacy has many definitions in literature. Some examples of the different definitions of privacy are "being left alone", "the control we have over information about our-

Table 1 Summary of related work

	Cloud security	Data sharing	Threats	Defense strategies	Requirements	Impact on society
Xiao an Xiao [14]	Y	N	Y	Y	N	Y
Chen and Zhao [15]	Y	Y	Y	N	Y	Y
Zhou [16]	Y	N	Y	Y	Y	Y
Wang et al. [17]	Y	N	N	Y	Y	Y
Wang [18]	Y	N	N	Y	Y	N
Oza et al. [19]	Y	N	Y	N	Y	Y
Saradhy and Muralidhar [20]	N	Y	Y	Y	Y	Y
Butler [21]	N	Y	Y	N	Y	Y
Mitchley [22]	N	Y	Y	N	N	Y
Feldman et al. [23]	N	Y	N	N	N	Y
Geoghegan [24]	N	Y	N	N	N	Y
Sahafizadeh and Parsa [25]	N	Y	N	N	Y	Y

Y yes, *N* no

selves" and also "the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others" [26] to name a few. The Organization for Economic Cooperation and Development (OECD) [15] defines it as "any information relating to an identified or identifiable individual (data subject)". The American Institute of Certified Public Accountants (AICPA) and the Canadian Institute of Chartered Accountants (CICA) in the Generally Accepted Privacy Principles (GAPP) standard [15] is "The rights and obligations of individuals and organizations with respect to the collection, use, retention, and disclosure of personal information." From these definitions it is clear that a person has some level of control of what they want to disclose about themselves and want to keep the rest of their information kept secret. Privacy should not be assumed to have the same meaning as confidentiality. Confidentiality is allowing only authorised user's to gain access to that information and no-one else. We briefly explain the need of privacy and confidentiality in a number of fields.

Privacy and Confidentiality of data in Healthcare: In the context of healthcare, patients reveal their health-related information to healthcare professionals in order to diagnose and treat illnesses [27]. The Health Insurance Portability and Accountability Act (HIPAA) [28] provides federal protection of an individual's personal health information and gives individual's rights to their information. The HIPAA Privacy Rule provides protection of patient's personal health information and how external entities such as doctors and nurses can gain access to the patient's data with the patient's consent. As [27] argues, since the patient decides to share their data with one or more healthcare professionals, their data is no longer private, but confidential.

Privacy and Confidentiality of data in Social Networking: Social networking has changed the lives of today's generation. There are many social networking sites with millions of users communicating with each other. Some examples are Facebook, Twitter, MySpace, Blogger, Flickr, digg, YouTube and the list goes on. Internet

privacy has been determined as the "right to be left alone" [29]. The technology that is built to support social networking does not effectively support privacy and may even sell personal information about the individual to third parties and it is mainly up to the individual to disclose information while maintaining privacy. The individual needs to make sure that they do not unknowingly disclose personal information about themselves. Simply disclosing their age, suburb and nationality is enough for malicious users to identify the person. Facebook had undergone scrutiny in the past for not strengthening its privacy measures on user profiles as private photos could still be viewed by non-private viewers through a friend-of-a-friend by simply having a friend comment on it [30].

Privacy and Confidentiality of data in Government: Nearly all governments collect information about its citizens and residents such as education, finance, gender, loans, earnings, medical costs, criminal offences and so on [31]. Governments also release data to the open public for its citizens to view. This may not guarantee the privacy of its citizens as some user may be able to infer information about a particular user through government data. In the United States for example, the Privacy Act of 1974 aims to protect an individual's privacy [32]. According to the Act, individuals have the right to see information the government has about them, modify or remove incorrect information, and also sue the government for violations of the Act including but not limited to, unauthorized access of personal information. Governments need to keep data private from other governments too [33] as the results can be devastating if information is leaked such as the WikiLeaks controversy [34].

Privacy and Confidentiality of data in Education: Schools usually collect all students personal and health information. These include name, phone, address, contact details, finance details, medical history and family history to name a few. It is usually strongly implied that schools keep this information confidential and private [35]. Failure to keep student personal information confidential can result in safety consequences for the student.

Privacy and Confidentiality of data in Corporations: Major businesses and organisations also require privacy and confidentiality of their data. Leakage of sensitive information can result in revenue loss for a company even to the point of shutting down.

3.2 Types of Attacks on the Cloud

There are a number of types of privacy and security attacks in the Cloud. The following contains a summary of the common types of attacks that may occur in the Cloud.

- *XML Signature Wrapping Attacks*—Using different kinds of XML signature wrapping attacks, one can completely take over the administrative rights of the Cloud user and create, delete, modify images as well as create instances [36].

- *Cross site scripting attacks*—Attackers can inject a piece of code into web applications to bypass access control mechanisms. Researchers found this possible with Amazon Web Services [36] in November 2011. They were able to gain free access to all customer data, authentication data, tokens as well as plaintext passwords.
- *Flooding Attack Problem* —Provided a malicious user can send requests to the Cloud, he/she can then easily overload the server by creating bogus data requests to the Cloud [37]. The attempt is to increase the workload of the Cloud servers by consuming lots of resources needlessly.
- *Denial-of-Service Attacks*—Malicious code is injected into the browser to open many windows and as a result deny legitimate users access to services.
- *Law Enforcement Requests*—When the FBI or government demand a Cloud Service Provider access to its data, the Cloud Service Provider is least likely to deny them. Hence, an inherent threat to user privacy and confidentiality of data.
- *Data Stealing Problem*—A term used to describe the stealing of a user account and password by any means [37] such as through brute-force attacks or over-the-shoulder techniques. The privacy and confidentiality of user's data will be severely breached. A common mechanism to prevent such attacks is to include an extra value when authenticating. This value can be distributed to the right user by SMS and hence mitigate the likelihood of data confidentiality issues.

3.3 The Motives of a Malicious User

While there is many literature on what can be done to secure a system against attackers, very little discusses the types of attackers and their motivations for carrying out such attacks. In reality, there are many different types of attackers with different reasons to attack users [38, 39]. The following contains some examples.

- *To steal valuable data*—Hackers love to steal data as some data stored in the internet are valued millions of dollars. With access to valuable data, they can then generate revenue, for example, WikiLeaks [34].
- *To cause controversy*—Some attackers purely love the thrill and excitement of causing chaos and the internet, and similarly the Cloud, is one of the best mediums to target mainly because of the popularity of the internet as well as it being more likely to steal data over the internet in comparison to a personal computer system.
- *To get revenge*—Former workers who were recently stripped of their position at an organisation may express their dissatisfaction by hacking the organisation's network. When an organisation makes use of the Cloud, this becomes all too easy for the former employee and there have been many cases of this happening in the real-world. For instance, there was the case of a former employee who managed to get access to the Cloud provider's server and deleted an entire season of a children's TV show [12].
- *To help*—A hacker, in contrast, may also try to help an organisation by identifying the security flaws in their system. A hacker may be confident enough to bypass

the existing security protocol and implant his or her own mechanisms to expose the protocol.
- *To prove intellect and gain prestige*—Attackers may also want to show off their skills and gain prestige among their social skills if they were able to hack a large organisation with solid security mechanisms. Some hackers make a career out of hacking organisations.
- *Are just curious*—Some hackers are curious to learn something about a company and/or organisation. These kinds of hackers don't usually have malicious intent as they may not be aware of breaking security rules however it does not mean these hackers are less dangerous whatsoever.

3.4 Examples of Real World Issues

There are many examples of real world privacy and security issues that have affected the Cloud. These issues have provided a barrier to the worldwide adoption of the Cloud. We present these issues as a list.

- In 2007, Salesforce.com leaked customer contact lists after an employee revealed the list to a phisher, and in turn allowed scammers to target phishing attacks against Salesforce customers [40].
- In April 2011, Sony was involved in a massive security blunder that potentially gave away 100 million credit card numbers. Hackers claimed to have stolen millions of credit card numbers from Sony's PlayStation Network [41].
- Google revealed in June 2011 that hackers from China stole passwords and attempted to break into email accounts to steal information [42]. More than 100 people were affected and included senior government officials. People started to argue whether this, and the Sony incident was start of the downfall of Cloud computing [43].
- Hotmail and Yahoo Mail users were also targetted in phishing attacks [44, 45]. The attacks involved a user either clicking a malicious link in the email or even viewing the email itself which would then run malicious code and attempt to compromise the user's account.
- Google Docs contained a flaw that inadvertently shared user docs with unauthorised users [13]. Other users could access and edit docs without the Google docs owner permission.
- There was also the issue of MegaUpload leaving its millions of legitimate users in cyber-limbo [46]. MegaUpload was a site where people could share files. Unfortunately due to the amount of illegal content such as pirated films and television shows, the site was forced to shut down in early 2012.
- A Distributed Denial-of-Service (DDoS) attack on Amazon Web Services forced many companies to shut down temporarily, such as Bitbucket [47].
- Facebook was the target of phishing attacks in early 2012 which attempted to steal user accounts and learn financial information [48]. Once accounts were stolen, the

user's profile would be locked out and the profile picture would change. In fact, Facebook has been the target of a number of phishing attacks such as Ramnit [49] which affected upto 45,000 users.

Each of these attacks contributes heavily to user suspicion and trust of storing sensitive data in the Cloud. From this list, it is clear why users are apprehensive about storing their most sensitive data in the Cloud and in order to gain trust of using the Cloud to store critical data, mechanisms need to be implemented to guarantee data is kept both confidential and secure from unauthorised users.

3.5 Recommended Guidelines for Private and Secure Cloud

According to [50], the above issues may have the following impacts on the Cloud:

Governance—Organisations usually have standards, practices, protocols, policies and procedures which employees must abide by and this can cover application development, testing, implementation, monitoring and so on. When an organisation makes use of Cloud services, there is always the possibility that employees bypass these rules, as there is a lack of organisational rules regarding the Cloud.

Compliance—Refers to an organisation's responsibility to operate in agreement with established laws, regulations, etc. There are a number of privacy and security laws within different countries, states, and so on and when using the Cloud, one has to consider whether they are likely to breach any privacy or security law as data stored in the Cloud is usually stored in multiple locations around the world, at times without the knowledge of the user.

Trust—It is a well-known fact that when a user or organisation chooses to outsource their data to the Cloud, they relinquish full control of their data and provide a high level of trust to the Cloud provider. As discussed in the introduction as well as in the next section, most data privacy and security attacks come from insider attacks. The Cloud provider usually has direct access to data and hence is more likely to steal data for illegal purposes. In terms of trust, there is also the issue of data ownership such as who owns the data, and contracts specifying whether the Cloud has some or no access to parts of its data.

Architecture—The architecture of the Cloud needs to be designed in a way to prevent privacy and security attacks. For instance, IaaS Cloud providers can provide Virtual Machine Images to consumers. An organisation which makes use of these images, may store very critical data. An attacker may examine the images to see whether they leak information. An attacker may also supply a corrupted virtual machine image to users and hence steal confidential data. It is important that the architecture of the Cloud is developed such that it ensures privacy and security as attackers are always on the lookout for security holes in Cloud architecture.

Identity and Access Management—As data sensitivity and privacy is becoming an ever-increasing issue of organisations, the identity and authorisation framework

present in the organisation may not extend into the Cloud and malicious users may be able to gain unwarranted access to data they are not allowed to.

Software Isolation—With multi-tenant Cloud computing architectures, computations for different consumers are carried out in isolation even if the software remains in a single software stack. Applications running in the Cloud are susceptible to attack and compromise and hence isolation is needed to prevent such attacks.

Data Protection—Data stored in a public cloud usually reside with other data from other organisations. When an organisation places their sensitive data in a public cloud, they must account for the possible privacy and security attacks by ensuring proper access control mechanisms such as encryption. Since data is stored "in the open", this provides a world of opportunities for malicious users to steal data. Similar concerns exist when data is in transit.

Availability—As defined in the NIST Security and Privacy Guidelines [50], availability is the extent to which an organisation's full set of computational resources is accessible and usable. Attacks such as Denial-of-Service attacks, server downtime, natural disasters affect availability and can affect stored data and more importantly causes downtime which affects an organisation greatly.

Incident Response—An incident response is an organised method of dealing with the consequences a security attack. The Cloud containing many layers such as application, operating system, network, database and so on, and a log is generated of any event as part of its intrusion detection system. Such complexity in its layers means it will take many hours to identify an attack in the Cloud.

4 Protection from Privacy and Security Attacks

In this section, we discuss what is currently being done to protect and/or mitigate the privacy and security attacks on the Cloud.

Currently, there is on-going research on how to protect the confidentiality and security of data stored in the Cloud. Cavoukian [51] proposes implementing security as a service in the Cloud using a discretion algorithm and also implementing an intrusion detection system for the Cloud. Sabahi [52] argues the need for a flexible and user-centric identity management such that in the future a user will not have to re-enter credentials for a website and can rely on an identity service to manage website access.

In order to protect a user's data confidentiality, some form of access control needs to be implemented in the Cloud. Access control should allow a user to choose who can view his data and who shouldn't. Access Control Lists (ACLs) were originally used [53], however, it was not effective as it was too coarse-grained and was not scalable; one of the primary features of the Cloud.

An alternative and effective access control technique is encryption. Encrypting data ensures data is protected from unauthorised users. There are two types of encryption; symmetric and asymmetric encryption. In symmetric encryption, a key is used to encrypt the data to make it virtually unreadable. The same key is also used to

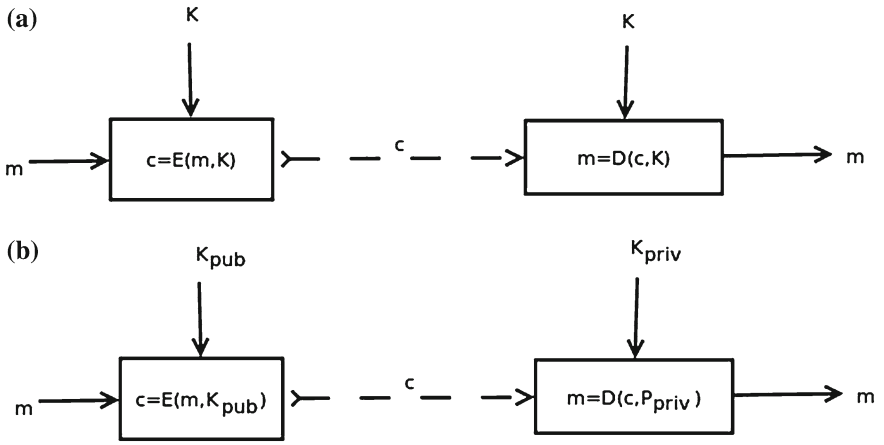


Fig. 1 Asymmetric encryption

convert the unreadable ciphertext to its original plaintext. This key must be kept confidential with the data owner. In asymmetric encryption, a public and private key is used to encrypt and decrypt data. A user encrypts the data using another person's public key. The other person then uses his private key to decrypt the data. The public key can be broadcast to the world but the private key must remain confidential with the user.

When involving data in the Cloud, encryption thus becomes crucial. Many works in literature suggest the need for encrypting data in the Cloud in some form or another. Huang et al. [13] states that encryption must occur in transit, at rest and on backup media. Gentry [54] proposes the use of homomorphic encryption to keep data secure and confidential. With homomorphic encryption, it is possible to perform operations such as querying and searching on encrypted data without ever having to decrypt the data and hence exposing privacy. Yao et al. [55] propose a system called 'TrustStore' which encrypts and partitions data on the client side and sends each partition to different Cloud storage providers. This greatly enhances the confidentiality of data as the chance of compromising two or more storage providers is low. However, it doesn't handle the case of data sharing and collaboration, which is the focus of this paper.

When considering data sharing and collaboration, simple encryption techniques do not suffice, especially when considering key management. To enable secure and confidential data sharing and collaboration in the Cloud, there needs to first be proper key management in the Cloud. This will be explained in detail in a later section.

A few research problems currently exist such as how do we manage and distribute keys for each granted user? How do we revoke their rights from accessing the data? Once a user is revoked rights, is it possible for a user to rejoin the group with the same rights?

We now discuss and review literature based on key management in the Cloud, which will later follow on to data sharing and collaboration.

5 Secure Data Sharing in the Cloud

In this section, we discuss the growing need for data sharing and the benefits of data sharing via the Cloud. We list the requirements of data sharing in the Cloud followed by the traditional approach to sharing data via the Cloud and why this isn't effective. We also discuss the key management problem and review a number of works that address this problem. We then review recent works that aim to provide private and secure data sharing in the Cloud and discuss the latest techniques used to achieve this.

5.1 Why Data Sharing is Important

Data sharing is becoming increasingly important for many users and sometimes a crucial requirement, especially for businesses and organisations aiming to gain profit. Historically, many people viewed the computer as "impersonal giants" who threatened to cut jobs of many people through automation. However, in recent times, it has been welcomed by a huge number of people as it has become significantly social [56]. It is thus not surprising that more and more people are demanding data sharing capability on their phones, computers and even recently Smart TVs.

People love to share information with one another. Whether it is with friends, family, colleagues or the world, many people benefit greatly through sharing data. Some of the benefits include:

- *Higher productivity*: Businesses get more work done as well as making collaboration with peers much more efficient and hence is key to satisfying their business goals. Hospitals also benefit from data sharing and this has led to the lowering of healthcare costs [57]. Students also benefit when working on group projects, as they are better able to collaborate with members and get work done more efficiently.
- *More enjoyment*: Many people of any age, gender or ethnicity can connect with friends, family and colleagues to share their experiences in life as well as catch up with others via social networking sites such as Facebook or MySpace. Employees and enterprise users can share their experiences through sites like Yammer. People can also share videos on YouTube or photos on Flickr, which can provide greater enjoyment with some people. In the past, connecting to a loved one in a different country was not possible except through letters. Hence social data sharing generally provides people with a rich experience as the sharing of personal information can provide people with deeper and stronger relationships.
- *To voice opinions*: Some people prefer to share information to the world in order to voice an opinion. Many people want to be heard and use social networking sites to

promote their opinion, which was not possible unless they formed protests. People are now using social networking sites such as Facebook, Twitter and YouTube to raise awareness about real issues in the world. Although, some campaigns have led to violent protests, online campaigns usually inform people of issues and encourage people to help a cause.

Data sharing is becoming increasingly prevalent in many industries and organisations. Hospitals are now benefitting from data sharing as this provides better, safer care of patients. There is now no need to repeat medical history every time a new health professional is consulted which means no more unnecessary tests. Hence, the health professional gets a more complete picture of medical history [57]. There is also a strong focus for the sharing of research data [58]. According to Feldman et al. [59], there is growing support for the sharing of research data in order to accelerate the pace of scientific discovery. Such sharing will allow for more rapid translation of science to practice. Financial institutions also benefit from data sharing and benefits include better customer support and better understanding of the needs of the customer [60]. Shared data can be used to improve modeling, analysis and risk tools.

With the advancements in Cloud computing, there is now a growing focus on implementing data sharing capabilities in the Cloud. With the ability to share data via the Cloud, the number of benefits increases multifold. As businesses and organisations are now outsourcing data and operations to the Cloud, they benefit further with the ability to share data between other businesses and organisations. Employees also benefit as they can share work and collaborate with other employees and can also continue working at home or any other place such as the library. They don't need to worry about losing work as it is always in the Cloud. With social users, the ability to share files, including documents, photos and videos with other users provides great benefit to them. The shutdown on the MegaUpload website where people could upload and share files with other people in the Cloud left millions of users around the world devastated [46]. The amount of illegal contents such as pirated films and full television shows forced the website to be shut down. This exemplifies the strong need for data sharing in the Cloud.

However, the main problem with data sharing in the Cloud is the privacy and security issues. As discussed in Sect. 4, the Cloud is open to many privacy and security attacks, which make many users wary of adopting Cloud technology for data sharing purposes. The work done to prevent the privacy and security issues of the Cloud as discussed in Sect. 4.2, is not sufficient enough when considering data sharing aspects. We next discuss the main requirements for secure data sharing in the Cloud and then review literature on securing data sharing in the Cloud.

5.2 Requirements of Data Sharing in the Cloud

To enable data sharing in the Cloud, it is imperative that only authorised users are able to get access to data stored in the Cloud. We summarise the ideal requirements of data sharing in the Cloud below.

- The data owner should be able to specify a group of users that are allowed to view his/her data.
- Any member of the group should gain access to the data anytime without the data owner's intervention.
- No other user, other than the data owner and the members of the group, should gain access to the data, including the Cloud Service Provider.
- The data owner should be able to revoke access to data for any member of the group.
- The data owner should be able to add members to the group.
- No member of the group should be allowed to revoke rights of other members of the group or join new users to the group.
- The data owner should be able to specify who has read/write permissions on the data owner's files.

We now look at the privacy and security requirement of data sharing in the Cloud. Achieving these requirements in the Cloud architecture can go a long way to attracting large numbers of users to adopting and embracing Cloud technology.

- *Data Confidentiality*: Unauthorised users (including the Cloud), should not be able to access data at any given time. Data should remain confidential in transit, at rest and on backup media. Only authorised users should be able to gain access to data.
- *User revocation*: When a user is revoked access rights to data, that user should not be able to gain access to the data at any given time. Ideally, user revocation should not affect other authorised users in the group for efficiency purposes.
- *Scalable and Efficient*: Since the number of Cloud users tends to be extremely large and at times unpredictable as users join and leave, it is imperative that the system maintain efficiency as well as be scalability.
- *Collusion between entities*: When considering data sharing methodologies in the Cloud, it is vital that even when certain entities collude, they should still not be able to access any of the data without the data owner's permission. Earlier works of literature on data sharing did not consider this problem, however collusion between entities can never be written off as an unlikely event.

5.3 Traditional Approach

A trivial solution to data sharing and collaboration in the Cloud involves a data owner distributing encryption keys to every user he authorises. Each user that has authorised access can then get the encrypted data from the Cloud and decrypt the data using the

supplied key. This ensures that no unauthorised user gets access to data even if he manages to download the ciphertext from the Cloud as he does not possess the key for decryption.

This solution however, is not both efficient and effective. Once the data owner decides to revoke a user from accessing their data, one trivial solution would be for the data owner to decrypt the data and re-encrypt the data again, this time with a new key and distribute this new key to the remaining users in the group. This can become extremely costly and places a huge burden on the data owner when considering group sizes in excess of thousands to millions of users. Furthermore, as members of the group continually join and leave, continually re-encrypting data and sending re-encryption keys to a group of this size becomes impractical for the data owner and infeasible to implement in the real world. Currently, there is ongoing research on this problem.

Zhao et al. [61], suggests a progressive elliptic curve encryption scheme (PECE) where a piece of data is encrypted a number of times using multiple keys and later decrypted using one key. Data sharing involves one user, say Alice, encrypting her data using her private key and storing the encrypted data to the Cloud. Another user, say Bob, sends a request for data access permission by sending his public key to Alice. Alice sends a credential to the storage provider for re-encryption of data and sends a credential to Bob to decrypt the data. This is an effective technique as it keeps data confidential as data is encrypted through the entire stages thus never allowing a malicious user to view the plaintext data. This technique also does not allow the permission bearer, in our case Bob, to share the file owned by the permission holder, in our case Alice, with other users. The main problem however with this technique is that it requires the data owner to be online at all times and hence makes it inefficient for everyday users. This technique also assumes the private key of the Cloud provider is shared with the data owner. Realistically, no system administrator would want to share their keys with users and thus making it impractical to be deployed.

5.3.1 The Need for Key Management in the Cloud

Key management is anything you do with a key except encryption and decryption [62] and covers the creation/deletion of keys, activation/deactivation of keys, transportation of keys, storage of keys and so on. Most Cloud service provider's provide basic key encryption schemes for protecting data or may leave it to the user to encrypt their own data.

Either way, there is a need to encrypt data that is involved in the Cloud. But how do we handle the keys that are used for encryption? Where should the keys be stored and who has access to those keys? How do we recover data if keys are lost? Both encryption and key management are very important to help secure applications and data stored in the Cloud [63]. Especially in recent times, there has been a strong need for Cloud providers to adopt a robust key management scheme for their services. However, there are still key management issues affecting Cloud computing as described in [64]. We discuss the 3 requirements of effective key management below.

- *Secure key stores*: The key stores themselves must be protected from malicious users. If a malicious user gains access to the keys, they will then be able to access any encrypted data the key is corresponded to. Hence the key stores themselves must be protected in storage, in transit and on backup media.
- *Access to key stores*: Access to the key stores should be limited to the users that have the rights to access data. Separation of roles should be used to help control access. The entity that uses a given key should not be the entity that stores the key.
- *Key backup and recoverability*: Keys need secure backup and recovery solutions. Loss of keys, although effective for destroying access to data, can be highly devastating to a business and Cloud providers need to ensure that keys aren't lost through backup and recovery mechanisms.

Tim Mather [65] states that key management in enterprises today are broken and that key management in the Cloud is a failed model that is neither effective nor scalable. What cloud computing needs are standards. Fortunately there are a number of standards of key management in the Cloud and is briefly described below.

- *OASIS Key Management Interoperability Protocol (KMIP)*—Used to define a single, comprehensive protocol for communication between encryption systems and enterprise key management systems [66, 67]. KMIP is becoming a widely accepted standard in industry and are looking to implement it within their frameworks.
- *NIST SP 800-57*—Provides general guidelines on key management, the recommended types of encryption schemes and protection requirements as well as information of key recovery [68].
- *IEEE 1619.3 Key Management*—Covers storage encryption and key management mainly for IaaS storage [64]. The standard has been disbanded since December 2010.
- *ISO/IEC 11770-5:2011*—Specifies key establishment mechanisms for multiple entities to provide procedures for handling cryptographic keys used in symmetric and asymmetric encryption algorithms [69].
- Other standards include ISO 11568-2:2012 [70], and IETF KeyProv.

Bruce Schneier [63] quotes "Key management is the hardest part of cryptography and often the Achilles' heel of an otherwise secure system". Pate and Tambay [63] describes that since technology is so broad as it spans various operating systems, storage, encryption and key management, virtualization and VM mobility and Cloud, key management solutions in the Cloud needs to be broader. Luther [62] on the other hand, states that key management is harder than cryptography where cryptography all boils down to math, key management involves technology, people, and processes. He states that strong encryption is nearly impossible to beat compared to key management which is not as robust.

5.3.2 Review of Works on Key Management

Lei et al. [71] illustrated the need for proper key management in the Cloud environment. A Cloud Key Management Infrastructure (CKMI) is proposed which con-

tains a Cloud Key Management Client (CKMC) and Cloud Key Management Server (CKMS). The protocol includes objects which contain keys and certificates, etc, the operations upon them such as creation, deletion, retrieval and updating of keys, certificates, and also attributes related to the object in question such as the object identifier. The method is effective for proper key management however, if the server is broken, all the user's data is lost and there is no proper backup and recovery mechanism, a key requirement of key management as described above.

Huang et al. [13] worked to build on top of the Leakage Resilient Authenticated Key Exchange (LR-AKE) first proposed by Fathi et al. [72] and proposed the LR-AKE Cluster mode protocol for effective key management. The LR-AKE involves the user remembering a password while additionally storing a high-entropy secret on the client machine to allow communication between different servers. In the LR-AKE Cluster mode, the client generates authentication secrets for each server and partial data keys. Each pair authenticates and communicates with each other to combine partial keys to reveal full data keys when user requests. The main weakness with this protocol is that if any one of the servers or the client fails, the data is lost as the keys used to access the data will not be available. The LR-AKE Cluster+ mode builds on the LR-AKE Cluster mode, where aside from the user personal password, the client chooses a random password (256 bits long) and another device (e.g., a USB drive) stores this random password as well as the authentication secrets for added security and higher availability. Secrets are required from both parties of the communication and hence data still remains information-theoretically secure and confidential. One of the drawbacks to this approach is that it requires the maintenance of a number of servers and the client, which adds unwanted complexity when trying to attract large number of users to the Cloud.

Sanka et al. [73] proposed capability lists for effective key management and data access where the data owner does not have to be online at all times. The model involves using a capability list where the data owner creates a list containing an entry for each user and the permissions for file access and stores this list in the CSP. When a user requests access to a file, he requests access to the file directly to the CSP, hence data owner does not have to be online at all times and only needs to be online when registering new users or revoking users from the list. The model is secure and confidential against the Cloud and unauthorised users since they never know the contents of the encrypted data since the key is a shared symmetric key between the data owner and user. The main issue with the model however, is that it assumes the CSP will not alter the capability list. The CSP has access to the unencrypted capability list and can maliciously alter or shut out files from users.

Bennani et al. [74] proposes a model which replicates the database in the cloud n times where n represents the number of roles. When a role is revoked access rights, the corresponding database is removed. Changing a roles access rights leads in the worst case to the creation from scratch a new view and re-keying the corresponding database. One of the main problems with this model is that it is infeasible to implement since it introduces high redundancy and hence is not efficient.

Table 2 Summary of literature on key management in the Cloud

Method	Data/Key redundancy	Data owner online at all times	Confidentiality preserved from CSP	Single point of failure
Lei et al. [71]	N	N	Y	Y
Huang et al. [13]	Y	N	Y	N
Sanka et al. [73]	N	N	N	N
Bennani et al. [74]	Y	N	Y	N

Y yes, *N* no

5.3.3 Discussion

The Table 2 shows a summary of the existing literature based on key management in the Cloud. The works that were reviewed had a strong focus on preventing the need for the data owner to be online at all times. Many of the works that were reviewed also had a strong focus on preventing the Cloud from viewing any of the plaintext at all times. However, in terms of achieving proper key management in the Cloud, some form of redundancy had to be introduced in some of the works.

Proper key management in the Cloud can lead to more secure and confidential sharing of data in the Cloud. A poor key management system can lead to the complete unreliability of the Cloud and can also lose trust from its consumers. Hence it is imperative that more research needs to done in achieving a more robust key management for the Cloud not only to attract more consumers and build trust but also to provide a foundation for secure and private data sharing in the Cloud.

5.4 Recent Approaches

In this section, we provide a review on current works of literature on enabling secure and confidential data sharing in the Cloud.

5.4.1 Attribute-Based Encryption

Attribute-Based Encryption (ABE) is one effective and promising technique that is used to provide fine-grained access control to data in the Cloud. Initially, access to data in the Cloud was provided through Access Control Lists (ACLs) however, this was not scalable and only provided coarse-grained access to data [53]. Attribute-Based encryption first proposed by Goyal et al. [75] provides a more scalable and fine-grained access control to data in comparison to ACLs.

Attribute-Based Encryption is an access control mechanism where a user or a piece of data has attributes associated with it. An access control policy is defined and

if the attributes satisfy the access control policy the user should be able to get access to the piece of data.

There are two kinds of ABE [53], which are described as follows.

- Key-Policy ABE (KP-ABE): The access control policy is stored with the user's private key and the encrypted data additionally stores a number of attributes associated with the data. A user can only decrypt the data if the attributes of the data satisfy the access control policy in the user's key. The access control policy is usually defined as an access tree with interior nodes representing threshold gates and leaf nodes representing attributes.
- Ciphertext-Policy ABE (CP-ABE): Essentially the converse of KP-ABE. The access control policy is stored with the data and the attributes are stored in the user's key.

ABE for Data Sharing and Collaboration

ABE is also used for data sharing and collaboration works. Tu et al. [76] made use of CP-ABE in the context of enterprise applications and also developed a revocation mechanism that simultaneously allows high adaptability, fine-grained access control and revocation. The department assigns users a set of attributes within their secret key and distributes the secret key to the respective users. Any user that satisfies the access control policy defined by the data collaborator can access the data. When a user is revoked access rights, the data is re-encrypted in the Cloud rendering the revoked user's key useless. The scheme is proven to be semantically secure against chosen ciphertext attacks against the CP-ABE model. However, the scheme is not elegant in the case of user revocation since the updating of ciphertexts after user revocation places heavy computation overhead even if the burden is transferred to the Cloud.

Li et al. [77] leverages ABE in the context of the sharing of personal health records (PHR) in the Cloud. Their framework consists of a public domain consisting of users who make accesses on professional records such as doctors, nurses and medical researchers, and also personal domain, which consist of users who are personally associated with the data owner such as family and close friends. Role attributes are assigned to the users in the public domain that represents their professional role and they retrieve their secret keys from an attribute authority. This is effective as the data owner need not be online at all times. In terms of access control, data owners specify role-based fine-grained access control policies for their PHR files. Using role-based access policies greatly reduces key management overhead for owners and users as the owner does not have to manage keys for each individual user.

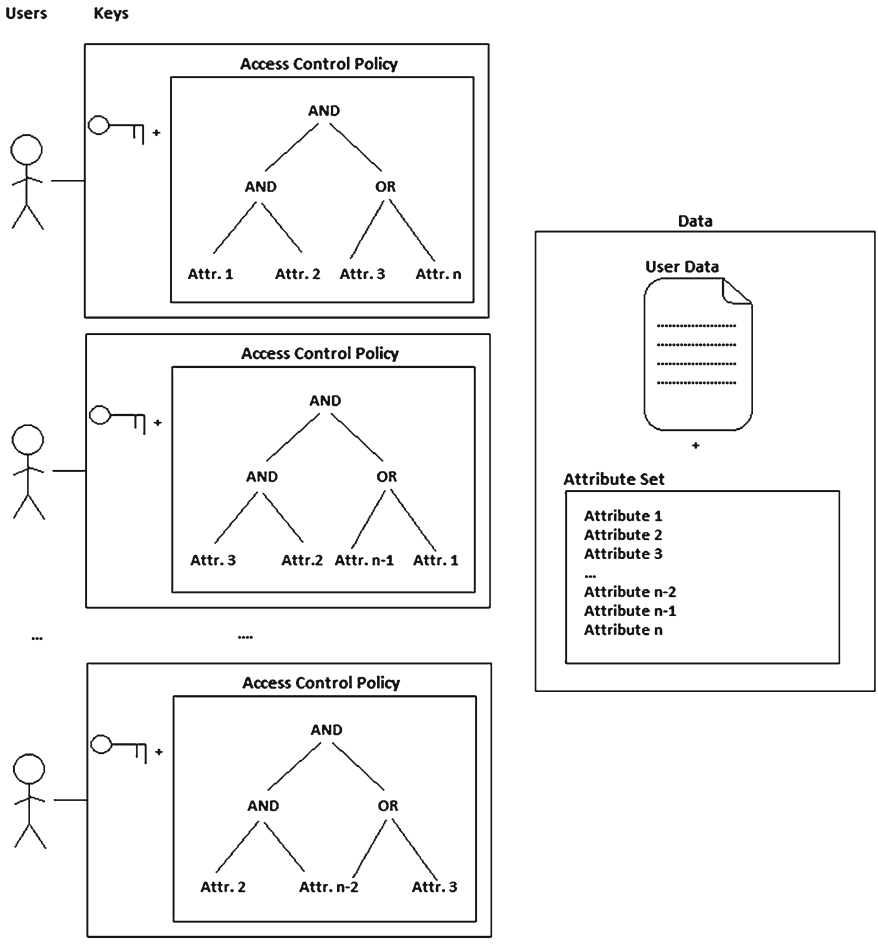


Fig. 2 Key-policy attibute-based encryption

5.4.2 Proxy Re-encryption

Proxy Re-encryption is another technique that is fast becoming adopted for enabling secure and confidential data sharing and collaboration in the Cloud.

Proxy Re-encryption [78] allows a semi-trusted proxy with a re-encryption key to translate a ciphertext under the data owner's public key into another ciphertext that can be decrypted by another user's secret key. At no stage will the proxy be able to access the plaintext. Researchers have utilized proxy re-encryption in relation to the Cloud and in particular for secure and confidential data sharing and collaboration in the Cloud.

We demonstrate a basic Proxy Re-encryption scheme with the diagram below. A user, say Alice, encrypts her data m, using her public key. When she wants to

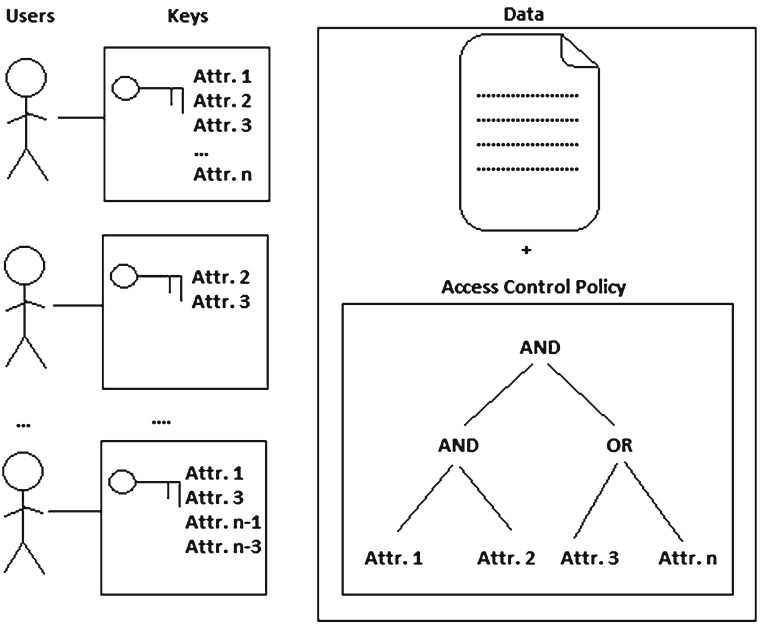


Fig. 3 Ciphertext-policy attribute-based encryption

Fig. 4 A basic proxy re-encryption scheme

share the data with another user, say Bob, she sends the encrypted data to a proxy. The proxy then converts the data encrypted under Alice's public key into data that is encrypted under Bob's public key and sends this to Bob. Bob can now use his private key to decrypt the ciphertext and reveal the contents.

Proxy Re-encryption for Data Sharing and Collaboration

A number of works in literature have propositioned proxy re-encryption for enabling secure and confidential data sharing and collaboration in the Cloud.

Tran et al. [79] uses the idea of Proxy Re-encryption scheme where the data owner's private key is divided into two parts. One half is stored in the data owner's machine while the other is stored in the Cloud proxy. The data owner encrypts the data with half his private key, which then gets encrypted again by the proxy using his other half of the key. Another user who has been granted access rights will then have the same key divided with different parts. One half will be kept on the granted user's machine and the other half stored on the Cloud proxy. The user who has access

rights can then retrieve the data as the proxy will decrypt the ciphertext with half the user's private key in the proxy and then decrypt again on the user's side to retrieve the full plaintext. When the data owner wishes to revoke a user from accessing the data, he simply informs the Cloud proxy to remove the user's key piece. The main strength with this scheme is that it doesn't require re-encryption if a user's rights are revoked and hence saves on computation costs, especially when considering the large number of users in groups. As with the PECE scheme described above [61], this scheme doesn't allow outsiders to view the original plaintext at any point as the data remains in an unreadable format in the Cloud. Only users with granted access rights can view the original plaintext. However, the main problem with this scheme is that of collusion attacks; if a revoked user and the proxy collude, that user then has access to the other entire users private key in the group. Also, the proxy may suffer from too many encryption and decryption operations. The model also assumes that the data owner has already given permission to a number of users to access the data.

5.4.3 Hybrid ABE and PRE

ABE and Proxy Re-encryption have also been used in combination with each other to provide extra security and privacy for data sharing and collaboration in the Cloud. A number of works in literature are taking advantage of combining the power of the two schemes to provide a more robust and guarantee further trust in the data owner for the secure sharing of data in the Cloud.

Yu et al. [80] was one of the first works, which combined ABE, Proxy Re-encryption and lazy encryption schemes for Cloud privacy and security. The scheme works by data owner encrypting his data using a symmetric key and then encrypting the symmetric key using a set of attributes according to KP-ABE scheme. A new user joins the system when the data owner assigns an access structure and its corresponding secret key and distributes this to the new user. To revoke a user, the data owner determines the minimum number of attributes, which will never satisfy the revoked user's access structure and update these as necessary. All the remaining users secret keys will also be updated. Due to the heavy burden of the data owner which may require him to be online at all times to provide key updates, proxy re-encryption is introduced to allow the Cloud to carry out these tasks. Hence most of the computational overhead is delegated to the Cloud. The data owner's data is kept secure and confidential at all times as the Cloud is only exposed to the ciphertext and not the original data contents.

Yang and Zhang [81] also proposed a combination of the ABE scheme and Proxy Re-encryption scheme to enable secure data sharing in the Cloud. The model involves a data owner, say Alice, encrypting data d with a random key k. Alice then determines another random value $k1$ and using access control policy pol, encrypts $k1$ using ABE. Alice then computes $k2$ using operations on k and $k1$, ie, $k2 = k * k1$ and encrypts with her public key using proxy re-encryption. The two keys (ABE key and proxy key) and the encrypted data are then stored in the Cloud. Using an authorisation list, if an authorised user exists, he can then obtain the proxy key which is then re-

encrypted with the user's key. Using this, he decrypts the ABE key, then calculates k, ie, $k1 * k2$ and finally obtains the decrypted file. This technique ensures data is kept confidential against the Cloud and from any unauthorised users. In the scenario that a user is revoked access rights, the data owner simply informs the Cloud to remove that user's entry in the authorisation list and hence is computationally efficient. However, this scheme does not deal with the scenario where a revoked user rejoins the group with different access privileges. The revoked user still has the decryption keys corresponding to ABE and hence in theory can regain access to data he is not allowed to.

Liu et al. [82] proposed a clock-based proxy re-encryption scheme (C-PRE) and combined CP-ABE to achieve fine-grained access control and scalable user revocation. In C-PRE, the data owner and the Cloud share a secret key and this key is used to calculate the PRE keys based on the Cloud's internal clock. The Cloud will re-encrypt the ciphertext with the PRE keys. Each user is associated with a set of attributes and an eligible time which determine how long the user can access the data. The data itself is associated with an access control structure by CP-ABE and also has an access time. When a user requests file access, the Cloud determines the current time using its internal clock and then uses the shared key to calculate PRE keys in time format for all the attributes in the access structure. The PRE keys are then used to re-encrypt the ciphertext. Only users whose attributes satisfy the access control structure and whose eligible time satisfies the access time can decrypt the data. The main benefit with this technique is that the re-encryption of all the data is delegated to the Cloud instead of the data owner and hence is efficient from the data owner's perspective. The user revocation problem is also solved since the data can only be accessed if the user's attribute satisfies the access control structure and their eligible time satisfies the access time. One problem with this technique though, is that data is re-encrypted every time a user makes an access request. Even though the re-encryption is delegated to the Cloud, it is still not a very efficient solution especially when considering very large data sizes.

5.4.4 Discussion

The Table 3 shows a summary of the existing literature based on secure and confidential data sharing in the Cloud. Many of the works reviewed had a strong focus on preventing collusion attacks as well as researching ways for the data owner to be online only when required. In terms of user revocation, some of the reviewed literature showed fast methods of user revocation where revocation involves simply removing a key for instance. Other works required the data to be re-encrypted and the keys to be re-distributed in a secure method and this mainly occurred with works that used ABE techniques.

Data sharing and collaboration in the Cloud is still currently a strong focus of research today and in particular many works are focusing on solving the user revocation problem as well as ways to manage the sharing and collaboration of large data sizes.

Table 3 Summary of literature on secure and confidential data sharing

Method	ABE	PRE	Likelihood of collusion attacks	User revocation	Data owner online all times
Zhao et al. [61]	N	N	N	F	Y
Tu et al. [76]	Y	N	N	S	N
Li et al. [77]	Y	N	N	F	N
Tran et al. [79]	N	Y	Y	F	N
Yu et al. [80]	Y	Y	N	S	N
Yang and Zhang [81]	Y	Y	N	F	N
Liu et al. [82]	Y	Y	N	S	N

Y yes, *N* no, *F* fast, *S* slow

6 Future Directions

In this chapter, we have reviewed literature on ways to provide a secure environment where a data owner can share data with members of his group while preventing any outsiders from gaining any data access in case of malicious activities such as data loss and theft. However, throughout the chapter we assume that members of the group will not carry out malicious activities on the data owner's data.

Auditing and Accountability in the Cloud is a potential for future research in the context of data sharing in the Cloud. As discussed in Sect. 1, many users, in particular organisations and enterprises, benefit from data sharing in the Cloud. However, there is always a likely chance that members of the group can carry out illegal operations on the data such as making illegal copies and distributing copies to friends, general public, etc in order to profit. A future research direction would be to find ways for a data owner to hold accountable any member that carries out malicious activities on their data.

Another research direction would be to give the data owner physical access control over his data. Instead of accountability, the data owner can create a set of access control rules on his data and send the data along with the access control policy. In this way, any member with access to the data can only use the data in such a way that abides by the access control policy. If a member attempts to make illegal copies of the data, the access control policy should "lock" the data to prevent the member from doing so.

Also, since data stored in the Cloud are usually stored and replicated in different geographical locations around the world, it is crucial that the legal jurisdictions are honored and followed. A potential research direction would be to find ways to store and process data in a way that does not breach the privacy and security laws of the region.

7 Summary

Data Sharing and Collobaration in the Cloud is fast becoming available in the near future as demands for data sharing continues to grow rapidly. In this chapter, we presented a review on enabling secure and confidential data sharing and collaboration using Cloud computing technology. We examined definitions related to Cloud computing and privacy. We then looked at privacy and security issues affecting the Cloud followed by what is being done to address these issues.

We then discussed why data sharing in the Cloud is important and the traditional approach to data sharing in the Cloud. We discussed key management in the Cloud and how proper key management leads to more secure and confidential data which can aid secure and private sharing of data in the Cloud. We reviewed current state-of-the-art literature related to key management in the Cloud. We explained the different techniques, namely ABE and PRE that are currently used to enable secure data sharing in the Cloud. We also reviewed current state-of-the-art literature in relation to secure and confidential data sharing in the Cloud and gave a brief overview on the future of data sharing in the Cloud where the data owner could have more control over the usage of their data.

References

1. Mell P, Grance T (2012) The NIST definition of cloud computing. NIST Spec Publ 800:145. National Institute of Standards and Technology, U.S. Department of Commerce. Source: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf. Accessed on Oct 2012
2. Wikipedia definition of Cloud computing (2012). Source: http://en.wikipedia.org/wiki/Cloud_computing. Accessed on Oct 2012
3. Healey M (2010) Why IT needs to push data sharing efforts. Information Week. Source: http://www.informationweek.com/services/integration/why-it-needs-to-push-data-sharing-effort/225700544. Accessed on Oct 2012
4. Gellin A (2012) Facebook's benefits make it worthwhile. Buffalo News.
5. Riley DA (2010) Using google wave and docs for group collaboration. Library Hi Tech News.
6. Wu R (2012) Secure sharing of electronic medical records in cloud computing. Arizona State University, ProQuest Dissertations and Theses
7. Pandey S, Voorsluys W, Niu S, Khandoker A, Buyya R (2012) An autonomic cloud environment for hosting ECG data analysis services. Future Gener Comput Syst 28(1):147–154
8. Bender D (2012) Privacy and security issues in cloud computing. Comput Internet Lawyer 1–15.
9. Judith H, Robin B, Marcia K, Fern H (2009) Cloud computing for dummies. For Dummies.
10. SeongHan S, Kobara K, Imai H (2011) A secure public cloud storage system. International conference on internet technology and secured transactions(ICITST) 2011, pp 103–109.
11. Zhou M, Zhang R, Xie W, Qian W, Zhou A (2010) Security and privacy in cloud computing: a survey. Sixth international conferences on emantics knowledge and grid (SKG) 2010:105–112
12. Rocha F, Abreu S, Correia M (2011) The final Frontier: confidentiality and privacy in the cloud, pp 44–50.
13. Huang R, Gui X, Yu S, Zhuang W (2011) Research on privacy-preserving cloud storage framework supporting ciphertext retrieval. International conference on network computing and information security 2011:93–97

14. Xiao Z, Xiao Y (2012) Security and privacy in cloud computing. IEEE Commun Surveys Tutorials 99:1–17
15. Chen D, Zhao H (2012) Data security and privacy protection issues in cloud computing. International conference on computer science and electronics, engineering, pp 647–651.
16. Zhou M (2010) Security and privacy in the cloud: a survey. Sixth international conference on semantics knowledge and grid (SKG) 2010:105–112
17. Wang J, Liu C, Lin GTR (2011) How to manage information security in cloud, computing, pp 1405–1410.
18. Wang Y (2011) The role of SaaS privacy and security compliance for continued SaaS use. International conference on networked computing and advanced information management (NCM) 2011:303–306
19. Oza N, Karppinen K, Savola R (2010) User experience and security in the cloud-An empirical study in the finnish cloud consortium. IEEE second international conference on cloud computing technology and science (CloudCom) 2010:621–628
20. Sarathy R, Muralidhar K (2006) Secure and useful data sharing. Decis Support Syst 204–220.
21. Butler D Data sharing threatens privacy, vol 449(7163). Nature Publishing, Group, pp 644–645.
22. Mitchley M (2006) Data sharing: progress or not? Credit, Manage 10–11.
23. Feldman L, Patel D, Ortmann L, Robinson K, Popovic T (2012) Educating for the future: another important benefit of data sharing. Lancet 1877–1878.
24. Geoghegan S (2012) The latest on data sharing and secure cloud computing. Law, Order 24–26.
25. Sahafizadeh E, Parsa S (2010) Survey on access control models. 2nd international conference future computer and communication (ICFCC) 2010, pp V1–1-V1-3.
26. Parker RB (1973) A definition of privacy. Rutgers Law Rev 275.
27. Schwab AP, Frank L, Gligorov N (2011) Saying privacy, meaning confidentiality. Am J Bioeth 44–45.
28. HIPAA Privacy (2012) U.S. Department of Health and Human Services. Source: http://www.hhs.gov/ocr/privacy/hipaa/understanding/index.html. Accessed on Nov 2012
29. Internet privacy? (2001) School Libraries in Canada 20–22.
30. Donlon-Cotton C (2010) Privacy and social networking. Law, Order 16–17.
31. Priscilla MR (1986) Privacy, government information, and technology. Public Admin Rev 629–634. Source: http://www.jstor.org/stable/976229. Accessed on Oct 2012
32. Federal Privacy Act. About.com Source: http://usgovinfo.about.com/library/weekly/aa121299a.htm. Accessed on Oct 2012
33. Mcbeth J (2011) Governments need privacy too. The Straits Times.
34. WikiLeaks. Source: http://wikileaks.org. Accessed on Oct 2012
35. Verma R (2012) Confidentiality and privacy issues. The Law Handbook. Education Law. Source: http://www.lawhandbook.org.au/handbook/ch06s03s08.php. Accessed on Oct 2012
36. Ruhr (2011) Cloud computing: Gaps in the 'cloud'. NewsRx Health Sci.
37. Zunnurhain K, Vrbsky SV (2010) Security attacks and solutions in clouds. CloudCom2010 Poster.
38. Motivations of a Criminal Hacker. Microsoft TechNet. Source: http://technet.microsoft.com/en-us/library/cc505924.aspx. Accessed on Oct 2012
39. Hacking Attacks-How and Why. Crucial Paradigm Web Solutions. Source: http://www.crucialp.com/resources/tutorials/website-web-page-site-optimization/hacking-attacks-how-and-why.php
40. Andy P (2007) Salesforce.com Scrambles To Halt Phishing Attacks. http://InternetNews.com. Accessed on Oct 2012
41. Charles A (2011) PlayStation Network: hackers claim to have 2.2m credit cards. The Guardian Technology Blog. Source: http://www.guardian.co.uk/technology/blog/2011/apr/29/playstation-network-hackers-credit-cards. Accessed on Oct 2012
42. Whitney L (2011) Feds investigate alleged attacks on Gmail accounts. CNet news. Source: http://news.cnet.com/8301-1009_3-20068229-83/feds-investigate-alleged-attacks-on-gmail-accounts. Accessed on Oct 2012

43. Jim C, Chyen Yee L (2011) Hacker attacks threaten to dampen cloud computing's prospects. Reuters article. Source: http://www.reuters.com/article/2011/06/03/us-cloudcomputing-idUSTRE7521WQ20110603. Accessed on Oct 2012
44. Dominguez K (2012) Trend micro researchers identify vulnerability in hotmail. Trend Micro. Source: http://blog.trendmicro.com/trendlabs-security-intelligence/trend-micro-researchers-identify-vulnerability-in-hotmail/. Accessed on Oct 2012
45. Choney S (2011) Hotmail, Yahoo Mail users also targets in attacks. NBC News. Source: http://www.nbcnews.com/technology/technolog/hotmail-yahoo-mail-users-also-targets-attacks-123078. Accessed on Oct 2012
46. Galvin N (2012) File-sharing service users in cloud over access to data. The Age.
47. Hulme G (2009) Amazon web services DDoS attack and the cloud. InformationWeek. Source: http://www.informationweek.com/security/amazon-web-services-ddos-attack-and-the/229204417. Accessed on Oct 2012
48. Hachman M (2012) New facebook phishing attack steals accounts, financial information. PC Mag. Source: http://www.pcmag.com/article2/0,2817,2398922,00.asp. Accessed on Oct 2012
49. Albanesius C (2012) Ramnit computer worm compromises 45K facebook logins. PC Mag. Source: http://www.pcmag.com/article2/0,2817,2398432,00.asp. Accessed on Oct 2012
50. NIST Privacy and Security guideslines (2012) NIST. Source: http://csrc.nist.gov/publications/nistpubs/800-144/SP800-144.pdf. Accessed on Oct 2012
51. Cavoukian A (2008) Privacy in the clouds. Identity Inf Soc 1(1):89–108
52. Sabahi F (2011) Cloud computing security threats and responses. IEEE 3rd international conference communication software and networks (ICCSN) 2011, pp 245–249.
53. Li J, Zhao G, Chen X, Xie D, Rong C, Li W, Tang L, Tang Y (2010) Fine-grained data access control systems with user accountability in cloud computing. IEEE second international conference on cloud computing technology and science(CloudCom) 2010, pp 89–96.
54. Naone E (2011) Homomorphic encryption. Technol Rev 50–51.
55. Yao J, Chen S, Nepal S, Levy D, Zic J (2010) TrustStore: making Amazon S3 trustworthy with services composition. 10th IEEE/ACM international conference cluster, cloud and grid computing (CCGrid) 2010, pp 600–605.
56. Scale ME (2009) Cloud computing and collaboration. Library Hi Tech News, pp 10–13.
57. Ratley N (2012) Data-sharing 'would benefit patients'. The Southland Times.
58. Melis RJF, Vehof H, Baars L, Rietveld MC (2011) Sharing of research data. Lancet 378(9808):1995
59. Feldman L, Patel D, Ortmann L, Robinson K, Popovic T (2012) Educating for the future: another important benefit of data sharing. Lancet 379(9829):1877–1878
60. What's in it for me? the benefits of sharing credit data (2011). Banker, Middle East.
61. Zhao G, Rong C, Li J, Zhang F, Tang Y (2010) Trusted data sharing over untrusted cloud storage providers. IEEE second international conference cloud computing technology and science(CloudCom) 2010, pp 97–103.
62. Luther M (2010) Federated key management for secure cloud computing. Voltage security conference presentation. Source: http://storageconference.org/2010/Presentations/KMS/17.Martin.pdf. Accessed on Nov 2012
63. Pate S, Tambay T (2011) Securing the Cloud-Using encryption and key management to solve today's cloud security challenges. Storage Networking Industry Association 2011. Source: http://www.snia.org/sites/default/education/tutorials/2011/spring/security/PateTambay_Securing_the_Cloud_Key_Mgt.pdf. Accessed on Nov 2012
64. Encryption and Key Management (2012) Cloud security alliance wiki. Source: https://wiki.cloudsecurityalliance.org/guidance/index.php/Encryption_and_Key_Management. Accessed on Nov 2012
65. Mather T (2010) Key management in the cloud. O'Reilly Community. Source: http://broadcast.oreilly.com/2010/01/key-management-in-the-cloud.html. Accessed on Nov 2012
66. OASIS Key Management Interoperability Protocol (2012) Web site. Source: https://www.oasisopen.org/committees/tc_home.php?wg_abbrev=kmip#overview. Accessed on Nov 2012

67. Key Management Interoperability Protocol (2012) Wikipedia definition. http://en.wikipedia.org/wiki/Key_Management_Interoperability_Protocol. Accessed on Nov 2012
68. Barker E, Barker W, Burr W, Polk W, Smid M (2007) Recommendation for key management-Part 1: general (Revised). Computer Security. NIST Spec Publ 800–857. Source: http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf. Accessed on Nov 2012
69. ISO/IEC 11770–5:2011 Information technology-Security techniques-Key management-Part 5: group key management. ISO Standards catalogue. Source: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=54527. Accessed on Nov 2012
70. ISO 11568–2:2012 Financial services - Key management (retail) - Part 2: Symmetric ciphers, their key management and life cycle. ISO Standards catalogue. Source: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=53568. Accessed on Nov 2012
71. Lei S, Zishan D, Jindi G (2010) Research on key management infrastructure in cloud computing environment. 9th international conference on grid and cooperative computing (GCC) 2010, pp 404–407.
72. Fathi H, Shin S, Kobara K, Chakraborty S, Imai H, Prasad R (2006) LR-AKE-based AAA for network mobility (NEMO) over wireless links. IEEE J Select Areas Commun 24(9):1725–1737
73. Sanka S, Hota C, Rajarajan M (2010) Secure data access in cloud computing. IEEE 4th international conference internet multimedia services architecture and application(IMSAA) 2010, pp 1–6.
74. Bennani N, Damiani E, Cimato S (2010) Toward cloud-based key management for outsourced databases. IEEE 34th annual computer software and applications conference workshops (COMPSACW) 2010, pp 232–236.
75. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. 13th ACM conference on computer and communications security (CCS '06) 2006, pp 89–98.
76. Tu S, Niu S, Li H, Xiao-ming Y, Li M (2012): Fine-grained access control and revocation for sharing data on clouds. IEEE 26th international parallel and distributed processing symposium workshops and PhD forum (IPDPSW) 2012, pp 2146–2155.
77. Li M, Yu S, Zheng Y, Ren K, Lou W (2013) Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. IEEE Trans Parallel Distrib Syst 131–143.
78. Wang X, Zhong W (2010) A new identity based proxy re-encryption scheme. International conference biomedical engineering and computer science (ICBECS) 2010:145–153
79. Tran DH, Nguyen HL, Zha W, Ng WK (2011) Towards security in sharing data on cloud-based social networks. 8th International conference on information, communications and signal processing (ICICS) 2011, pp 1–5.
80. Yu S, Wang C, Ren K, Lou W (2010) Achieving secure, scalable, and fine-grained data access control in cloud computing. In: INFOCOM, 2010 proceedings IEEE, pp 1–9
81. Yang Y, Zhang Y (2011) A generic scheme for secure ata sharing in cloud. 40th international conference parallel processing workshops (ICPPW) 2011, pp 145–153.
82. Liu Q, Wang G, Wu J (2012) Check-based proxy re-encryption scheme in unreliable clouds. 41st international conference on parallel processing workshops (ICPPW) 2012, pp 304–305.

Adaptive Security Management in SaaS Applications

Mohamed Almorsy, Amani Ibrahim and John Grundy

1 Introduction

Despite the potential benefits, cost savings and revenues that can be gained from adopting the cloud computing model, a downside is that it increases malicious attackers' interest and ability to find vulnerabilities to exploit in cloud software and/or infrastructure. In addition, the cloud model is still not fully mature and a lot of issues that impact the model's credibility and pervasiveness are still open. These include vendor lock-in, multi-tenancy and isolation, data placement and management, service portability, elasticity engines, SLA management, and cloud security, which are well known open research problems in the cloud computing model [1].

Cloud consumers consider security as a major concern that hampers their adoption of the cloud computing model [2]. This is because: (1) enterprises outsource the security management of their cloud-hosted assets to a third party (cloud provider) that hosts their IT assets. This leads to the well-known loss of control problem [3], where cloud consumers do not have security control on their outsourced assets; (2) co-existence of different tenants' assets in the same location and using the same instance of a software service. This makes them unaware of the soundness of service security and what security controls are used to safeguard each tenant's data and ensure no data confidentiality breaches exist; (3) the lack of security guarantees in the Service Level Agreements (SLAs) between cloud consumers and cloud providers. Most existing SLAs focus on reliability, availability and performance of services rather than security [4]; and (4) hosting valuable enterprise assets on publicly accessible infrastructure increases the possibility of attacks and interest of attackers to exploit vulnerabilities in such services to achieve benefits (such as compromising one tenant's data by other tenants who may be competitors).

M. Almorsy (✉) · A. Ibrahim · J. Grundy
Faculty of Information and Communication Technologies, Swinburne University of Technology, John Street, Hawthorn, Victoria 3122, Australia
e-mail: malmorsy@swin.edu.au

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_3,

From the cloud providers' perspective, security requires a lot of expenditure (e.g., security solutions' licenses), resources (security is a resource intensive aspect), and is a difficult problem to master due to its complexity (in terms of number of services, stakeholders, security solutions, etc.). However, ignoring or delaying the improvement of security in the cloud computing roadmap will not meet expected revenues and take-up. Thus, cloud providers have to understand consumers' concerns and seek out new security solutions that resolve such concerns.

Finally, from the security vendors' perspective, developing different security adaptors to integrate with different cloud services is a big headache. The development of a security management framework helps mediate between cloud services and security controls. Such a framework should integrate with security solutions through a common security interface (thus vendors will have to develop only one adaptor), and at the same time integrates with the cloud services to be secured using runtime software instrumentation approaches.

Security management systems help in capturing and defining enterprise asset security, enforcing specified security details, monitoring the security status of these assets, and improving security to meet target security objectives that may also change overtime according to business needs. The security challenges of the cloud computing model make it too hard to depend on manual approaches that require deep involvement of stakeholders, either cloud or service providers or service consumers, to deliver the intended security level. In order to address cloud security challenges, we have identified five main areas we need to consider in order to deliver an adaptive security management framework for the cloud computing model.

- **Cloud computing**: We need to study the cloud computing model characteristics and define the main factors that contribute to the cloud computing security problem. We need to identify the key requirements that should be addressed when developing such a security management model for the cloud computing model. This issue was discussed in previous chapters of this book;
- **Security management**: We need to study the existing security management efforts and standards. We need to identify the key limitations of these efforts when applied to the cloud computing model, and which one(s) to use or extend when addressing the cloud computing model;
- **Security analysis**: We need to determine the main security analysis tasks; what are the existing security analysis efforts in the web applications security analysis area; and how far these efforts support automation of the security analysis task. Moreover, we need to know how these efforts are extensible to support the discovery of existing as well as new vulnerabilities that emerge at runtime;
- **Security engineering**: We need to capture, inject, and update cloud services' security for different tenants at runtime taking into consideration different SaaS multi-tenancy models. We need to study the existing security engineering efforts; identify the key limitations of these efforts that arise from applying these techniques to the cloud services; and how they fit with the multi-tenancy requirements. Moreover, we need to know how much automation is possible with these

approaches to facilitate the automated integration of these approaches with the target cloud services;
- **Security monitoring**: We need to determine what security monitoring platforms exist and how these platforms fit into the cloud multi-tenancy model—i.e., how can we capture different tenants' security metrics and how can we plug-in security probes that collect measurements required to assess the security status specified by different tenants.

The remaining part of this chapter is organized as follows. In Sect. 2, we discuss the key security management standards, differences, and limitations that fit with the cloud computing model. In Sect. 3, we discuss key efforts in security analysis (as a main source of security requirements). In Sect. 4, we discuss key efforts in security enforcement. In Sect. 5, we discuss key efforts and limitations in security monitoring area. In Sect. 6, we introduce our proposed solution for the cloud computing security management problem and the main framework architecture components. In Sect. 7, we introduce a structure.

2 Security Management Standards

Information security management systems [5–7] are defined as systems that deliver a holistic "model for establishing, implementing, operating, monitoring, reviewing, maintaining and improving the protection of information assets". We have identified two key security management standards. The first one is the Federal Information Security Management Act introduced by the National Institute of Standards and Technology–NIST-FISMA [5]. The second one is the International Organization for Standardization and the International Electro-technical Commission–ISO/IEC—ISO27000 [6]. Below we summarize these two standards.

2.1 NIST-FISMA Standard

NIST-FISMA was originally declared as an e-Government Act in 2002 [5]. The FISMA standard delivers the guidelines to develop an agency-wide security management program that help in categorizing information systems, capturing security objectives, enforcing specified security, and monitoring the security status of agency assets. The standard includes a set of guidelines and standards that help implementing the information security management program as follows:

- Standards for categorizing information and information systems by mission impact—i.e., the impact of a security breach on a given information system on the assigned security.
- Standards for minimum security requirements for information and information systems. Based on the security categorization of the information systems, a set

of minimum security requirements and security controls baseline is selected from the set of the available security control baselines—i.e., if an information system is assigned a low security impact, this means that the low impact security requirements and controls baseline is selected and should be enforced by the security experts.
- Guidance for selecting appropriate security controls for information systems. Different information systems may have different natures that may require using specific rather than common security controls.
- Guidance for assessing security controls in information systems and determining security control effectiveness. This guideline defines how to select the security controls to be assessed, the method of the assessment, metrics that could be used, and how the assessment could be conducted.
- Guidance for the security authorization of information systems. This guideline specifies who should be responsible for authorizing the security management plan developed including how the identified risks are addressed /mitigated.
- Guidance for monitoring the security controls and the security authorization of information systems. This guideline defines, for each security controls' family (FISMA standard divides the security controls into a set of 17 security controls families), the set of security metrics that should be measured to monitor the security status of a given system, the frequency of applications, nature of the metric, formula of the metric, unit of measure, etc. (Fig. 1).

2.2 ISO27000 Standard

The ISO27000 standard [6, 8] provides a model to guide the definition and operation of information systems security management. The ISO27000 targets all types of organizations other than federal agencies as intended in the FISMA standard. The ISO27000 standard has a series of security standards that address different areas in the information systems security management framework as follows:

- ISO 27001: This standard gives an overview of the specification of any ISMS that is based on ISO27000 standard. It shows how the ISMS standard is aligned with the Plan-Do-Check-Act (PDCA) management model. It summarizes the key terminologies existing in the security management process and gives a summary of security controls objectives that should be operated.
- ISO 27002: This standard focuses on security controls' implementation guidance to help organizations during the ISMS implementation, reviewing and authorization phases. It shows how these phases could be done to address different security targets including Human Resources, physical security, communication security, access control, etc.
- ISO 27003: This standard gives guidance on implementation of different ISMS phases including planning processes, do processes, check processes, and act processes phases.

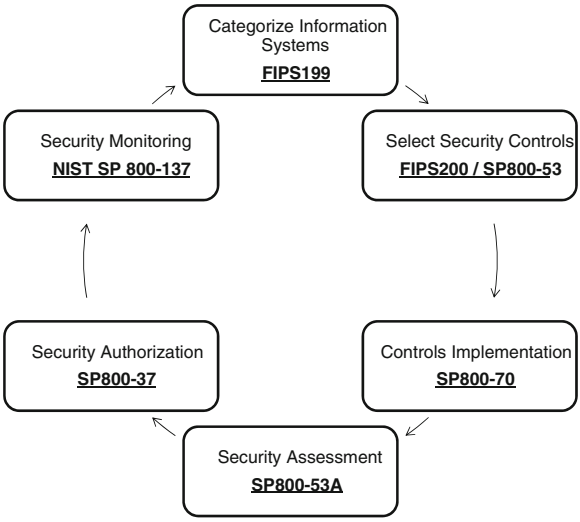


Fig. 1 NIST-FISMA main phases, flow, and standards

- ISO 27004: This standard addresses the ISMS measurements and metrics that could be used, stakeholders and responsibilities, measurement operations, data analytics of the measurement results, and further improvement actions that could be taken.
- ISO 27005: This standard addresses the security risk management process. It details a methodology for information security risk management including risk analysis, treatment, and acceptance.
- ISO 27006: This standard provides guidelines to help organizations in the accreditation process of ISMS certification. It documents the key requirements that should be satisfied and how they can be addressed (Fig. 2).

2.3 Differences Between NIST-FISMA and ISO27000

We have determined that there are a lot of similarities exist between ISO27000 and NIST-FISMA standards. This includes the general approach, phases of security management, complexity of both standards to implement and satisfy, relatively similar concepts, and both provide a list of security controls (NIST specifies links to ISO27000 security controls). However, we found a set of key differences between them as well. NIST-FISMA targets federal agencies while ISO27000 target commercial organizations; however, there is no problem in applying NIST standard to commercial organizations. NIST-FISMA focuses mainly on one or more IT systems. On the other hand, The ISO27000 has organizational-wide focus. NIST uses IT

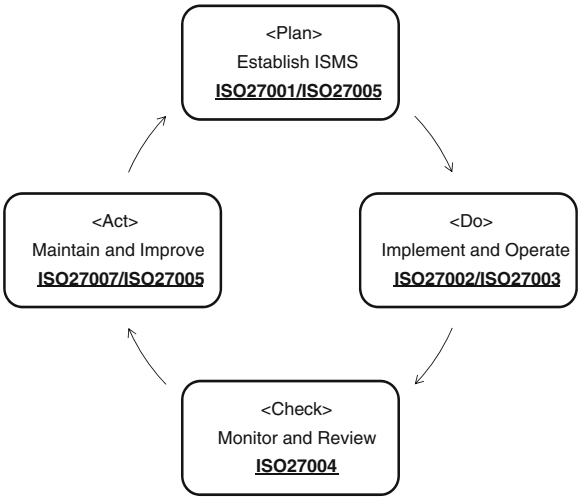


Fig. 2 ISO27000 main phases, flow, and standards

systems' categorization as a selector to decide the set of security controls (baseline) to apply, while ISO27000 assumes that the set of security controls provided in the standard are available to be picked up and used according to the situation. In our opinion, this helps in the security controls selection phase by minimizing the scope of security controls to select from (minimize the possibility of error or missing security). Moreover, this security controls baseline could be customized later according to identified and assessed security risks.

2.4 Security Management Standards and the Cloud Computing Model

Both ISO27000 and NIST-FISMA standards assume that the assets (i.e., IT systems) owner has full control over the security management process of their assets—i.e., these assets are mostly hosted internally inside their network perimeter or at least they can specify and monitor the security of their assets if hosted on a service provider. Thus both standards, in terms of their current specifications, do not fit well with the cloud computing model and the multi-tenancy model where tenants do not have any control on their outsourced assets where service consumers do not have any participation in securing the cloud services . This is a well-known security problem with the cloud computing model known as the "loss-of-control" problem. Multi-tenancy adds a new, complex dimension to the loss-of-control security problem. These security management standards are not designed to take into consideration the service sharing concept introduced by multi-tenancy—i.e., how to capture, enforce,

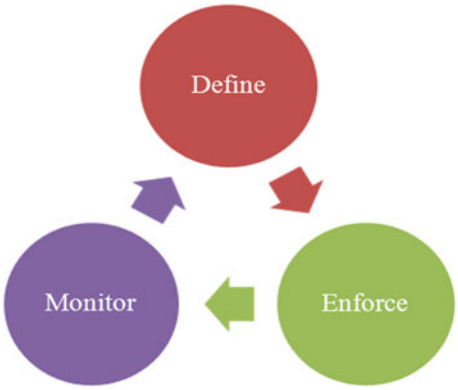


Fig. 3 Information security management main phases

and monitor service security status for different tenants given that these security requirements may change overtime. Moreover, the set of service tenants evolves at runtime. New tenants may register to use the service at runtime. At the same time, other tenants may unregister from the service.

2.5 Rethinking About Security Management Standards

From this analysis of current security management standards, we need to rethink security management systems by considering three key phases: defining security, enforcing security, and monitoring and improving security, as shown in Fig. 3.

- **Defining Security**: This task focuses on how to identify IT assets to be secured and how to categorize them according to their importance, how to capture different stakeholders' security objectives and goals, how to identify security threats, vulnerabilities and attacks, and how to identify the security requirements and controls that mitigate these (identified) security risks and satisfy these (specified) security objectives.
- **Enforcing Security**: This task focuses on how to implement and configure the specified security controls and how to integrate these security controls within the target IT systems that need to be secured, how to manage changes required to effect new security updates resulting from new business needs and new security risks.
- **Monitoring and Improving Security**: This task focuses on how to capture stakeholders' security metrics, how to generate necessary security probes that can collect measurements of security controls and IT system status, how to analyse these results, and how to improve the operated security to improve the security status to match the specified stakeholders' security objectives.

In the next sections, we discuss each of these building blocks with detailed analysis of the existing efforts and their key limitations, taking into consideration key cloud security challenges, especially service outsourcing and multi-tenancy dimensions.

3 Security Analysis

Possible system threats and vulnerabilities represent a key source of security requirements. Security analysis approaches usually depend on (1) *static techniques* that are applied directly to the system source code. This includes pattern matching: searching a "pattern" string inside the source code and the instances where patterns occurred; (2) *Dynamic techniques* introduce faults in order to test the behaviour of the system under test. Some knowledge about the system is required; and (3) *Hybrid techniques* that use both static and dynamic analysis techniques to achieve high accuracy and detect complex vulnerabilities.

Attack Analysis is a security analysis area that targets identifying possible attack paths required to achieve a specific attack goal on a target system entity. Chinchani et al. [9] propose an approach to represent a possible attack vector on a given target component. The approach is based on modelling a system as a set of entities (targets). These targets are connected to each other if there is a physical link. Each target has a set of information it processes (keys). If the attacker does not know this information then there will be a cost to get it. Thus the attack vector reflects the set of entities that an attacker has to go through in order to reach his target. This approach was introduced as a replacement of attack graphs (action-centric) that suffer from the state explosion problem. Sheyner et al. [10] propose an automated approach to generate an attack graph required to achieve a given attack goal. Their approach is based on creating a network model that reflects all atomic attacks existing in each node. They used a modified version of model checking that can generate all possible combinations of possible paths (not only paths that violate or satisfy a given property). They specify the attack goal as CTL expression. The CTL expression is the negation of the attacker objectives. Hewett and Kijsanayothin [11] introduce another approach for formal attack analysis that focuses on hosts rather than on network entities. This is another approach to mitigate the state explosion problem arising from using attack graphs. Ou et al. [12] introduce an approach for attack graph generation based on logic-programming. This approach represents attack scenarios as a set of data-logs and associated rules.

Threat Analysis is the second side in the security analysis triangle that aims to identify possible threats on a given system using the set of identified vulnerabilities or attack graphs and the system architecture details. We may conduct threat analysis on high level system architecture before the system exists using a set of common known weaknesses in the underlying technologies used. Yee et al. [13] introduce an automated approach for threat identification based on a predefined set of expert-defined rules. Given a UML diagram of the target system along with the set of vulnerabilities existing in a given program, it applies the rules to identify the possible system

threats. The approach has a lot of limitations related to the system model and the threat identification rules. Measuring the attack surface of an application is another approach for threat analysis. Manadhata and Wing [14] propose a metric for system security based on system attack surface. This approach is based on calculating entry and exist points and the communication channels used. Abi-Antoun and Barnes [15] propose an approach that looks for security flaws in a given system based on the Microsoft STRIDE model. They introduce a set of rules that can be used to identify and decide the possibility of a given threat based on checking rules related to the preconditions of a discovered vulnerability and the existence of sufficient security controls.

Vulnerability Analysis Existing vulnerability analysis efforts are mostly designed to analyse against specific vulnerability types, such as SQL Injection and Cross-Site Scripting. Jimenez et al. [16] review various software vulnerability prevention and detection techniques. Broadly, static program analysis techniques work on the source code level. This includes pattern matching that searches for a given string inside source code, tokens extracted from source code, or system byte code e.g., calls to specific functions. NIST [17] has been conducting a security analysis tools assessment project (SAMATE). A part of this project is to specify a set of weaknesses that any source code security analysis approach should support including SQL Injection, XSS, OS command Injection, etc. They have also developed a set of test cases that help in assessing the capabilities of a security analysis tool in discovering such vulnerabilities. Halfond et al. [18] introduce a new SQL Injection vulnerability identification technique based on positive tainting. They identify "trusted" strings in an application and only these trusted strings can be used to create certain parts of an SQL query, such as keywords or operators. Dasgupta et al. [19] introduce a framework for analysing database application binaries to automatically identify security, correctness and performance problems especially SQLI vulnerabilities. They adopt data and control flow analysis techniques as well as identifying SQL statements, parameters, tables and conditions and finally analyse such details to identify SQLI vulnerabilities. Martin et al. [20] and Lam et al. [21] introduce a program query language PQL that can be used to capture definition of program queries that are capable to identify security errors or vulnerabilities. A PQL query is a pattern to be matched on execution traces. They focus on Java-based applications and define signatures in terms of code snippets. This approach limits their capability to locate vulnerability instances that match semantically but not syntactically. Wassermann and Su [22] introduce an approach to finding XSS vulnerabilities based on formalizing security policies based on W3C recommendation. They conduct a string-taint analysis using context free grammars to represent sets of possible string values. They then enforce a security policy that the generated web pages include no untrusted scripts. Jovanovic et al. [23] introduce a static analysis tool for detecting web application vulnerabilities. They adopt flow-sensitive, inter-procedural and context-sensitive data flow analysis. They target identifying XSS vulnerabilities only. Ganesh et al. [24] and Kieyzun et al. [25] introduce a string constraint solver to check if a given string can have a substring with a given set of constraints. They use this to conduct white box and dynamic testing to verify if a given system is vulnerable to SQLI attacks. Bau et al. [26] perform an

analysis of black-box web vulnerability scanners. They conducted an evaluation of a set of eight leading commercial tools to assess the supported classes of vulnerabilities and their effectiveness against these target vulnerabilities. A key conclusion of their analysis is that all these tools have low detection rates of advanced and second-order XSS and SQLI. The average percentage of discovered vulnerabilities is only 53 %. Their analysis shows that these tools achieve 87 % in detecting session management vulnerabilities and 45 % in detecting cross site scripting vulnerabilities. Kals et al. [27] introduce a vulnerability scanner that uses a black-box to scan web sites for the presence of exploitable SQLI and XSS. They do not depend on a vulnerability signature database, but they require attacks to be implemented as classes that satisfy certain interfaces. Balzarotti et al. [28] introduce composition of static and dynamic analysis approaches, "Saner", to help validating sanitization functions in web applications. The static analysis is used to identify sensitive sources/sinks methods. Dynamic analysis is used to analyse the identified suspected paths.

4 Security Enforcement

4.1 Security Engineering

Software security engineering aims to develop secure systems that remain dependable in the face of attacks [29]. Security engineering activities include: identifying security objectives that systems should satisfy; identifying security risks that threaten system operation; elicitation of security requirements that should be enforced on the target system to achieve the expected security level; developing security architectures and designs that deliver the security requirements and integrates with the operational environment; and developing, deploying and configuring the developed or purchased security controls. Approaches typically focus on security engineering during system design. Misuse cases [30] capture use cases that should not be allowed and may harm the system operation. UMLSec [31] extends UML with a profile with set of stereotypes to annotate design elements with security requirements. UMLSec provides a comprehensive UML profile but it was developed mainly for use during the design phase. SecureUML [32] provides a meta-model to design RBAC policies of a target system. Both approaches are tightly coupled with the software system design models.

4.2 Adaptive Application Security

Adaptive application security is another key area in security engineering that focuses on enabling a given system to adapt its security capabilities at runtime. Extensible Security Infrastructure [33] is a framework that enables systems to support adaptive

authorization enforcement through updating in memory authorization policy objects with new low level C code policies. It requires developing wrappers for every system resource that catch calls to the resource and check authorization policies. Strata Security API [34] hosts systems on a strata virtual machine. This enables interception of system execution at the instruction level based on user security policies. The framework does not support securing distributed systems and it focuses on low level policies again specified in C code. Serenity [35] enables provisioning of appropriate security and dependability mechanisms for Ambient Intelligence systems at runtime. Security attributes are specified on system components at design time. At runtime the framework links such Serenity-aware systems to the appropriate security and dependability patterns. Serenity does not support dynamic or runtime adaptation for new unanticipated security requirements. Morin et al. [36] propose a security-driven and model-based dynamic adaptation approach enabling applications to reflect the specified context-aware AC policies. Engineers define security policies that take into consideration context information. Whenever the system context changes, the tool updates the system architecture to enforce the suitable security policies.

4.3 Multi-tenancy Security Engineering

Multi-tenant security engineering is a new branch of security engineering. Cai et al. [37] proposed an approach to transform existing web applications into multi-tenant SaaS applications. They focus on the isolation problem by analyzing applications to identify required isolation points that should be handled by the application developers. Guo et al. [38] developed a multi-tenancy enabling framework based on a set of common services that provides security isolation and performance isolation. Their security isolation pattern considers the case of having different security requirements (for authentication and access control only). However, it depends on the tenant's administration to manually configure security policies, map tenant's users and roles to the application's predefined roles. Pervez et al. [39] developed a SaaS architecture that supports multi-tenancy, security and load dissemination. Their architecture is based on a set of services that provide routing, logging, security. Their proposed security service delivers predefined authentication and authorization mechanisms. No control by service consumers of the security mechanisms is supported and no isolation is provided between the authentication and authorization data of different tenants. Menzel et al. [40] proposed a model driven approach and language to specify security requirements on web services and web applications composed of web services. Each application instance (and its services) is deployed on a VM. They assume that web applications are composed of web services only, and that multi-tenant security is maintained through using VMs for each tenant (the simplest case of supporting multi-tenancy).

5 Security Monitoring and Improving

NIST [41] characterizes security metrics into three types: (i) *Implementation metrics*. These metrics are intended to demonstrate progress in implementing information security solutions and related policies and procedures; (ii) *Effectiveness/efficiency metrics*. These metrics are intended to monitor if the implemented security controls are implemented correctly, operating as intended and meeting the desired outcomes. Effectiveness focuses on the robustness of the security controls while efficiency focuses on the capability of the security controls to mitigate the security objectives; (iii) *Impact measures* are used to articulate the impact of IT security on missions including cost savings, and public trust. Existing efforts in information security measurements and monitoring focus on proposing guidelines or processes to be followed when defining metrics and collecting measurements. Chandra and Khan [42] introduce steps to identify the required security metrics in a given system. This includes specify metric requirements, identify vulnerabilities, identify software characteristics, analyse security model, categorize security metrics, specify security metric measures, design metric development process, develop a security metric, and finalize the metric suite. Similar efforts have been introduced for the cloud [43].

6 A Collaboration-Based Cloud Security Management Framework

6.1 Aligning NIST-FISMA with the Cloud Computing Model

To build our adaptive model-based cloud computing security management approach, we found it crucial to base such an approach on well-known and well-defined security management standards, such as ISO27000 or NIST-FISMA. However, such security management standards are far from covering the full complexity of the cloud computing model and mainly multi-tenancy and outsourcing of IT assets. In this Section, we introduce our proposed alignment of the NIST-FISMA standard to fit with the cloud computing model, enabling cloud providers and consumers to maintain their security management processes on cloud platforms and services. This framework, as summarized in Table 1, is based on improving collaboration between cloud providers, service providers and service consumers in managing the security of the cloud platform and the hosted services. Below we explain how we aligned each phase of the NIST-FISMA standard with the cloud computing model.

Service Security Categorization

Each service (S_J) hosted on the cloud platform can be used by different tenants. Each service tenant (Ti), or cloud consumer (CC) owns their information only in the shared

Table 1 Alignment of NIST-FISMA standard with the cloud computing model

Phase	Task	CP	SP	CC	Input	Output
Security categorization	Categorize security impact (SC)	Informed	Informed	Responsible	Business objectives	Security impact level
Security controls selection	Register security controls	Responsible	Responsible	Responsible	Control datasheet	Security controls registry
	Generate security controls baseline	Responsible (Automated by the framework)			Service SC + Controls registry	Controls baseline + matching status
	Assess service risks	Responsible (planned to be automated)			Service + platform arch. + CVE + CWE	Service Vulns baseline + Threats + Risks
	Tailor security baseline	Responsible (planned to be automated)			Baseline + Risk assessment	Security mgmt plan (SLA)
Controls implementation	Implement security controls	Responsible (planned to be automated)			Security mgmt plan	Updated security plan
Security assessment	Define security metrics	Responsible	Informed	Responsible	Security objective	Security assessment plan
	Assess security status	Responsible (Automated by the framework)			Security assessment plan	assessment report
Service authorization	Authorize service	Informed	Informed	Responsible	Security plan + assessment report	Service authorization document
Security monitoring	Monitor security status	Responsible (Automated by the framework)			Security assessment plan	Security status report

service (S_J). The tenant is the only entity that can decide/change the impact of a loss of confidentiality, integrity and availability on their business objectives. Each tenant may assign different impact levels (Low, Medium, or High) to security breaches of their information. NIST has introduced a new project that proposes a new model for security management of the cloud computing model–FedRAMP [44]. In FedRAMP, the cloud provider specifies the security categorization of services delivered on their cloud platform. However, this is not sufficient as the cloud provider does not have sufficient knowledge about the impact of information security breaches on their tenants' business objectives. Our approach enables cloud consumers to be involved in specifying the security categorization of their information. Moreover, our approach enables both scenarios where we can consider the security categorization (SC) per tenant or per service. The security categorization of the service is calculated as the maximum of all tenants' categorizations:

$$\begin{aligned} SC(T_i) = \{&(\text{confidentiality, impact}),\\ &(\text{integrity, impact}),\\ &(\text{availability, impact})\} \in, \text{Impact}\{\text{Low, Medium, High}\} \end{aligned} \tag{1}$$

$$\begin{aligned} SC(S_j) = \{&(\text{Confidentiality, Max}(\forall T_i(\text{impact}))),\\ &(\text{Integrity, Max}(\forall T_i(\text{impact}))),\\ &(\text{Availability, Max}(\forall T_i(\text{impact})))\} \end{aligned} \tag{2}$$

Security Control Selection

The selection of the security controls to be implemented in protecting tenants' assets has two steps: (a) *baseline security controls selection*—the FISMA standard provides a catalogue of security control templates categorized into three baselines (low, medium and high). Based on the security categorization of the tenant or the service we select the initial baseline of controls that are expected to provide the required level of security specified by tenants. (b) *tailoring of the security controls baseline*—we tailor the security controls baseline identified to cover the service possible vulnerabilities, threats, risks and the other environmental factors as follows:

1. **The service risk assessment process**
   - *Vulnerabilities Identification*—this step requires being aware of the service and the operational environment architecture. We consider the involvement of the service provider (SP) who knows the internal structure of the provided service and the cloud provider (CP) who knows the cloud platform architecture.
   - *Threat Identification*—the possible threats, threat sources and capabilities on a given service can be identified by collaboration among the SPs, CPs, and

CCs. CCs are involved as they have the knowledge about their assets' value and know who may be a source of security breaches.
- *Risk Likelihood*—based on the capabilities of the threat sources and the nature of the existing vulnerabilities, the risk likelihood is rated as low, medium or high.
- *Risk Level (Risk Exposure)*—based on the risk impact (as defined in phase 1) and risk likelihood we drive the risk level as (Risk Level = Impact X Likelihood).

2. **The security controls baseline tailoring process**
Based on the risk assessment process, the selected security controls baseline can be tailored to mitigate new risks and fit with the new environment conditions (*scoping of the security controls*) as follows:

- Identify the common security controls; the cloud stakeholders decide on which security controls in the baseline they plan to replace with a common security control (either provided by the CPs or by the CCs);
- Identify critical and non-critical system components; the SPs and CCs should define which components are critical to enforce security on it and which are non-critical (may be because they are already in a trusted zone) so no possible security breaches;
- Identify technology and environment related security controls; used whenever required, such as wireless network security controls;
- *Compensating Security Controls*—whenever the stakeholders find that one or more of the security controls in the tailored baseline do not fit with their environment conditions or are not available, they may decide to replace such controls with a compensating control;
- *Set Security controls parameters*—the last step in the baseline tailoring process is the security controls' parameters configuration, such as minimum password length, maximum number of unsuccessful logins, etc. This is done by collaboration between the CPs and CCs. The outcome of this phase is a security management plan that documents service security categorization, risks, and the tailored security controls baseline.

Security Controls Implementation

The security plan for each tenant describes the security controls to be implemented by each involved stakeholder based on the security control category (common, service specific). The common security controls implementation is the responsibility of the common control provider who may be the CPs (in case of internal security controls) or the CC (in case of external controls). The service-specific security controls implementation is the responsibility of the SPs. Each stakeholder must document their security controls implementation configuration in the security management plan.

Security Controls Assessment

Security controls assessment is required to make sure that the security controls implemented are functioning properly and meet the security objectives specified. This step includes developing a security assessment plan that defines: what are the security controls to be assessed; what are the assessment methods to be used; and what are the security metrics for each security control. The results of the assessment process are documented in a security assessment report. This step may result in going back to the previous steps in case of deficiency in the controls implemented or continuing with the next steps.

Service Authorization

This step represents the formal acceptance of the stakeholders on the identified risks involved in the adoption of the service and the agreed on mitigations. The security plan and security assessment plan are the security SLA among the involved parties.

Monitoring the Effectiveness of Security Controls

The CPs should provide security monitoring tools to help the CCs in monitoring the security status of their assets. The monitoring tools should have the capability to capture the required security metrics and report the collected measures in a security status report either event-based or periodic-based. The results of the monitoring process may require re-entering the SMP to handle new unanticipated changes.

6.2 Security Automation

After aligning the FISMA standard with the cloud model we adopted a set of security standards to help improving the framework automation and its integration with the existing security capabilities, as shown in Fig. 4 and examples listed in Table 2.

- ***Common Platform Enumeration (CPE)*** [45]: The CPE provides a structured naming schema for IT systems including hardware, operating systems and applications. We use the CPE as the naming convention of the cloud platform components and services. This helps in sharing the same service name with other cloud platforms and with the existing vulnerabilities databases such as NVD [46].
- ***Common Weakness Enumeration (CWE)*** *and* ***Common Attack Pattern Enumeration and Classification (CAPEC)*** [45]: The CWE Provides a catalogue of the community recognized software weaknesses. The CAPEC provides a catalogue of the common attack patterns. Each attack pattern provides a description of the attack scenario, likelihood, knowledge required and possible mitigations.

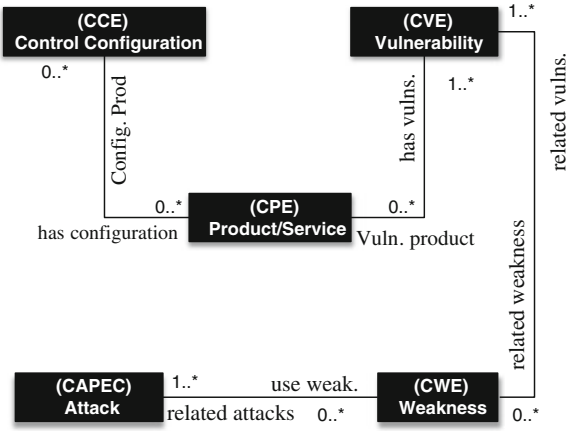


Fig. 4 A class diagram of the adopted security standards and their relationships

We use the CWE and CAPEC as a reference for the cloud stakeholders during the vulnerabilities identification phase.

- ***Common Vulnerability and Exposure (CVE)*** [45]: The CVE provides a dictionary of the common vulnerabilities with a reference to the set of the vulnerable products (encoded in the CPE). It also offers vulnerability scoring that reflects the severity of the vulnerability. We use the CVE to retrieve the know vulnerabilities discovered in the service or the platform under investigation.
- ***Common Configuration Enumeration (CCE)*** [45]: The CCE provides a structured and unique naming to systems' configuration statements so that systems can communicate and understand such configurations. We use the CCE in the security controls implementation phase. Instead of configuring security controls manually, the administrators can assign values to security control templates' parameters. Our framework uses these configurations in managing the selected security controls.

Table 2 Formats of the adopted security standards

Standard	Format	Example
CPE	cpe:/part: vendor : product: version : update : edition: language	cpe:/a:SWINSOFT: Galactic:1.0: update1:pro:en-us
CVE	CVE-Year-SerialNumber	CVE-2010-0249
CWE	CWE-SerialNumber	CWE-441
CAPEC	CAPEC-SerialNumber	CAPEC-113
CCE	CCE-softwareID-SerialNumber	CCE-17743-6

6.3 Cloud Security Management Framework Architecture

Our framework architecture consists of three main layers: a management layer, an enforcement layer, and a feedback layer. These layers, shown in Fig. 5, represent the realization of the ISMS phases.

- ***Management layer***: This layer is responsible for capturing the security specifications of the CPs, SPs, and CCs. It consists of: (a) The security categorization service used by the hosted services' tenants to specify security categorization of their information maintained by the cloud services; (b) The collaborative risk assessment service where all the cloud platform stakeholders participate in the risk assessment process with the knowledge they possess. (c) The security controls manager service is used to register security controls, their mappings to the FISMA security controls' templates, and their log files structure and locations. (d) The security metrics manager service is used by the cloud stakeholders to register security metrics they need to measure about the platform security. (e) The multi-tenant security plan (SLA) viewer service is used to reflect the tenant security agreement. This shows the tenant-service security categorization, vulnerabilities, threats, risks, the selected mitigation controls and the required metrics. (f) The multi-tenant security status viewer. This reflects the current values of the security metrics and their trends.
- ***Enforcement layer***: This layer is responsible for security planning and security controls selection based on the identified risks. The selected security controls are documented in the security management plan. The implementation service

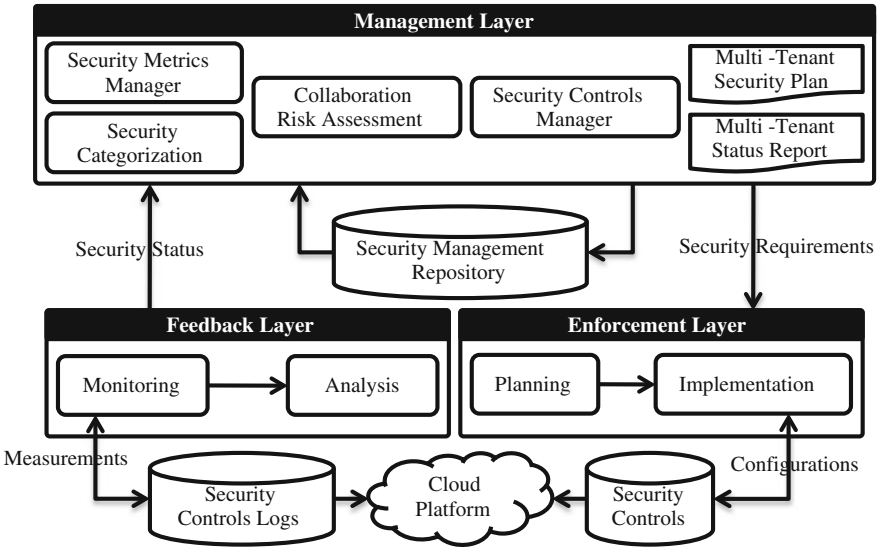


Fig. 5 Our collaboration-based security managmenet framework architecture

then uses this plan to maintain security control configuration parameters and the mapping of such parameters to the corresponding security controls.

- ***Feedback layer***: This layer has two key services. The monitoring service is responsible for collecting measures defined in the security metrics manager and storing it in the security management repository to be used by the analysis service and by the multi-tenant security status reporting service. The analysis service evaluates the collected measures to make sure that the system is operating within the defined boundaries for each metric. If there is a deviation from the predefined limits, the analysis service will give alerts to update the current configurations.

7 Usage Example

To demonstrate the capabilities of our cloud computing security framework and our prototype tool implementing this framework we introduce a motivating example, shown in Fig. 6, that happens in any SaaS delivery platform.

Consider SwinSoft, a software house developing business applications. Lately, SwinSoft decided to develop a multi-tenant SaaS application "Galactic-ERP". During the development of Galactic, SwinSoft used some of the external services developed and deployed on GreenCloud (a cloud platform that will host Galactic as well) and BlueCloud (a cloud platform hosting some business services). In the meanwhile, SwinSoft has got two tenants (Swinburne and Auckland) who are interested to use Galactic service. Both tenants have their own business as well as their own security requirements. Both of them are worried about the loss-of-control problem arising from the adoption of the cloud model. They would like to maintain their own security requirements on their cloud hosted assets.

The first step in our approach is to register the Galactic ERP service in the cloud platform service repository so that it can be used by the CCs. This step can be done either by SWINSOFT or by GC. In this step we use the CPE name as the service ID,

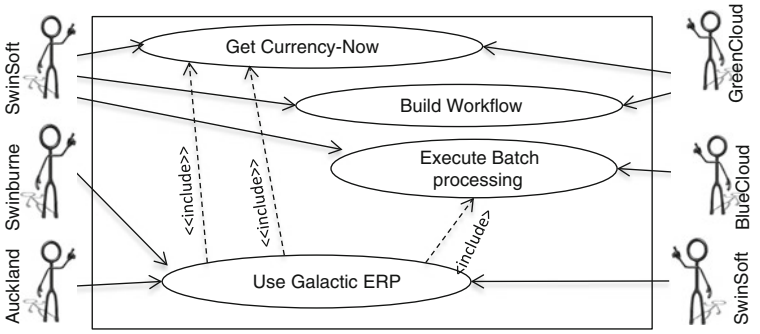


Fig. 6 Motivating example

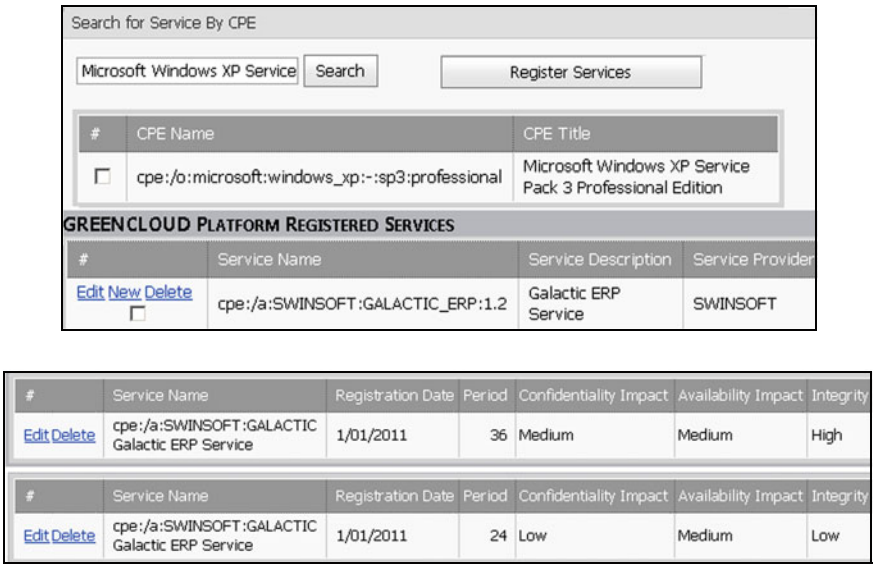


Fig. 7 Registering a service (*top*) and tenants (*bottom*)

Fig. 7 (top). A new tenant, Auckland, can register their interest in using the Galactic service. Then Auckland will be granted a permission to manage the security of his information maintained by Galactic service. The same is done by Swinburne, Fig. 7 (bottom). Now Auckland and Swinburne can use our framework to maintain their SMP on their assets as follows:

1. ***Service Security Categorization***: The Swinburne security administrator specifies the impact level of losing the confidentiality, integrity, and availability of their data maintained by the Galactic ERP service. The same will be done by the Auckland security administrator, as shown in Fig. 7 (bottom). Whenever a new tenant registers their interest in a service and defines their security categorization of data processed by the service (or any of the existing tenants update his

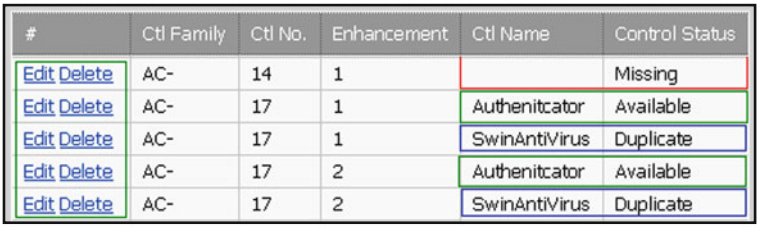


Fig. 8 Security controls baseline with controls' status

security categorization), the framework will update the overall service security categorization.

2. ***Security Controls Selection***: GC as a cloud provider already publishes their security controls database. Swinburne and Auckland can register their own security controls using the security controls manager service. Based on the security categorization step, the framework generates the security controls' templates baseline. This baseline identifies the security controls' templates that are: **satisfied** (matches one of the registered security controls), **missing** (does not match registered security controls), and **duplicate** (more than one matched control), shown in Fig. 8.

   (a) ***The Service Risk Assessment Process***: Galactic vulnerabilities are identified for the first time by SWINSOFT with the help of GC who know the architecture of the service and the hosting cloud platform. Both SWINSOFT and GC have the responsibility to maintain the service vulnerabilities list up to date. The framework enables to synchronize the service vulnerabilities with the community vulnerabilities database–NVD. Each CC—Swinburne and Auckland—should review the defined threats and risks on Galactic and append any missing threats. The framework integrates with the CWE and CAPEC databases to help stakeholders in identifying possible vulnerabilities whenever the service does not have vulnerabilities recorded in the NVD.

   (b) ***The controls baseline tailoring process***: The CCs decide which security controls in the baseline they plan to replace with common security controls provided by the CP or the CC, as shown in Fig. 8. Then SWINSOFT, Auckland, and Swinburne select the critical service components that must be secured. Swinburne and Auckland define their security controls' parameter configurations. The security controls provided by the cloud platform can only be reviewed.

   The final outcome of this step is a security management plan that documents the service security categorization, vulnerabilities, threats, risks, and the tailored security controls to mitigate the identified possible security breaches, as shown in Fig. 9.

3. ***Security Controls Implementation***: Each stakeholder implements the security controls under their responsibility as stated in the security plan and the security controls configurations as specified before.
4. ***Assessing the implemented security controls***: The controls to be assessed and the objectives of the assessment are defined by GC, Auckland and Swinburne, and are documented in the tenant security assessment plan. The execution of such a plan, the assessment process, should be conducted by a third party. Our framework helps in assessing security controls status when using security controls that integrate with our framework (the framework can understand and read their log structure). The outcome of the assessment phase is a security assessment report.
5. ***Service Authorization***: Swinburne and Auckland give their formal acceptance of the security plan, assessment plan, and the assessment reports. This acceptance represents the authorization decision to use Galactic by the CC.

The Security Management plan for the service: Galactic ERP Service

#	Registration Date	Registration (Mths)	Security Categorization
	1/01/2011	24	Low

Vulnerability Name	Vulnerability Description
CVE-2005-0413	Multiple SQL injection vulnerabilities in MyPHP Forum 1.0 allow remote attackers to execute arbitra
CVE-2005-2471	pstopnm in netpbm does not properly use the "-dSAFER" option when calling Ghostscript to convert a
CVE-2005-4195	Multiple SQL injection vulnerabilities in Scout Portal Toolkit (SPT) 1.3.1 and earlier allow remote

Threat Name	Threat Description	Threat Source
DenialSrv	Denial of service	Attacker
InfoCopy	Copy of information at storage	Internal
InfoMod	Modification of information while being transfered	Attacker
MemMod	Modification of data being processed	Maleware

Risk Name	Risk Probability	Confidentiality Impact	Availability Impact	Integrity Impact	Risk Level
DOS	0.7	Low	High	Low	Medium

Control Name	Control Description	Control Baseline	Control Type	Control Family
Authenitcator	an authenitcation security control	Low	Specific	Access Control
SwinAntiVirus	an antivirus security solution	Low	Common	System and Information Integrity
SwinIPS	an intrusion prevention system	Low	CommonControl	System and Information Integrity

Measurement Name	Measurement Description	Frequency	Measrement Steps	Security Control
LoginActivity	Identify the user login rates	48	count(logstatus)	Authenitcator

Fig. 9 Auckland security management plan

6. ***Monitoring the effectiveness of the security controls***: The framework collects the defined security metrics as per the assessment plan of each tenant and generates status reports to the intended cloud stakeholders. A report shows the metrics status and trends, as shown in Fig. 10.

The procedure we went through in the example above should be applied not only for published services but also on the cloud platform services themselves. In this case the CP uses our framework to manage the platform security from a consumer perspective. We have done this for the Galactic exemplar used above.

8 Discussion

Our approach provides a security management process; a set of standards-based models for describing platforms, platform services, and services; the security needs of different stakeholders; known threats, risks and mitigations for a cloud deployment; and a tool supporting security plan development and partial automation of a derived security plan. Our approach is comprehensive, supporting all stakeholder perspectives, and collaborative, allowing different stakeholders to develop a

Measurement Name	Measurement Date	Measurement Value
LoginActivity	12/01/2011	8
LoginActivity	13/01/2011	10
LoginActivity	14/01/2011	14
UnsuccessfulLogins	10/01/2011	2
UnsuccessfulLogins	11/01/2011	4
UnsuccessfulLogins	12/01/2011	6
UnsuccessfulLogins	13/01/2011	8

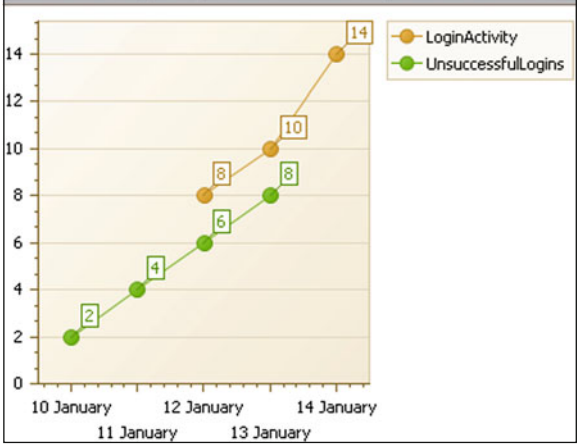


Fig. 10 Sample of Swinburne security status report

mutually-satisfying security model. It addresses the multi-tenancy nature of shared cloud-hosted services when tenants have different security requirements and different SMPs. This is achieved by maintaining and managing multiple security profiles with multiple security controls on the same service. Such controls are delivered by different security vendors. This enables managing traceability between controls and identified risks, and identifies which risks are still not fully mitigated.

The security management process (SMP) of a cloud service has two possible scenarios: either to let each tenant go through the whole SMP as if he is the only user of the service (tenant-based SMP), or to accumulate all tenants' security requirements on a given service and maintain the SMP at the service level (service-based SMP). The later scenario is more straight forward because cloud stakeholders collaborate together to secure the cloud platform and their services with one set of security requirements. The former scenario gives the CCs more control in securing their cloud hosted asset but it has the following problems: (i) the current multi-tenancy feature delivered by cloud services enables tenants to customize service functionality but it does not enable tenants to customize service security capabilities; (ii) the underlying cloud platform infrastructure, such as the VM OS, does not often support multi-tenancy. This means that we cannot install multiple anti-viruses or anti-malware

systems on the same OS and be able to configure each one to monitor specific memory processes for a certain user. One solution may be to use a VM for each tenant [40]. However, this work around may not be applicable if the service is not designed for individual instances usage or if the cloud platform does not support VM technology.

Whenever the CCs are not interested in following the security standards or require a light-weight version of our approach, they can leave out as many steps as they want including security controls implementation and customization, security assessment and service authorization steps. The mandatory steps are service categorization and controls selection. Another variation of our framework is to enable CPs to deliver predefined security versions for the service such as service X with (low, medium, high) security profile. CCs can select the suitable version based on their security needs.

9 Adaptive Cloud Computing Security Management

The new cloud security management approach we have introduced in the previous sections addresses the loss-of-control and lack-of-trust problems by getting cloud stockholders involved in securing their outsource cloud assets. Our framework is based on the NIST-FISMA standard after aligning it to fit with the multi-tenant cloud model. Moreover, it adopts a set of security standards to automate the security management process. However, our framework lacks two key points: (i) automated integration of security solutions with the target services at runtime without a need for service customization or special preparations at service design time; (ii) automated security analysis of cloud services using an online security analysis service that can analyse services against such vulnerabilities as well as new vulnerabilities at runtime.

Figure 11 shows a more refined version of our framework using models as an abstraction approach. Each stakeholder summarizes their information in models according to their roles. Cloud providers model their platform details, service providers model their service details, and cloud consumers model their security model. These models are weaved in secure-system model (integrated model reflecting critical system entities and security details to be applied on these entities). This model is used to generate a security management plan that guides the configuration of security controls, integration of security controls within the target critical entities either in the service or in the cloud model. In our approach we move from top to bottom in refinement process starting from models to real configurations "Enforcement". On the other side, we collect measure from the services and security controls and consolidate such measures into metrics reflecting security status "Feedback".

Figure 12 shows the high-level architecture of our adaptive-security management framework that we have been working on over the last three years. This security framework should be hosted on a cloud platform and used to manage cloud services security. Our approach architecture is inspired from the MAPE-K autonomic computing [47]:

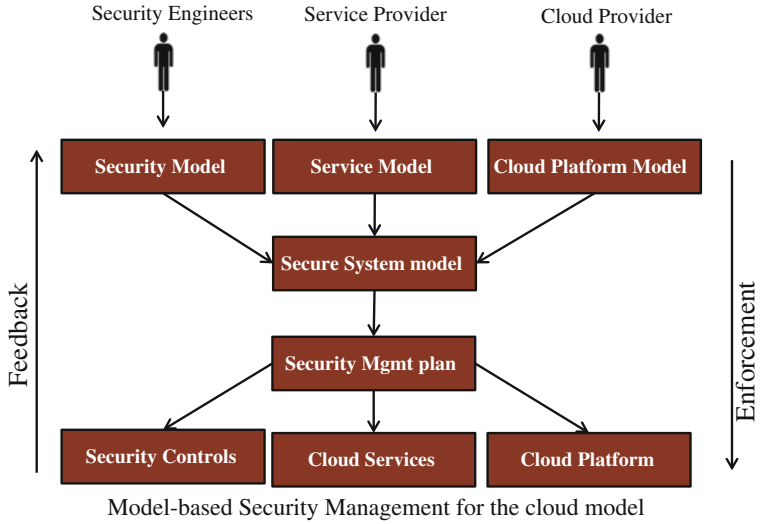


Fig. 11 General approach

- **Management Component**
  This is a model-based security management component that is responsible for capturing services and security details where service provider system engineers model their services architecture, features and behavior and tenants' security engineers model and verify their own security objectives, requirements, architecture, and metrics. Both models are then woven together in a tenant secure-system model that guides the next steps of security enforcement and monitoring.

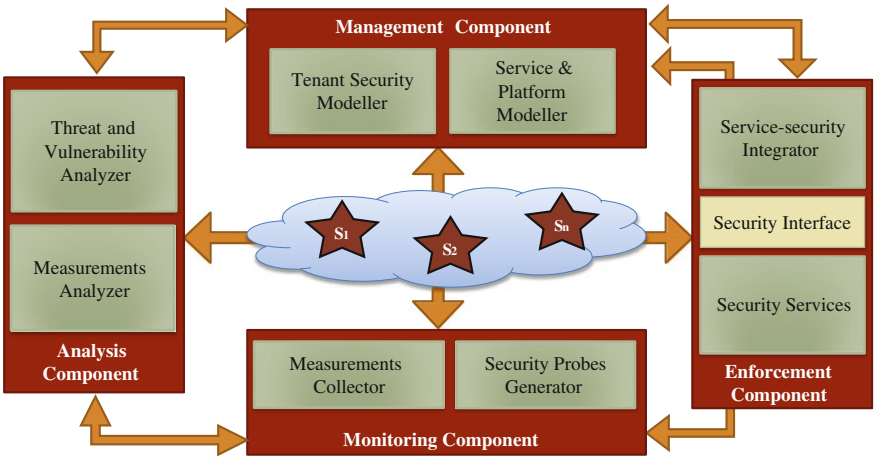


Fig. 12 The high-level architecture of our adaptive-security management framework

- **Enforcement Component**
  This component is responsible for integrating the specified security details specified by different tenants with the target cloud services. To support flexible security controls integration with the target services, we developed a common security interface that defines a set of functionalities to be realized by the security vendors through a common security controls adaptor. This enables security controls to easily integrate with our enforcement component which integrates with the cloud services.
- **Monitoring Component**
  This component is responsible for generating the required security probes for tenants' specified metrics (captured in the management layer). These probes are then deployed in the cloud services to start capturing measures. Moreover, this component is responsible for collecting the measures from these probes (according to metrics specified frequencies) and passing such measures to the analysis component.
- **Analysis Component**
  The analysis component is responsible for two main tasks: performing security analysis of the cloud services including vulnerability and threat analysis. The analysis component analyses the deployed services and their architectures to identify flaws and security bugs. Such issues are delegated to the security management component in order to incorporate in the security status reports for tenants as well as dynamically updating the security controls deployed to block the reported security issues. The analysis component also analyse the measurements reported by the monitoring component against a set of predefined metrics stable ranges—e.g., number of incorrect user authentications per day should be less than three trials, so the analysis component should analyse the reported measures of incorrect authentications. This may also include taking corrective actions to defend against such probable attack.

10 Future Research Directions

The area of the cloud computing security is relatively new. Many security problems need to be addressed to promote a trust of the cloud computing model. Here, we summarize three of the key research problems:

- **Data Confidentiality**: Service consumers are worried about their assets (data) security. Thus, they tend to keep their data encrypted while in transmission, storage, and processing; however, applications need to work on plaintext data. Thus, these applications will need to decrypt customers' data on the cloud platform. This task is prone to attacks from malicious insiders (cloud platform administrators) who have access to the physical servers and may deploy any malicious software to access plain data while being processed in memory. It is highly required to find some approaches that conform that tenants' data confidentiality cannot be breached by malicious insiders.

- **Tenant-oriented Security Engineering**: Multi-tenant applications are shared among different service tenants who may be competitors or malicious tenants. Thus, each tenant is interested in defining their own security requirements and enforce their own security controls. Moreover, the set of service tenants emerges at runtime; new customers register to use the service and existing customers may unregister from using the service. Thus, there is a high need for security engineering approaches that help in developing cloud services that can capture, enforce, and integrate different tenants' security requirements and controls at runtime. This also requires developing some security standards that both service providers and security vendors have to follow in order to facilitate the integration between services and required security solutions.
- **Security SLA**: The area of service level agreement becomes one of the hot topics with the wide-adoption of the service outsourcing either as SOA or as cloud computing. However, most of the efforts in the SLA management focus on how to negotiate and define SLA terms including availability, reliability, and performance but not security. Moreover, they focus on how to monitor and avoid violation of the SLA terms. Thus, there is a big need to security SLA management approaches that can define security terms to agree on, monitor the realization and satisfaction of these terms and take proactive and corrective actions whenever needed.

11 Conclusion

In this chapter, we introduced a new cloud computing security management model based on joint-collaboration between different cloud platform stakeholders according to who owns the piece of information required to go through the full security management process. This in turn reflects our proposed alignment of the NIST-FISMA standard as one of the main security management standards. We also introduced a usage example of the proposed approach where we have different tenants sharing the same service instance while each stakeholder would like to enforce his security requirements on his cloud hosted assets. We discussed our comprehensive, adaptive security management platform that helps in capturing tenants' security requirements, realizing these requirements and integrating security controls with target cloud services at runtime, and monitoring the security status of these cloud services according to the tenants' security objectives captured in terms of security metrics.

References

1. European Network and Information Security Agency (ENISA) (2009) Cloud computing: benefits, risks and recommendations for information security. http://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-risk-assessment. Accessed on July 2010
2. International Data Corporation (2010) IDC ranking of issues of cloud computing model. http://blogs.idc.com/ie/?p=210. Accessed on July 2010

3. Kandukuri BR, Paturi R, Rakshit A (2009) Cloud security issues. In: Proceedings of the (2009) IEEE international conference on services computing, pp 517–520
4. Chaves SAD, Westphall CB, Lamin FR (2010) SLA perspective in security management for cloud computing. In: Sixth international conference on networking and services, Cancun, Mexico, pp 212–217
5. National Institute of standards and technology (NIST) The federal information security management act (FISMA), U.S. Government Printing 2002, Washington. http://csrc.nist.gov/drivers/documents/FISMA-final.pdf. Accessed on Aug 2010
6. International Organization for Standardization (ISO) (2009) ISO/IEC 27000—Information technology, security techniques, information security management systems, overview and vocabulary. ISO/IEC 27001:2005(E). http://webstore.iec.ch/preview/info_isoiec27000%7Bed1.0%7Den.pdf. Accessed on July 2010
7. Humphreys E (2008) Information security management standards: compliance, governance and risk management. Inf Sec Tech Rep 13:247–255
8. Tsohou A, Kokolakis S, Lambrinoudakis C, Gritzalis S (2010) Information systems security management: a review and a classification of the ISO standards. In: Sideridis A, Patrikakis C (eds) Next generation society. Technological and legal issues. Springer, Berlin, pp 220–235
9. Chinchani R, Iyer A, Ngo H, Upadhyaya S (2004) A target-centric formal model for insider threat and more. Technical Report 2004–16, University of Buffalo, US2004
10. Sheyner O, Haines J, Jha S, Lippmann R, Wing JM (2002) Automated generation and analysis of attack graphs. In: Security and privacy, 2002. Proceedings. 2002 IEEE Symposium on, pp 273–284
11. Hewett R, Kijsanayothin P (2008) Host-centric model checking for network vulnerability analysis. In: Computer security applications conference, (2008) ACSAC 2008. Annual, pp 225–234
12. Ou X, Govindavajhala S, Appel AW (2005) MulVAL: a logic-based network security analyzer. Presented at the 14th USENIX security symposium, MD, USA, August, Baltimore
13. Yee G, Xie X, Majumdar S (2010) Automated threat identification for UML. In: Proceedings of the international conference on security and cryptography (SECRYPT), pp 1–7
14. Manadhata PK, Wing JM (2011) An attack surface metric. IEEE Trans Softw Eng 37:371–386
15. Abi-Antoun M, Barnes JM (2010) Analyzing security architectures. Presented at the proceedings of the IEEE/ACM international conference on automated software engineering, Antwerp, Belgium
16. Jimenez W, Mammar A, Cavalli A (2009) Software vulnarabilities, prevention and detection methods: a review. In: Proceedings of European workshop on security in model driven architecture, Enschede, The Netherlands, pp 6–13
17. NIST (2007) Source code security analysis tool functional specification version 1.1. Accessed on 2011
18. Halfond WGJ, Orso A, Manolios P (2006) Using positive tainting and syntax-aware evaluation to counter SQL injection attacks. In: Proceedings of 14th ACM SIGSOFT international symposium on Foundations of software engineering, Oregon, USA, pp 175–185
19. Dasgupta A, Narasayya V, Syamala M, A static analysis framework for database applications. In: Proceedings of (2009) IEEE international conference on data. Engineering, pp 1403–1414
20. Martin M, Livshits B, Lam MS (2005) Finding application errors and security flaws using PQL: a program query language. In: Proceedings of the 20th annual ACM SIGPLAN conference on object-oriented programming, systems, languages, and applications CA, USA, pp 365–383
21. Lam MS, Martin M, Livshits B, Whaley J (2008) Securing web applications with static and dynamic information flow tracking. In: Proceedings of (2008) ACM SIGPLAN symposium on partial evaluation and semantics-based program manipulation, California, USA, pp 3–12
22. Wassermann G, Su Z (2008) Static detection of cross-site scripting vulnerabilities. In: Proceedings of 30th international conference on Software engineering, Leipzig, Germany, pp 171–180
23. Jovanovic N, Kruegel C, Kirda E (2006) Pixy: a static analysis tool for detecting web application vulnerabilities. In: Proceedings of 2006 IEEE symposium on security and privacy, pp 258–263

24. Ganesh V, Kieżun A, Artzi S, Guo PJ, Hooimeijer P, Ernst M (2011) HAMPI: a string solver for testing, analysis and vulnerability detection. In: Proceedings of 23rd international conference on Computer aided verification, Snowbird, UT, pp 1–19
25. Kieyzun A, Guo PJ, Jayaraman K, Ernst MD (2009) Automatic creation of SQL injection and cross-site scripting attacks. In: Proceedings of 31st international conference on, software engineering, pp 199–209
26. Bau J, Bursztein E, Gupta D, Mitchell J (2010) State of the art: automated black-box web application vulnerability testing. In: Proceedings of 2010 IEEE symposium on security and privacy, pp 332–345
27. Kals S, Kirda E, Kruegel C, Jovanovic N (2006) SecuBat: a web vulnerability scanner. Presented at the proceedings of 15th international conference on World Wide Web, Edinburgh, Scotland
28. Balzarotti D, Cova M, Felmetsger V, Jovanovic N, Kirda E, Kruegel C, Vigna G (2008) Saner: composing static and dynamic analysis to validate sanitization in web applications. In: Proceedings of 2008 IEEE symposium on security and privacy, pp 387–401
29. Anderson R (2001) Security engineering: a guide to building dependable distributed systems. Wiley, New York
30. Sindre G, Opdahl A (2005) Eliciting security requirements with misuse cases. Requirements Eng 10:34–44
31. Jürjens J (2001) Towards development of secure systems using UMLsec. In: Fundamental approaches to software engineering, vol 2029. Springer, Berlin, pp 187–200
32. Lodderstedt T, Basin D, Doser J (2002) SecureUML: a UML-based modeling language for model-driven security. In: The 5th international conference on the Unified Modeling Language, Dresden, Germany, pp 426–441
33. Hashii B, Malabarba S, Pandey R et al (2000) Supporting reconfigurable security policies for mobile programs. Presented at the proceedings of the 9th international World Wide Web conference on computer networks, Amsterdam, The Netherlands
34. Scott K, Kumar N, Velusamy S et al (2003) Retargetable and reconfigurable software dynamic translation. Presented at the proceedings of the international symposium on Code generation and optimization, San Francisco, California
35. Sanchez-Cid F, Mana A (2008) SERENITY pattern-based software development life-cycle. In: 19th international workshop on database and expert systems application, pp 305–309
36. Morin B, Mouelhi T, Fleurey F, Le Traon Y, Barais O, Jézéquel J (2010) Security-driven model-based dynamic adaptation. Presented at the the 25nd IEEE/ACM international conference on automated software engineering, Antwerp, Belgium
37. Cai H, Wang N, Zhou MJ (2010) A transparent approach of enabling SaaS multi-tenancy in the cloud. In: 2010 6th World Congress on Services (SERVICES-1), pp 40–47
38. Guo CJ, Sun W, Huang Y, Wang ZH, Gao B (2007) A framework for native multi-tenancy application development and management. In: E-Commerce technology and the 4th IEEE international conference on enterprise computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007. The 9th IEEE international conference on, pp 551–558
39. Pervez Z, Lee S, Lee Y-K (2010) Multi-tenant, secure, load disseminated SaaS architecture. In: 12th international conference on advanced communication technology, Gangwon-Do, South Korea, pp 214–219
40. Menzel M, Warschofsky R, Thomas I, Willems C, Meinel C (2010) The service security lab: a model-driven platform to compose and explore service security in the cloud. In: 2010 6th World Congress on Services (SERVICES-1), pp 115–122
41. Chew E, Swanson M, Stine K, Bartol N et al (2008) Performance measuremenet guide for information security. National Institute of Standards and Technology
42. Chandra S, Khan RA (2009) Software security metric identification framework (SSM). Presented at the proceedings of the international conference on advances in computing. Communication and Control, Mumbai, India
43. Bayuk J (2011) Cloud security metrics. In: 2011 6th international conference on system of systems engineering (SoSE), pp 341–345

44. NIST Concept of Operations (CONOPS)—FedRAMP NIST2012
45. Mitre Corporation (2010) Making security measurable. Available at http://measurablesecurity.mitre.org/
46. National Institute of Standards and Technology–NIST (2010) National vulnerabilities database home. Available at http://nvd.nist.gov/
47. Salehie M, Tahvildari L (2009) Self-adaptive software: landscape and research challenges. ACM Trans Auton Adapt Syst 4:1–42

Smart Resource Allocation to Improve Cloud Security

Eddy Caron, Frédéric Desprez and Jonathan Rouzaud-Cornabas

1 Introduction

Virtualization is now widely used in modern datacenters. Thanks to mature software stacks and the widespread availability of plaforms all over the world, the Cloud is now available for many applications of different kinds. Security and performance are the main goal users want to achieve when porting applications over IaaS or PaaS platforms. Security has been proven to be sometimes difficult to obtain [3, 60, 85] and several issues have been raised in public Clouds and public domain virtualization software stacks. Several different kinds of attacks and security issues can be observed that may lower the impact of Clouds. On the performance side, the expectations are higher than what can be actually obtained on today's public Clouds. Shared nodes lead to performance degradation that are not appropriate for high performance applications. Isolation is then a critical issue both for security and performance concerns.

The purpose of this chapter is to present the limitation of using virtualization technology as the sole approach to isolate workloads and users within a Cloud. In highly secured environments, strong isolation is done by unshared resources environment for two tasks with different security clearance. It is still the strongest defense against covert channels (and other attacks). But this approach eliminates most of the current public and private Clouds but also the way how virtualization is used. With the

E. Caron (✉)
ENS, LIP - UMR CNRS - ENS Lyon - INRIA - UCBL, 5668 Lyon, France
e-mail: eddy.caron@ens-lyon.fr

F. Desprez
INRIA, LIP - UMR CNRS - ENS Lyon - INRIA - UCBL, 5668 Lyon, France
e-mail: frederic.desprez@inria.fr

J. Rouzaud-Cornabas
CNRS, IN2P3 Computing Center, CNRS, IN2P3, Lyon-Villeurbanne, France
e-mail: jonathan.rouzaud-cornabas@ens-lyon.fr

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_4,

widespread usage of virtualization, the need of strong isolation in such environment becomes critical.

First, in Sect. 2, we present the micro-architecture of modern computer. We also present the virtualization limitations and how they can be exploited to attack security and privacy in the Clouds. We show that the same issue exists for performance and network isolation. Accordingly, we need a mechanism that can provide such strong isolation. Then, in Sect. 3, we introduce a method to model Cloud platforms and applications. Moreover, we introduce a way to let the user express her/his security requirements to cope with the virtualization limitation we highlighted. We present how a Cloud Service Provider (CSP) can modified its resource allocation software to take into account these requirements. Moreover, as these requirements induce fragmentation within the Cloud platform, we introduce algorithms that reduce it. In Sect. 5, we introduce a Big Data use case and show how the resource allocation can improve both security USDA and privacy within Clouds. Moreover, we highlight the trade-off between isolation security requirements and platform consolidation. Finally, we go beyond isolation requirements and present our roadmap toward a new software in Cloud Middleware Platform (CMP) to do smart resources allocation and self-configuration of security appliances based on security requirements.

2 Micro-Architecture: Where Virtualization Failed

Since the last few years, the complexity of physical machines' hardware topology has increased dramatically [15]. The number of cores, shared caches, and memory nodes have completely changed the micro-architecture of computers. From a simple CPU architecture during the Pentium era, we have now access to complex multi-core, multi-level caches that can be specific to a core, shared between some or all. Symmetric Multithreaded Processors (SMP)[1] brings another level of hierarchy. SMP is a mean to share the resources of a core between multiple logical processors.

With the increasing number of cores, scalability becomes a major issue. To address it, modern processors use non-uniform interconnects.[2] This technology is called Non-Uniform Memory Access (NUMA) architectures. But memory is not the only resources to be accessed through these non-uniform interconnects, Input/Ouput Devices accesses are done in a similar manner (Non-Uniform Input/Output Access—NUIOA). In this type of architecture, some cores have faster access than others to some I/O devices and memory banks [24]. For example, Fig. 1 shows the inner architectural components of five modern platforms. As one can see, depending on whether the data is stored in a directly connected memory bank or in a remote one, the access to it will need to passthrough one (or more) CPU. The same is true for the I/O devices. Accordingly, the placement of tasks on CPU. and their related data on memory banks is critical to exploit performance on these modern architecture.

[1] HyperThreading in Intel processor.

[2] HyperTransport for AMD and QPI for Intel.

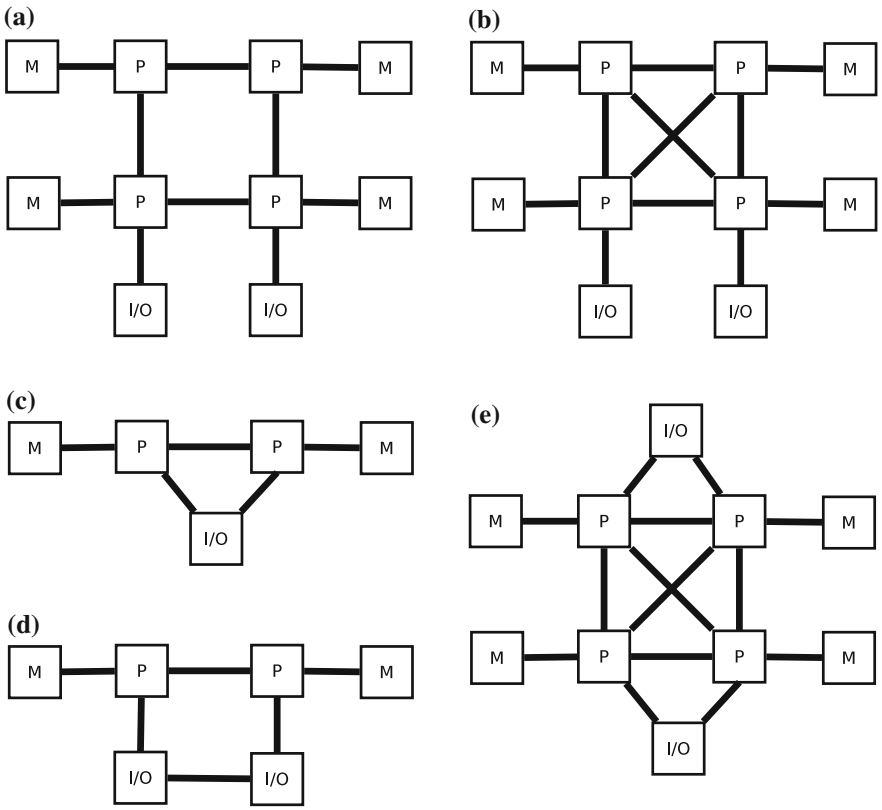


Fig. 1 Interconnection of processors (*P*), memory (*M*) and I/O chipsets in some modern architectures [24]: **a** four AMD Istanbul processors; **b** four AMD Magny-Cours processors; **c** and **d** two Intel Westmere-EP processors; **e** four Intel Nehalem-EX processors

For instance, Fig. 2 presents a simple but yet already complex topology of a computer based on one Intel Core i7-2620M. Each core has two Processing Unit (PU) thanks to SMP. The two PUs of a core share two caches (L1 and L2) and all the cores and PUs share a global cache (L3). The size of the caches is increasing with the level from 32 KB for the first one to 4,096 KB for the third. A more complex topology is presented in the Fig. 3, it is the hierarchical topology of a computer based on two AMD Opteron™ 6164 HE. As one can see, there are three level of cache. Two are dedicated to a core (L1 and L2) and one is shared (L3) between all the cores of a NUMA Node. All the caches have different size from 64 KB to L1 to 5,118 KB for L3. Moreover, each socket groups two NUMA Nodes. Each NUMA Node and sockets have a direct access to a part of memory banks.

The inner-topology of computer has a large impact on performance depending where the different process of an application are placed. For example, the DMA throughput can decrease by up to 42 % when accessing a GPU from a distant NUMA

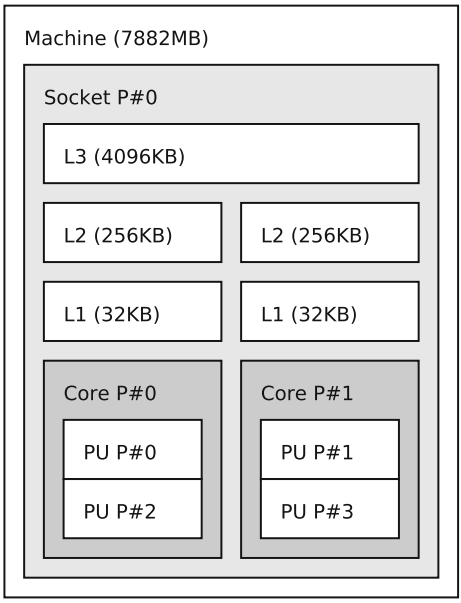


Fig. 2 Micro-architecture of computer with Intel Core i7-2620M

node. Thus, exporting the topology to the VM is critical. Virtualized NUMA (vNUMA) is already available [3]. But there is not a one-to-one mapping between virtual and physical NUMA. Indeed, some physical NUMA can be shared through two vNUMA by VMs and two NUMA can be aggregated into one vNUMA. Accordingly, performance optimization can be buried by the vNUMA not really providing the same kind of hardware connections it exposes.

2.1 Covert Channel

Nowadays, virtualization is used to guarantee a strong isolation between tenants sharing an infrastructure. This means that virtualization is used to prevent interference between VMs but also to avoid data exfiltrating across VM boundaries.

As presented in the previous section, a VM can share, with other VMs, a NUMA and a set of memory banks, I/O devices and caches without being able to know with whom. But contrary to the memory and CPU, the micro-architectural component are not properly virtualized in modern platform. Therefore, the sharing of micro-architectural components can be used as covert channels to leak information between concurrent VMs. In non-virtualized multi-process contexts, it has been demonstrated that isolation does not protect against a wide variety of side-channel attacks that can be used to extract sensitive data such as cryptographic keys [85]. "Access-driven" attacks are the most efficient way to build such attacks. These attacks exploit

Fig. 3 Micro-architecture of computer with dual-AMD Opteron™ 6164 HE

micro-architectural components such as caches. In [83], authors present new work on the creation of covert channels of communication between virtual machines through the L2 cache processors. They focus on the L2 caches even if other covert channels exist. Their goal is to quantify the rate of leakage through this channel. In previous work [60], three channels were studied: memory (0.006 bps), disk (0.0005 bps), and L2 (0.2 bps). The L2 cache is slow but effective enough to steal cryptographic keys for example. The purpose of [83] is to see if they can optimize the speed. In experimental testbeds, they were able to reach a rate of 223.71 bps. On Amazon EC2, they were able to reach to get a flow between 1.27 and 10.46 bps. Other studies of covert channels within Clouds have been made in [52, 84].

Following the studies presented in [3, 60] applies the same approach to build a scenario where a malicious VM is able to extract the private ElGamal decryption key from a co-resident VM running GnuPG. To do it, the attacker monitors the usage of shared micro-architectural components to extract information about the key. Moreover, contrary to other attacks on micro-architectural components, the one presented in [85] does not rely on SMP. Indeed, in modern public Clouds such as Amazon EC2 and Microsoft Azure, SMP is disabled as cache-based covert channel attacks are easy to build [42]. For example, L1 caches that are dedicated to one core are the easiest way to create covert channels [57] between VMs.

In [81], authors present a new inter-VMs attack in the Cloud. Unlike previous works, they do not use the L2 cache CPU but the memory bus. This new channel allows them not to be limited to a set of cores sharing a cache but they can reach the entire physical machine. To do this, they observe the memory bus contention as covert channel. They are able to create a covert channel with a bandwidth of 746.8 bps (+/− 10.1) on their architecture. In Amazon EC2, most of the time they are able to reach a bandwidth of 343.5 bps (+/− 66.1) and 107.9 bps (+/− 39.9) in the worst case. This improvement is at least a factor of 10 with approaches via the L2 cache.

2.2 Noisy Neighbors: Performance Interference

Virtualization is also used in Cloud Computing to provide performance isolation between multiple tenants. With a perfect performance isolation, no performance interference between two VMs must exist. Thus, noisy VM will have no impact on the performance of other VMs. Hypervisors implement resources allocation algorithms that fairly shares resources between different VMs. But these algorithms focus only on CPU time and memory capacity and do not take into account other resources [7]. Furthermore, current hardware virtualization does not provide mechanisms to limit micro-architectural components usage.

In the previous section, we have shown that the lack of strong isolation with virtualization and in particular on micro-architectural components can lead to covert channel. But with the lack of strong isolation and in particular on micro-architectural components, the performance of a VM can suffer from interference coming from another one running on the same hardware. In [76], authors show that the performance

loss can be up to 80 % on a cache-sensitive benchmark because of the behavior of a collocated VM. But others works have shown the same type of performance loss can be achieved by attacking shared memory [8] and [48]. Away from performance interference due to the lack of strong isolation on single computer, several works have studied the strong performance interference in Cloud Computing platforms [58, 62, 78]. As one can expect, all the shared resources (compute, storage and network) in Clouds are affected by interference issues.

The effectiveness of such interference could lead to a new class of attacks called Resource-Freeing Attacks (RFA) in [76]. *"A resource-freeing attack leverages these observations of contention to improve a* VM's *performance by forcing a competing* VM *to saturate some bottleneck resources"*. The purpose of such attack would be to create interference that leads a VM to starve due to its inability to access specific micro-architectural components and thus freeing other resources. Accordingly, the VM that launches the attack could used these resources to boost its performance. Authors of [76] have shown they can gain a performance boost of up to 60 % in lab and 13 % on Amazon EC2.

2.3 Detection of Covert Channel

Amazon provides a service that dedicates physical machines to an user: dedicated instances. But this service comes with a high price. Cloud Service Providers could also monitor the cache miss and memory access to detect cross-VM attacks. Reference [84] continuously monitors memory access latencies and generates alarms when anomalies are detected. But memory probing incurs high performance overhead. Moreover, their approach has a high false positive rate.

VALID [12] is a specific language for security assurance. The goal is to verify the respect of security properties *i.e.* to detect violations of them. The language can express security goals such as isolation and generally express security properties based on information flow. Their purpose is to verify the configuration of a virtual infrastructure to find the bad configuration that could lead to the rupture of a safety property. For example, they want to be able to detect if a firewall misconfiguration of a virtual machine would allow someone to access a service which it should not. In [12], they introduce three security properties: correctness operational (check that the service is available and has been correctly deployed), failure resilience (very close to the concept of redundancy), and isolation (from a network flow point of view). For isolation, they offer to check the isolation of a service or a group of services. Their definition is consistent with the concept of security zone of Amazon EC2, which brings together a group a set of virtual machine sharing the same rules of firewall. To allow the exchange of information, they have added to their model guardian which verifies whether an information flow between two services is allowed. In principle, a guardian is very close to the concept of the firewall. Accordingly, their approach is limited to detect isolation faults in network and is not able to do it for micro-architectures.

These approaches are limited to detect such attack on known channels. Furthermore, it is not possible to block the attack neither than protecting against attacks on unknown covert channels that can arise due to new micro-architectural design.

2.4 Improving Isolation Within a Physical Machine

As we have presented in Sect. 2.1, it is possible to create covert channels between VMs by using the lack of proper virtualization of micro-architectural components.

In [59], authors propose to improve the isolation of virtual machines while minimizing the loss of consolidation. In the worst case, protecting against covert channels between VMs leads to place a single virtual machine per physical machine. Accordingly, consolidation of the infrastructure is enormously damaged and the platform is largely underutilized. The approach presented in [59] consist in offering better refinement within this isolation. They propose two algorithms: cache hierarchy aware core assignment and page coloring based cache partitioning. Indeed, one risk factor in terms of covert channels between virtual machines is the Last Level Cache (LLC) in multi-core processors and the memory. This attack can lead to leakage of data but also affect the availability of virtual machines. Their placement algorithm takes into account the problem of cache and allows to assign virtual machines on cores that do not share the same cache. It limits the risk of breaking the isolation between virtual machines without having to use multiple physical machines. A similar method is proposed for the memory using page coloring. But their approach introduces a very large overhead that renders it inapplicable on real world Cloud platforms.

In [72], authors propose a protection that allows the hypervisor to counter attacks on CPU caches. Their approach applies to the LLC but not to the cache closest to the processor indeed they are too small (L2 or L3 cache but not L1). In addition, their approach does not take into account the case where 2 VMs share a core i.e. (SMP) , because in this case it is impossible to guarantee anything. Their approach has an overhead equivalent to 3 loss of memory and cache. In addition, it brings an overhead of 5.9–7.2 % in terms of computation. Accordingly, it causes less overhead than [59]. Other works around software mechanisms to increase performance isolation exist for other resources: disk [28], memory bandwidth [77], and network [63].

For the cryptographic key issue, numerous publications exist on covert channel resistant algorithms [54, 56]. But these algorithms are slower than the classic ones. For the noisy neighbors problem, several works [10, 19] have studied how scheduling algorithms can detect two applications with conflicting resources usage patterns and schedule them so they do not share resources. But they do not prevent such attacks.

The NoHype concept [35, 71] consists in removing the virtualization layer while retaining the key features enabled by virtualization. Indeed, a malicious tenant can attack [37, 49, 80] the virtualization layer i.e. the hypervisor and inspect the memory thus leaking confidential information e.g. cryptographic keys or modify the software running inside another VM. NoHype has some limitations: one VM per core, memory partitioning and I/O devices virtualization are done by the hardware. Accordingly, it

is not possible with their concept to share a core or to do over-provisioning. They limit covert channel by enabling one VM per core but as we have show, other covert channels exist within the micro-architecture components. Moreover, they are nots able to do fair sharing on I/O and memory bus. Thus the performance isolation between VMs with NoHype remains weak.

2.5 Network Isolation

On Cloud, network access is critical. It's true that a VM is useless if it is not able to communicate with others services or to be accessed by clients running inside or outside the Cloud. But as the physical machines, network resources are shared between the tenants of a Cloud. As for the physical machines, a strong isolation is critical to provide performance and security isolation. Similarly to cross-VM attacks, it has been demonstrated in [9] that detecting co-residency and creating covert channel on a network is possible.

Performance Isolation The state of the art techniques for network-level isolation in Ethernet-based networks rely on VLANs and Class of Service (COS) tags. But these techniques do not scale on Clouds. Indeed, the number of VLANs is limited to 4096 and most switches support 8 COS when they support them. Furthermore, VLANs do not provide any performance isolation.

OpenFlow [43] and NOX [27] provides scalable and easier to use approaches. But they require to change all the networking devices that do not support OpenFlow. OpenNebula as OpenStack are able to use OpenFlow to create isolated virtual networks.

Bandwidth reservation mechanisms such as RSVP and MPLS- TE are limited to enforce static performance isolation. They can not be disabled when no congestion is happening and thus are not able to allow best-effort use of unused bandwidth. Max-min fair allocation is a good candidate to provide performance isolation but it is limited to a proper TCP stack. In Clouds, tenants can modified their networking stack thus a network isolation mechanism can not trust it. Performance isolation can also be done through Virtual Machines Placement algorithms [14, 36, 79] that take into account the bandwidth requirement and migrate VM if a network contention happens. These works only focus on performance isolation and can not be used to improve network security.

QCN is a new Ethernet standard for congestion control in datacenter networks [30]. By using QCN, a switch can send congestion feedback to sender thus it could fix the issue. But QCN feedback can not been send between subnets. Accordingly, this approach is not adapted for Clouds.

In [63], authors present their approach named Seawall to provide performance isolation between tenants. Seawall is using rate controllers at end hosts thus providing a decentralized and scalable service. Moreover, Seawall is based on a software thus avoiding dependency with hardware functionality. Seawall does not provide fairness between network flows but between VMs. This approach avoids that a VM with

multiple parallel network flows grabs all the network resources. But Seawall focus on performance isolation and does not provide secured isolation between different tenants.

Packet Filtering Network isolation can be done through filtering packets when they leave and enter a VM. By putting such mechanism at the hypervisor level, a malicious VM can not temper it. HostVLAN [53] segregates networks according to logical network ID associated with MAC addresses. This approach permits to avoid encapsulation of network packet or rewriting part of the packet. To prevent attacks using source-MAC-address spoofing, the source MAC address in each packet is checked before it is transmitted on the network. If a spoofing is detected, the packet is dropped. But HostVLAN can not isolate networks that span on multiple domains. OpenNebula also provides network security mechanisms [68]. But they are limited to firewall at the hypervisor level.

Overlay Networks A well studied approach to improve security on virtual network is to use overlay networks. Overlay network is a method for building a network on top of another one. The three main goals of overlay networks on Clouds are to provide an abstraction layer that isolates the virtual network from the shared physical environment on top of which it runs, to be independent of the physical characteristics of the underlying network, and to be independent of the physical location. All the overlay network approaches are running a service within the VM or at the hypervisor level that encapsulates the network packets. The main drawback of overlay networks is the overhead it adds into packet processing in the hypervisor and sometimes in the network appliances.

Project VIOLIN [34] connects VMs from different subnets to a virtual LAN. The physical links are emulated through UDP tunneling and the switches are emulated by a central service. The VNET project [70] proposes a similar approach but the tunneling is done with TCP and supports SSL. VNET remains centralized but can create direct connections between VMs. VNET/U approach [70] is running within a VM. Accordingly, the tenant is responsible to create a virtual network within its VMs. The Cloud provider does not have any way to control this virtual network. Moreover, if the VM is compromised, the whole overlay network could be compromised too. VNET/P approach [82] focuses on creating a high performance overlay network and does not provide security mechanisms. IPOP[23] proposes a mechanism to create a virtual network upon a Peer-to-Peer (P2P) overlay thus avoiding centralization related issues. IPOP supports encrypted P2P links and VPN tunnels. ViNe [75] is based on a set of virtual gateways that create a virtual network spanning multiple datacenters. Communications between the virtual gateways can be encrypted with SSH tunnels but there is no security between the VM and the switches. VANs [6, 39] is similar to VIOLIN and VNET but proposes a distributed architecture. They do not propose any encryption mechanism.

To conclude, overlay networks can be either managed by the user (inside the VM) or by the provider (at the hypervisor level). Both managements have pros and cons. If done by the user, he must know how to setup such network and is responsible of the security of his network. If one VM is compromised, the whole network security can be compromised too. But, it is easier for an user to create a network that spans over

several Clouds. If done by the provider, the creation and management of the overlay network is totally transparent for the user. Moreover, by placing the overlay network component at the hypervisor level, the network security remains even if the VM is compromised. Studies have shown that overlay networks operate at the hypervisor level have better performance by avoiding to add another layer of abstraction. But creating a cross-Clouds network requires that the different providers share a common way to manage and operate the networks and are agree to share these interfaces with others.

2.6 Discussions

As we have shown in this section, the micro-architecture of modern platforms is evolving very fast. From a single CPU processor few years ago, we have now access to massively parallel platforms with complex hierarchy of micro-architectural components.

The current trend of security in Clouds is to use virtualization as the sole mechanism to enforce security and performance isolation between users and their VMs. But the lack of proper isolation of micro-architectural components lead to the ability of creating covert channels between these VMs. It has been shown that it is possible to use these covert channels to extract sensitive informations such as cryptographic keys. But the security isolation is only one fold of a two folds issue. Performance isolation is also critical for Clouds. Indeed, if a VM is not able to efficiently use its available resources due to a noisy neighbor, it can lead to availability issue. As we have shown, a VM can applied such noisy neighbor behavior to slowdown a collocated VM. That leads the collocated VM to release resources and the VM who launches the attacks is able to extract more performance from the Cloud.

We have shown that providing mechanisms to detect such attacks is still an open issue with only few works proposing to analyze the network configuration and none being able to do it for the micro-architectural components. Even if few papers propose solutions to enforce a better isolation where micro-architectural components are shared, these mechanisms are introducing high overload.

Furthermore, we have shown that the lack of proper security and performance isolation is the same for the network than for micro-architectural components. Several works exists to extend the network protocol to increase the security and performance isolation between multiple virtual networks sharing the same hardware. But some are dedicated toward performance isolation whereas other to security. Moreover, some such as OpenFlow require specific hardware. The current trend is to either used overlay networks or OpenFlow approach. But they all have trade-off between usability, security and performance isolation.

As we have shown depending of the micro-architectural components shared between VMs, it is more or less effective to create covert channels. The same is true for performance isolation and for network isolation. The sharing (or not) of micro-architectural components and networks brings to complex security and performance

trade-offs. Providing the tenant with the ability to specify the level of sharing his VMs can accept for both network and micro-architectural components would give the ability to the user to configure the quality of isolation for each of his VMs. In the rest of the chapter, we propose an approach to provide adaptable isolation. We show how it can be ported to any resource management system at node level (hypervisor) and at Cloud level.

3 Modeling Platforms and Application for Resource Allocation

In the previous section, we show that using virtualization for security and performance isolation is not enough. Furthermore, mechanisms that enhance the quality of isolation through more control are leading to a large overhead. Through smart allocation of resources to virtual machines, it is possible to have strong isolation. Indeed, if competing workloads are not sharing the same resources, covert channels can not be created and noisy workloads are not interfering with others. But running each VMs on dedicated hardware is not possible as it dramatically reduces consolidation. By taking into account isolation within the allocation mechanism, it is possible to reduce covert channel while keeping a good consolidation of the platform. But, before being able to allocate resources to a workload on a distributed system, we need to model both.

3.1 Distributed System

A distributed system can be seen as a set of nodes (physical machines), network devices (routers, switch, etc.) and network links. Nowadays, the network devices are physical machines that provide specialized services i.e. packet routing, packet filtering, etc. Distributed system models already exist such as ADAGE [38] or the one proposed in [33]. There are also models that describe the hierarchy of a single Physical Machine (PM) such as hwloc [15]. Some standardization efforts was done in the Open Grid Forum through the GLUE Workging Group (WG). The aim of this WG is to provide a model which is expressed via a schema independent of information system implementations. The schema will define the set of attributes and attribute semantics, the relationship between the attributes and the syntax for attribute values where applicable [4]. Cloud modelization exists too for example CloudML [25] is a description language designed for Cloud environment that expresses resources, services, and requests in an integrated way.

For the isolation requirement, we need such fine grain model. Indeed, if we only have a model of the distributed system, we would be able to express the need of isolation between different VMs on different physical machines or clusters. But, this approach can reduce the usability and consolidation of the distributed systems. Indeed, a VM can block all the resources of a physical machines for others VMs

whereas it needs just a subset of the resources. It is not an issue if the user wants dedicated clusters or other blocks of the distributed systems. But if we have given to the user the capacity to express finer grained policy, it would have used it. It will be possible to have a good enough isolation for the VM without blocking all the resources of the physical machines if we were able to have a fine grain placement that takes into account the inner hierarchy of the physical machines. On the contrary, a fine grained model of a single physical machines will have expressiveness for the inner hierarchy but will lack it for the distributed systems. Thus, we need both types of models to be able to express requirements at every levels of the hierarchy of the distributed systems. But as we want to describe large scale distributed systems, we want to keep the two models separated for the sake of scalability. Indeed, performing search and modifications on large models is compute intensive and slow. Consequently, the distributed system models will provide links to each inner PM model.

3.1.1 Modeling Micro-Architectural Components

As we have presented at the beginning of Sect. 2, the architecture of modern Physical Machines (PM) is evolving fast and becomes more and more complex. Furthermore, with the increase usage of accelerators such as Intel Many Integrated Core Architecture or GPU, the physical machines are heterogenous. But to efficiently allocate resources to workloads, we need a common model that abstracts physical machines.

Obtaining models of virtualized platforms is indeed a challenge and several studies have tried to solve it for recent hypervisors. In [41], the authors study the overhead of different HPC applications over virtual machines to show their advantages and drawbacks. In [32], a basic model of two popular state-of-the-art virtualization platforms, Citrix XenServer 5.5 and VMware ESX 4.0, has been presented to be used for application performance prediction. In [73], using the vConsolidate benchmark, the authors show that modeling a virtualized environment requires to model the contentions between visible resources (cores, memory capacity, I/O devices, etc.), model the contentions of invisible resources (shared microarchitecture resources, shared cache, shared memory bandwidth, etc.), and finally model the overhead of the hypervisor. In [5], a careful attention to the network communication model within a datacenter is presented using virtual networks. It allows to get predictable data transfers.

hwloc provides a fine grained model of the processors and cache topology within a physical machine. It has been shown to be useful to allocate MPI processes and taking into account the inner hierarchy of PM. The processors are hierarchically linked based on their sharing of caches, memory or memory banks. As we have shown in Sect. 2, covert channels can arise from the sharing of micro-architectural components. Accordingly, we need a model that describes them. Using this model, we will be able to take into account micro-architectural components when allocating resources. By doing so, we will be able to share (or not) the micro-components between multiple workloads.

But, Virtual Machines can use other type of resources on a PM, mainly Input/Output (I/O) devices. The hwloc description has already been extended to include other resources such as GPU. Our model is based on hwloc description. But it is extended to include I/O devices such as hard drive disk and network cards. A description of the model is presented in Fig. 4. Each node in the model presents a device *e.g.* Core and can have parameters *e.g.* Frequency. Some nodes such as CPU are representing a set of devices and can also have parameters. The arrow represents a link between two devices. Through this model, we are able to represent both the amount of each resources the workload running in the VMs will used and the inner architecture of the PMs. We have all the information required by an allocation mechanism to select resources for each VM. Moreover, it is easy to extend the model to represent new architecture platforms and devices. Thus our model is able to adapt in advance the hardware without the need of creating a new one. Finally, by providing such abstract representation of the inner hierarchy of a PM, the allocation algorithm will be able to easily cope with new platforms without the need of in-depth modification.

Distributed System Model

We want to model the hierarchy of links and switches within a Cloud distributed between multiple datacenters. But we also want to model the geographical hierarchy of the Cloud and its datacenters. As for the PM, all the distributed systems do not share a common hierarchy. Thus, we need a model that can be adapted to those different hierarchies if we want to be able to model a large set of distributed systems. Moreover, we want to be able to extend our model to not only contain the physical hierarchy but also the functional hierarchy. A functional hierarchy describes a hierarchy that is not related to hardware resources but to other criteria such as geographical location. For example, a functional hierarchies can be used to divide a platform into availability zone that does not share the same backup components.

Not all the datacenters have the same network hierarchy. Conventional architectures rely on treelike network configuration built from expensive hardware. The current trend [2, 26] in Cloud's datacenter network is to go away from these expensive hardware. By using commodity network switches and/or physical machines for switching packets, it is possible to build more flexible datacenters for a cheaper price. Accordingly, the network topology is constantly evolving.

The same behavior is true for Clouds. For example, Amazon EC2 is distributed in 8 geographical regions with multiple datacenters with different bandwidth between each. Microsoft Azure is also distributed in 8 geographical locations but they are not the same and the network linking them is different between the one of Amazon EC2.

As for the PMs, we need a common model to describe all the datacenters and Clouds. What we want to describe is the network and functional hierarchy between the PMs. Moreover, the description will be use by the resources allocation mechanisms thus it must contain all the information they required. And, as for the PM, the datacenter and the Cloud network hierarchies are evolving. The description must be able to easily adapt to them. The model presented in Fig. 5 is able to do that.

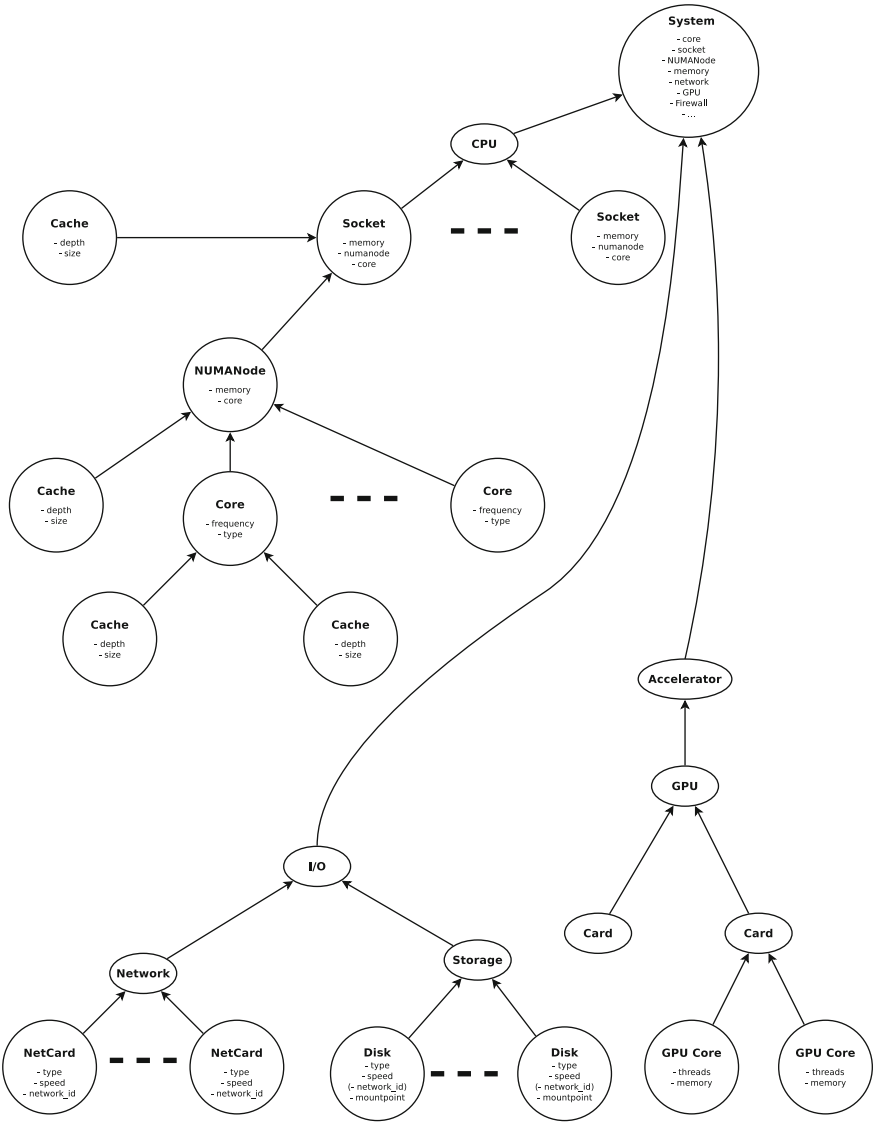


Fig. 4 Inner model of a PM

All the PMs are part of at least one group. Both PMs and groups are called nodes in the model. A group can be seen as a network shared between all its nodes. A group can be part of other groups and so on. Moreover, each PM and group can be part of a set of functional groups. Our model can be seen as a tree but with multiple roots. In most of the case, the roots are connected through links. Links are no more than a group.

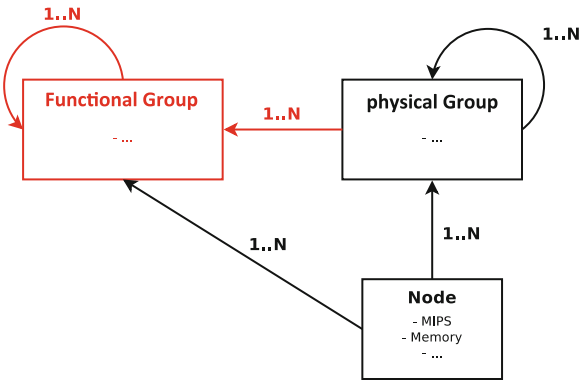


Fig. 5 Distributed system model

Each node contains concise information about its resources. For the PMs, this information describes the quantity of static resources and free resources *e.g.* 4 cores and 2 free (not used) cores. For the other nodes, this information describes the aggregated quantity of resources of all the children nodes *e.g.* a cluster has 200 cores and 20 are not used. Moreover, all the nodes except PMs contain unit capacities. Unit capacities present the largest block of resources available on at least one single children. For example, a cluster with 200 cores and 20 free cores will also have a core unit capacity of 4 if the largest number of free core on at least one of its PM is 4.

Grid'5000 Example When we want to model a new distributed system, we first need to describe its hierarchy. For the Grid'5000 platform [16], the physical hierarchy is the following one from the higher level (roots) to the PMs: Link, Site, Cluster, PM. Figure 6 presents how our model can be used to describe a platform hierarchy such as the one of Grid'5000. Each PM are part of a cluster. Each cluster are part of a site. And the sites are connected one to another through links. Moreover, as Grid'5000 is distributed at an international scale, the model also contain these geographical aspects (in red). Figure 7 describes a part of Grid'5000 platform using our model.

Amazon EC2 Example As for Grid'5000, before modeling Amazon EC2 or any other Clouds distributed between multiple datacenters, we need to describe its hierarchy. The physical hierarchy is the following one from the higher level (roots) to the PMs: Link, Region, Zone, Cluster, Rack, PM. Figure 8 presents the distributed system model for a Cloud hierarchy. As datacenters are geographically distributed, we express these functional aspects within the model. This is very important as for laws reason, we need to have access to these informations when placing Virtual Machines. Figure 9 presents a description of a part of the Amazon EC2 Cloud using our model.

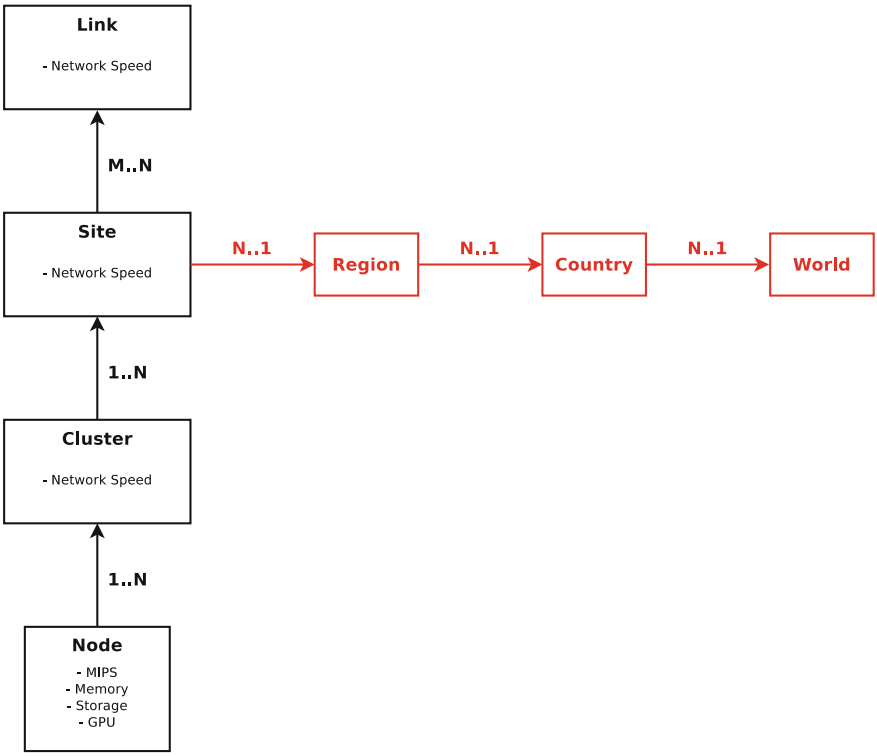


Fig. 6 Hierarchical model of Grid'5000 platform

Linking the Two Models

The distributed system model contains very concise information about resources contained in a PM. Indeed, it is required to keep this genericity to avoid to have a model too complex. If the model was too complex, it will be hard to work with it and the algorithm that manipulates it will grow in complexity and thus in computation time. However, we need a reference to the fine grained model of the inner hierarchy of each PM. Indeed, it is required to have a way to access this inner model when needed *e.g.* when an isolation requirements asks for isolation inside a PM. Thus, there is a single link between each PM of the distributed system model and an instance of the inner model specific to the PM. As Clouds and Grids have a heterogeneity limited to few tens of hardware compositions, these inner model are instantiated from a small set of inner descriptions.

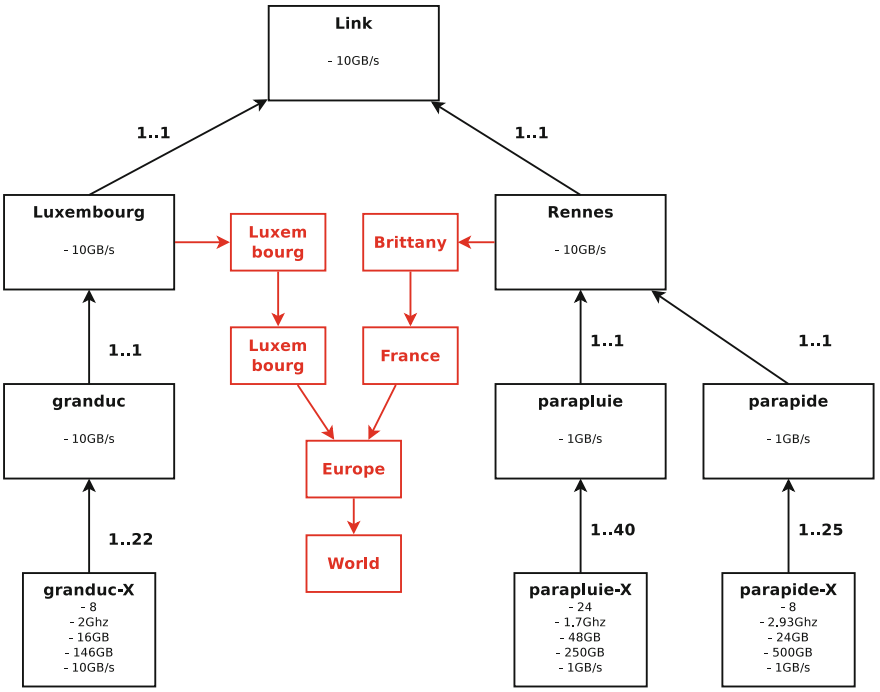


Fig. 7 Grid'5000 platform model

3.2 Application

As for distributed systems, we need a model to describe the applications. On Clouds, an application is a set of virtual resources that executes a workload. Currently, an application is a set of Virtual Machines linked through Virtual Networks. It can also include some Virtual Storage.

OVF [55] is a standard for describing the deployment organization of a set of virtual images. Its target infrastructure is the Cloud. OVF can be extended to support most of these resources requirements [22]. DADL (Distributed Application Description Language) [47] is a language description of distributed applications. It can describe applications needs. Its target infrastructure is the Cloud. Close to DADL aims, Pim4Cloud DSL is a component-based approach to model software deployment in the Clouds. It can be used to help the application designer to deploy his application in the Clouds [13].

But neither can be used to express the complex composition of virtual resources that composed the applications and the non-functional requirements on them such as the isolation requirement. We need a model that is able to represent both thus giving the ability to the user to express their isolation requirements on each part of their workloads.

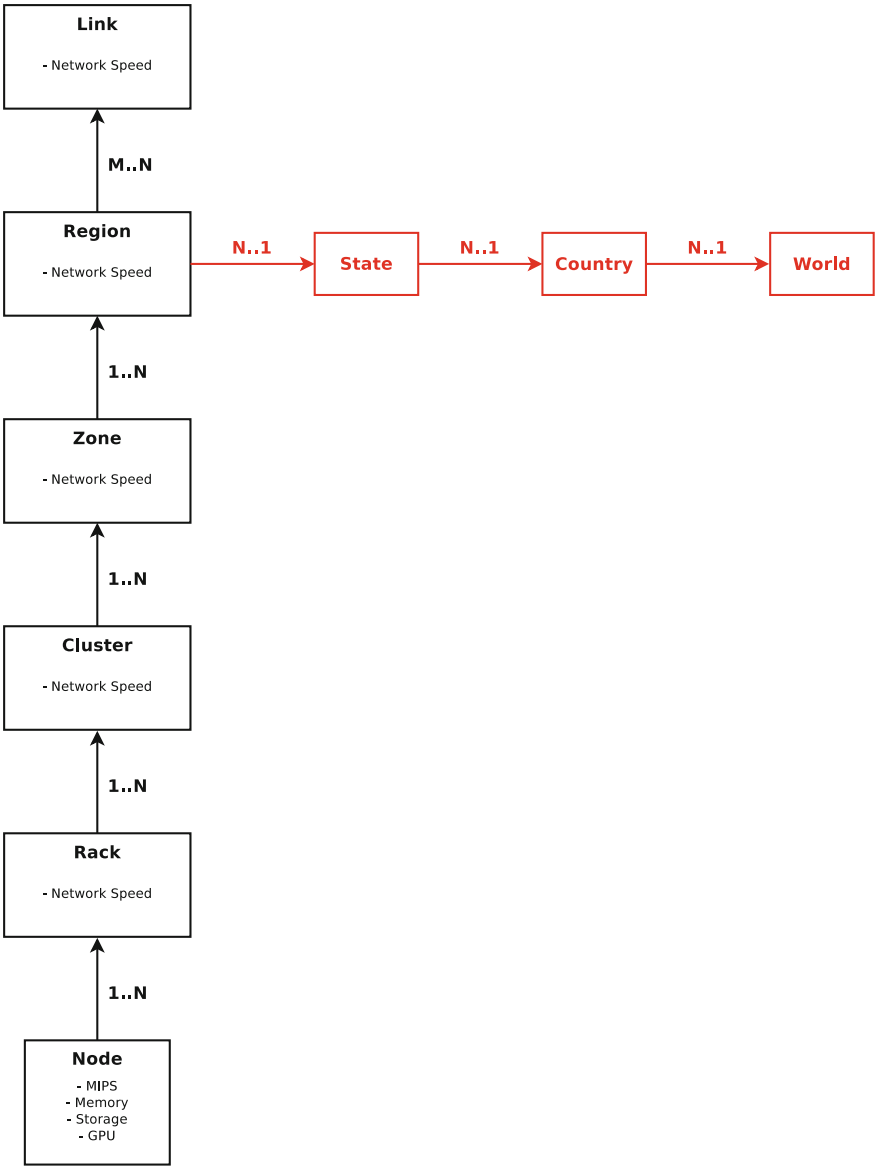


Fig. 8 Description of a cloud hierarchy using our model

The model we proposed here is oriented toward the description of applications built from Virtual Machines, Virtual Storage, and Virtual Networks but it can be extended to support more components e.g. load balancers. In addition, the model also includes non-functional requirements such as isolation ones. Figure 10 presents

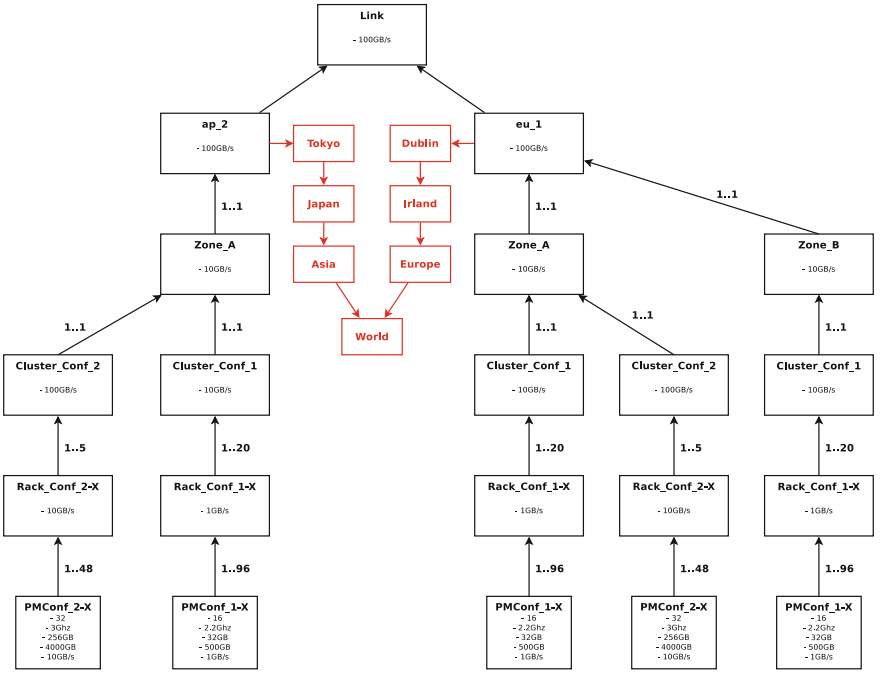


Fig. 9 Amazon EC2 platform model

our application model. The description of a Virtual Network (Network) is minimalist: its name, its type e.g. InfiniBand or IP and optionally a list of IPs affected to it.

The description of a Virtual Machine is divided into three modules:

Virtual Image (**Image**) it describes a filesystem. The based parameters of an Image are: Name, Path e.g. to the file that contains the file system and its size. On Amazon EC2, Virtual Image are called AMI for Amazon Machine Image.

Template it describes the resources requirements of the VM. It has at least the following parameters: Name, Number of core (core), and memory Size (memory). Each template is linked to at least one Image (the one that contains the Operating System of the VM). But a template can be linked to a set of Image e.g. a data storage inside a file. Moreover, a template is linked to at least one Virtual Network (Network) but can be linked to several of them. On Amazon EC2,part of Templates are called Instance Types but the link between an Instance Type and an AMI is only done when a VM is requested. Thus, a Template is the combination of an Instance Type and an AMI.

VM it describes a single Virtual Machine. It is linked to one and only one template. The VM is an instantiation of the linked template. A template can be instantiated multiple times. This multiple instantiation is represented by a number on the arrow between the VM and Template modules. On Amazon EC2, a VM is called an Instance.

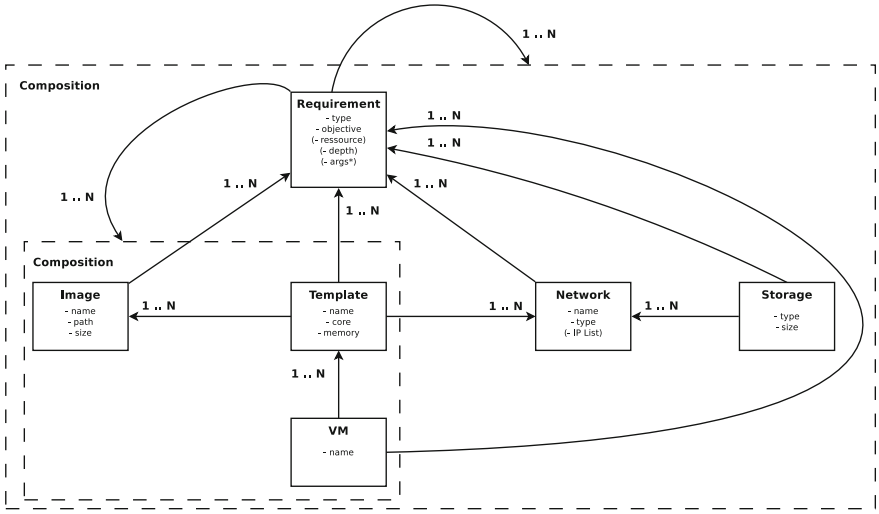


Fig. 10 Application model

Modern Cloud applications are also using Virtual Storage such as remote block storage and databases. Accordingly, our model is able to describe them. A Virtual Storage is composed of a type e.g. block, key-value or relational database and a size.

Our application model also contains a special module called composition. A composition contains a set of modules (a subset or all of the application modules). It is used as a easy way to create links between groups of modules and/or single modules. For example, a set of Templates is using the same set of Images.

The Requirement module allows a user to express the non-functional requirement on its application. Here we focus on isolation requirement. The isolation requirements are described by the Requirement module. This requirement can be linked to a VM, a template or an Image. It can also be linked to a composition. It is also possible to link requirement to a Virtual Network or a Virtual Storage. It can be used either for performance or security isolation. Moreover, the requirement module is not limited to User's Isolation Requirements but can described other one such as profile isolation, template isolation or redundancy that are not presented here.

A Requirement takes between three and four parameters: an objective, a type, a level whereas in the distributed system or in the PM hierarchy and a list of users if needed. The objective is the requirement name e.g. User Isolation. For the Isolation Requirement, the types are:

Alone : the user requests to be strictly alone;
Friends : the user only accepts to share resources with a set of users;
Enemies : the user never wants to share resources with a set of users;

Using the isolation requirements, a user can request to never share L2 cache with other users, to allow sharing of PMs with a set of users he trust e.g. from the same

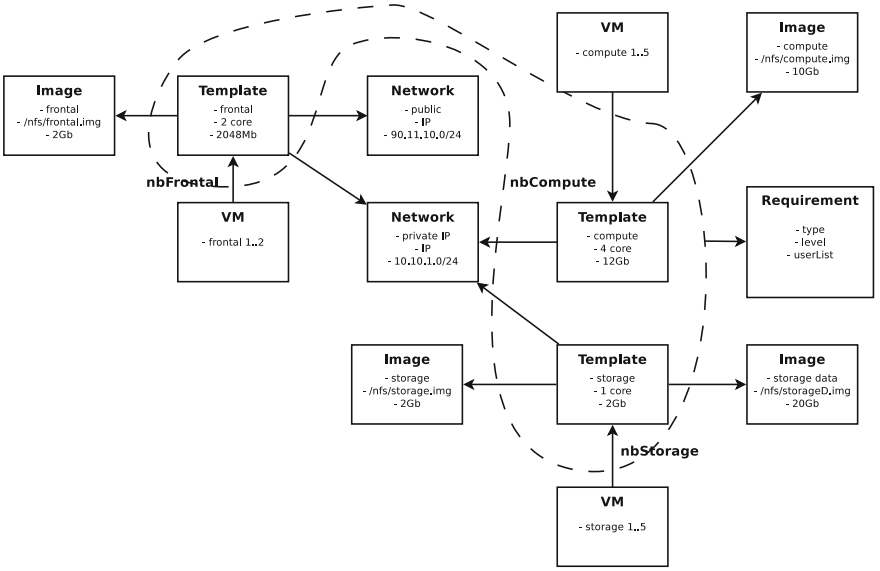


Fig. 11 Virtual cluster model

company and to never share a cluster with another user he do not trust and/or who is known to be a noisy neighbor.

Example: Virtual Cluster In this chapter, all the experimentations have been done using the modeled application described in Fig. 11: Virtual Cluster. It is a simple cluster description: a set of frontal VMs, a set of storage VMs and a set of computing VMs. The frontal VMs are connected to a public network (public) such as Internet. All the VMs are interconnected through a private network (private). Each template is linked to an operating system Image. Moreover, the storage Template is also linked to another Image that acts as storage filesystem. The three templates of the application are part of a composition. An User's Isolation Requirement is applied to this composition. In this application example, we do not use the module Virtual Storage as we built a dedicated virtual storage out of virtual machines. The storage part could be replaced by a Virtual Storage module if the Cloud Service Provider has such storage offer.

3.3 Discussions

With the distributed system we have proposed in Sect. 3.1 and the application model introduced in Sect. 3.2, we are able to describe both the platform and the applications that run on it. The purpose of these models is to ease the resources allocation process and takes into account both hardware and non-functional hierarchy. As we have

shown, in Sect. 2, it is required to know the hierarchy of a platform to avoid sharing hardware components that could lead to covert channels. But allocating dedicated resources to each VM is dramatically decreasing the consolidation of a platform and not all workloads required it. We have shown that our application model allows a user to express his isolation requirements on all the parts of his application. In the next section, we show how all these informations are used to allocate resources to applications.

4 Resource Allocation with Security Requirements

To help guide the placement of virtual machines, the main approach is to use constraints which reflect the goals expressed by the user and the entity that operates the infrastructure. By expressing the placement problem as Constraint Satisfaction Problems (CSP), it is possible to use a solver to resolve it. Constraints can represent different goals such as energy saving [51]. But solving large problems is time consuming thus using a decentralized and hierarchical approach can increase the computation of a solution [20, 21].

The principle of using a linear program or a linear system to solve the problem of placement of virtual machines in the Cloud with constraints has been studied numerous times as in [64, 74]. The constraints can modelize numerous goals such as performance and availability [33]. These goals are then translate into collocation and anti-collocation constraints. As for the CSP approach, linear program approaches do not scale. To improve the performance of placement algorithm, hierarchical representation allows to reduce the search space [33]. As numerous works propose algorithms that takes into account constraints in VM placement, [46] propose a method of comparison and evaluation of these different algorithms. In [69], the authors present an approach for sharing cluster resources among competing sequential and parallel jobs. The connection between nodes and datacenters is sometimes taken into account like in [11, 44], and [18].

Avoiding cache and processor resource contention through scheduling algorithms is a well studied approach [10, 45, 86]. At the cluster level, numerous works have also been done [40, 67]. The purpose of these approaches is to increase performance isolation between processes.

The current issue is to have efficient and scalable algorithms that are able to work at the scale of multi-datacenters Cloud infrastructure. (CSP) or linear program approaches do not scale well. Indeed, placing VMs on Clouds can be formalized as a multi-dimensional bin-packing. Therefore, the complexity of solving such problem is NP-Complete. Heuristics allow to have almost optimal solutions with a reduced complexity. In this chapter, we propose to use such heuristics. They are able to take into account isolation requirements while placing VMs on a multi-datacenter Clouds or any infrastructures described through the distributed system model. But our models are not dedicated toward heuristics. They could also be used by CSP or linear program approaches.

4.1 User Isolation Requirement

The purpose of the user isolation requirement is to enforce isolation between users. As previously stated, the isolation requirement has two goals:

- increasing the security of a user by reducing the number of covert channels e.g. L2 cache;
- avoiding interference from noisy neighbors *e.g.* another application that has the same behavior on constraint resources such as CPU cache or network links.

The requirement has a scope where it applies *i.e.* a level in the hierarchy of the distributed systems. For example, if the requirement's scope is PM, it will apply to all the level below the PM *e.g.* NUMA and socket and at the PM level. As we previously say, the isolation requirement has three different types: alone, friends and enemies.

First, we propose an algorithm that verifies if a node of the distributed system can fulfill a given isolation requirements. Secondly, we propose an algorithm that verifies if the placement of a VM on a node does not contradict the isolation requirements of already placed VMs. The same algorithms can be used for the placement of VM inside a PM by modifying the lowest level of hierarchy from PM to Core.

Verify if a node respects a requirement Algorithm 1 checks if a node $curNode$ and all its children fulfill an user isolation requirement req. In this algorithm, the set of users for the isolation requirement's type $FRIENDS$ and $ENEMIES$ are represented as a bit vector. A bit i is set to 1 if the user with the id i is part of the set and 0 if it is not the case. Each node also contains a list of users ($node_{users}$) that are already using its resources. This list is also represented as a bit vector.

The Algorithm 1 is divided in three main sub function. Each algorithm check the requirement if its type is respectively $ALONE$, $FRIENDS$ or $ENEMIES$.

Algorithm 1 Checks if a node $curNode$ and all its children fulfill an isolation requirement req requested by user $user$: $Isolation()$

Require: $curNode, req, user$
  if $req_{type} = ALONE$ **then**
    return **IsolationAlone**($curNode, req, user$)
  else if $req_{type} = FRIENDS$ **then**
    return **IsolationFriends**($curNode, req, user$)
  else if $req_{type} = ENEMIES$ **then**
    return **IsolationEnemies**($curNode, req, user$)
  end if

Verify if a VM can be accepted on a node But it is not enough to verify if a node respects an isolation requirement of a VM. We also need to check if the requirements of the already placed VMs on the node will be respected if the new VM is placed there. To simplify this verification, when a new VM is started on a node, its requirements are translated into two user lists ($allowedUser$ and $deniedUser$) and one boolean $alone$ that are stored as variables on the node. In practice, if a friend $userList$ is

expressed by the VM it is added to the $allowedUsers$ list and the same is done with an enemies $userList$ in the $deniedUser$. Moreover, if the VM has a isolation requirement with the type $alone$, the boolean $alone$ on the node is set to $true$.

4.2 Heuristics

BestFit

Algorithm 2 provides a way to select a PM where a VM can be placed following the bestfit approach. The isolation requirements are taken into account using concepts introduced previously. Algorithm takes into account the hierarchy. It applies the sorting algorithm at each level of the hierarchy. First, it sorts all the root of the distributed systems and selects the best fitting one. Then it does the same with each child of the best fit node. Recursively, it reaches a subset of PMs e.g. the PMs belonging to a cluster that fits the capacity and isolation requirements and returns this list. The list can be then sorted. It uses a Breath First Search approach (BFS) to navigate through the distributed system hierarchy.

BestFit with Properties

The second Algorithm designed called BestFit with properties is a variant of Algorithm 2. But contrary to the previous one, it starts to use best fit only when it reaches a level in the hierarchy that is equal to the higher level expressed by the isolation requirements designed by a $maxLevelRequirement$ variable. Thus, it will iterate through all the branches of the tree at the beginning. By doing so, the algorithm is able to choose the best fit node at a specific level in the hierarchy and is not limited to the children of the best fit node of each level. Unless the higher level expressed by isolation requirements is the higher level of the hierarchy, the algorithm will iterate through more nodes but it can find a best fit nodes that are behind a non best fit nodes. For example, an application specified an isolation requirements at the cluster level. If using the first version of the best fit hierarchy-aware algorithm, it will select a cluster in best fit site. With this algorithm, it will iterate through all the sites and all the clusters belonging to each site. It will then select the best fit cluster.

5 Use-Cases: Virtual Clusters on Clouds

To study the impact of isolation on performance and consolidation, we have done numerous experimentation with MapReduce as our case of study. We defined three levels of isolation:

Algorithm 2 BestFit algorithm for an instance of the template vmt on a set of nodes $nodeList$: $BestFit()$

```
Require: listNode, vmt
  fitNodes ← ∅
  for all curNode ∈ listNode do
    if isAllow(curNode, vmt_user) then
      if checkCapacity(curNode, vmt) then
        checkReq ← true
        for all req ∈ vmt_requirements do
          checkReq ← checkReq ∧ Isolation(curNode, req, vmt_user)
        end for
        if checkReq then
          fitNodes ← fitNodes ⋃ curNode
        end if
      end if
    end if
  end for
  sort(fitNodes)
  if size(fitNodes) > 0 then
    if fitNodes[0]_level = PM then
      return fitNodes
    else
      for all curNode ∈ fitNodes do
        nextFitNodes ← BestFit(curNode_children, vmt)
        if size(nextFitNodes) > 0 then
          return nextFitNodes
        end if
      end for
    end if
  else
    return ∅
  end if
```

1. **No isolation**: the workload shares a MapReduce cluster with other workloads.
2. VM **isolation**: a set of VMs run a dedicated MapReduce cluster for the workload. The VMs have no isolation requirements.
3. **Host isolation**: as for VM isolation, the workload runs in a dedicated cluster. Furthermore the VMs are deployed with an isolation requirement expressing to be alone at the PM level.

5.1 Scenarios

We create 11 experimentation scenarios with different combinations of isolation levels.[3] For all the scenarios, we have 3 users with a MapReduce workload

[3] Each scenario has been launched 10 times, the results presented here are the average of these executions

Table 1 Level of isolation per user for each scenario

	User 0		User 1		User 2	
	Isolation	# of VMs	Isolation	# of VMs	Isolation	# of VMs
1	1	21	1	21	1	21
2	1	14	1	14	2	7
3	2	7	2	7	2	7
4	1	21	1	21	1	21
5	1	14	1	14	2	7
6	2	7	2	7	2	7
7	1	14	1	14	3	7
8	2 with user 1 3 with user 2	7	2	7	2 with user 1 3 with user 0	7
9	1	14	1	14	3	7
10	2	7	2	7	3	7
11	3	7	3	7	3	7

(HADOOP-BLAST workload). For each user, the workload is launched 3 times. The scenarios are summarized in Table 1, the number of VMs is the number of VMs that compose the MapReduce cluster where the workloads of the user are running. The first three scenarios do not take into account isolation requirements while all the other do. We have used OpenNebula [66] as IaaS Cloud middleware and a customized version of Haizea [50, 65] to allocate resources to VMs and take into account isolation requirements.

In Scenarios 1 and 4, all the workloads are launched on a shared MapReduce cluster composed of 21 VMs. In Scenarios 2 and 5, the workloads of the user 0 and 1 are launched on a shared MapReduce cluster composed of 14 VMs and the workloads of the user 2 are launched on a dedicated MapReduce cluster composed of 7 VMs. In Scenarios 3 and 6, each user runs his workloads on a dedicated MapReduce cluster composed of 7 VMs. In Scenarios 7 and 8, the user 0 never shares a physical machine with user 2. In Scenarios 7, user 0 and 1 share a MapReduce cluster of 14 VMs and user 2 has a dedicated MapReduce cluster composed of 7 VMs. In Scenario 8, each user runs his workloads on a dedicated MapReduce cluster composed of 7 VMs. Moreover, due to the isolation requirements, the VMs of the user 1 can be on the same physical machines than the ones of the user 0 or 2. In Scenarios 9 and 10, the user 2 never shares a physical machine with other users. In Scenario 9, user 0 and 1 share a MapReduce cluster of 14 VMs and user 2 has a dedicated MapReduce cluster composed of 7 VMs. In Scenario 10, each user runs his workloads on a dedicated MapReduce cluster composed of 7 VMs. In Scenario 11, each user never shares a physical machine with other users and they all have a dedicated MapReduce cluster composed of 7 VMs.

5.2 Experimentation Platform

We use Grid'5000 to run our experimentation. Each scenario has been ran 10 times. We use a set of six homogeneous physical machines linked through a 1 Gb/s network. Each of the physical machines have 2 AMD Opteron™ 6164 HE 1.7 GHz for a total of 24 cores and 48Gb of memory. One physical machine is used as OpenNebula controller and Virtual Machine image repository. The five others host the Virtual Machines.

We use OpenNebula 3.2 for our experimentation on Debian Squeeze 64bit. For every experimentation, we use MySQL as a backend for OpenNebula. Indeed, SQLite has been shown to be slower than MySQL. OpenNebula is setup to use non-shared file system (`scp` between the controller and the other physical machines). We use KVM as hypervisor. Moreover, we install ebtables[4] to isolate the network of each user or setup a shared isolated network for 2 or 3 users. This isolation is only for security purposes, there is no network performance isolation in this experimentation. When we use Haizea as a scheduler, it is a modified version that takes into account our isolation requirements. Moreover, it is modified to implement a ranking algorithm (bestfit) based on resources used on each physical machine and has been extended to support OpenNebula 3.X API.

Each time we launch an occurrence of a scenario, we reinstall all the physical machines from scratch. The installation and configuration of OpenNebula is done from the source. It is automated by a Chef script.[5] The same is done for the MapReduce cluster. We use HADOOP 0.20.2 for HADOOP File System and MapReduce on a Debian Squeeze 64bit. The MapReduce workload is a HADOOP BLAST application.[6]

5.3 Results

Deployment

The first step of each experimentation is to deploy a set of VMs on which the MapReduce cluster(s) are instantiated. Figure 12 shows the amount of seconds it takes to deploy all the VMs linked to an user. As under some scenarios, the VMs are shared between multiple users, the deployment time is the same. For example for the scenarios 1 and 4, the deployment time is the same for the three users. For the scenarios 2 and 5, the scenario is the same for the user 0 and 1.

The deployment time is ranging between 300 s and 380 s for between 7 and 21 VMs. The deployment time does not change a lot when the number of VMs to deploy

[4] The ebtables programs is a filtering tool for a Linux-based bridging firewall. http://ebtables.sourceforge.net/

[5] http://wiki.opscode.com/display/chef/About

[6] http://salsahpc.indiana.edu/tutorial/hadoopblast.html

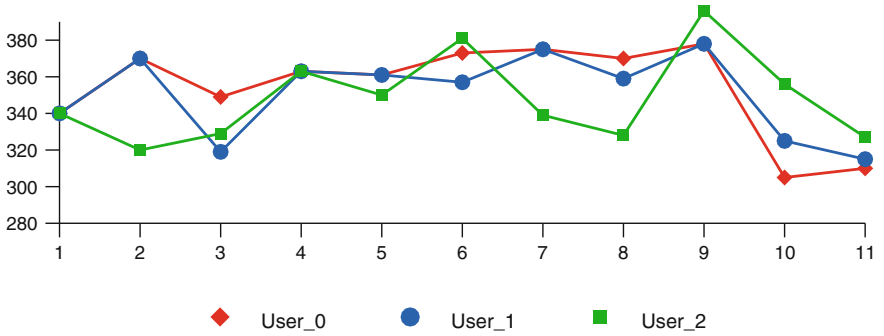


Fig. 12 Deployment completion time for each scenario and users

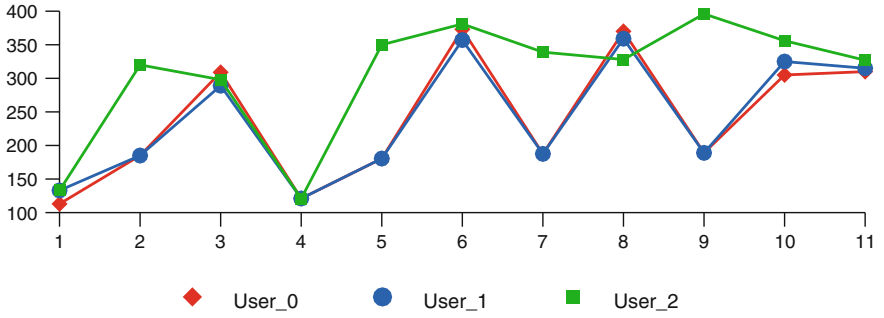


Fig. 13 Normalized deployment completion time for each scenario and users

increase. For example, in the scenario 5, deploying 14 VMs only takes 10 s more than deploying only 7.

OpenNebula and Haizea have almost the same deployment time. Furthermore, our modified version of Haizea does not take longer to compute a placement and deploy the virtual machines than the default OpenNebula scheduler. Indeed, most of the deployment time is due to the network transfer between the Cloud controller and the other physical machines. As all the VMs are deployed concurrently there is a contention on the network link of the Cloud controller. This contention is the slowing factor.

Figure 13 presents (in seconds) why it is interesting for a MapReduce as a Service (MRaaS) provider to deploy a cluster shared by multiple users. Indeed, as we say earlier, the deployment time is weakly linked to the number of VMs. Thus, at shown in the Fig. 13, deploying 21 VMs for three users reduces the amount of deployment time per user. Indeed, it almost divides by three the deployment time that we would had if we have deployed a 7 VMs cluster for each user.

To conclude, in term of deployment time, it is cheaper to deploy a large MapReduce cluster for multiple users than to deploy a dedicated small MapReduce cluster for each user.

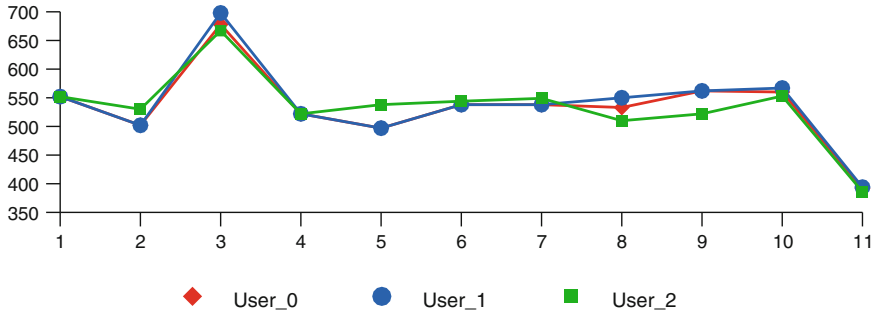


Fig. 14 Contextualization completion time for each scenario and users

Contextualization

After deploying the virtual machine, the contextualization process downloads, installs and setups the MapReduce cluster and the HADOOP-BLAST workload. As for the deployment time, the contextualization time is weakly linked to the number of VMs. Figure 14 shows the completion time (in seconds) of the contextualization process for each user under each scenario. The only improvement is during the scenario 11 when each user has dedicated physical machines. As for the deployment time, this is due to the network contention.

Figure 15 shows (in seconds) why it is interesting for a MRaaS provider to share a large MapReduce cluster between multiple users. Indeed, as most of the contextualization can be done in parallel without concurrency, it is cheaper to contextualize large cluster for multiple users than small cluster dedicated to each user.

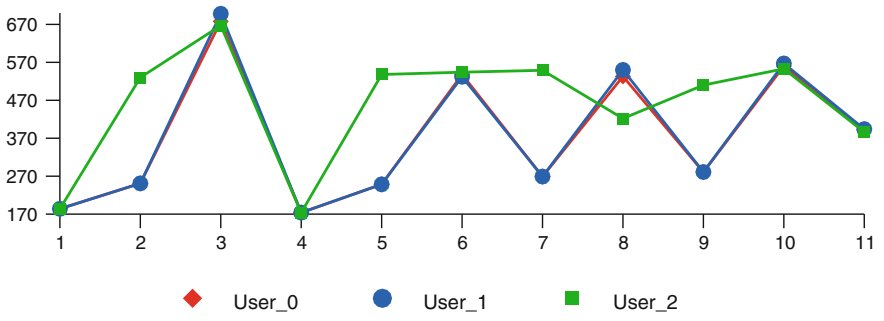


Fig. 15 Normalized contextualization completion time for each scenario and users

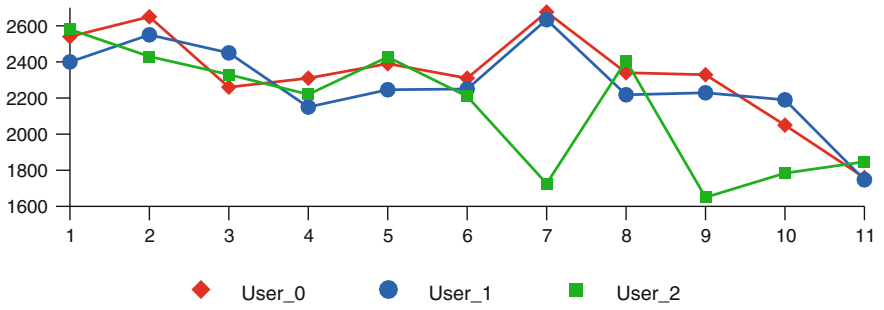


Fig. 16 MapReduce completion time for each scenario and users

Performance

Figure 16 shows the completion time[7] (in seconds) for each user under each scenario. Shared clusters with concurrent workloads do not impact the performance. The size of the MapReduce clusters does not have a big impact on performance too. The only real impact that can be seen based on the experimentation is when the MapReduce clusters are running on dedicated physical machines. For the scenarios 7, 9, 10 and 11, the user 2 has dedicated physical machines to run his MapReduce cluster due to the isolation requirements. In these scenarios, the completion time of the user 2 is decrease of 300 s in average.

Figure 17 shows the decrease of completion time in percentage in comparison of the average of completion time for the three users. The size of the MapReduce cluster and the number of workloads that shares the MapReduce cluster produce an increase or decrease of 10 % on the completion time. But the isolation of the MapReduce cluster on dedicated physical machines improves the performance by up to 25 %

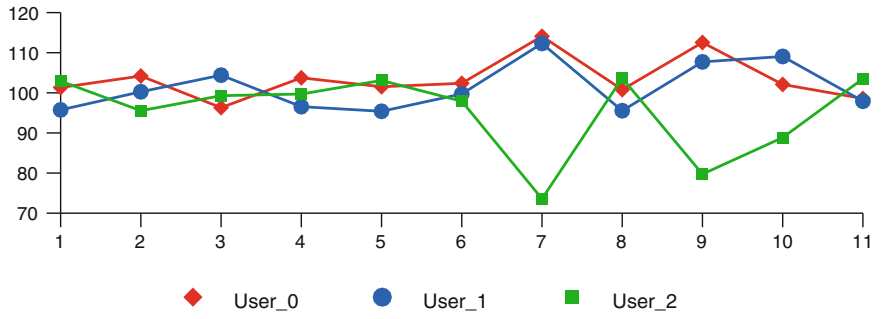


Fig. 17 Normalized MapReduce completion time for each scenario and users

[7] The completion time of the MapReduce workload is the sum of execution time of three HADOOP BLAST workload.

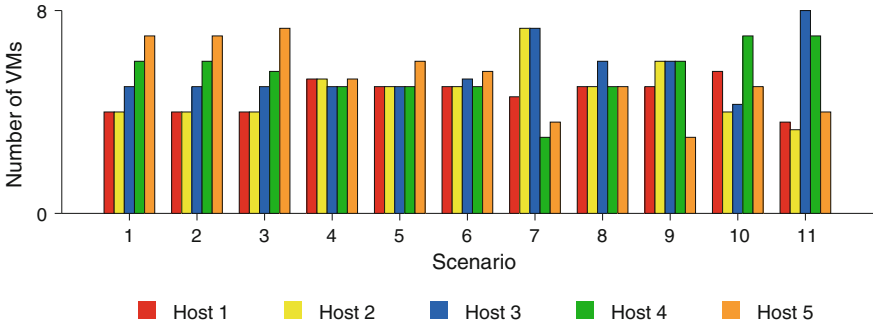


Fig. 18 Number of VMs per host for each scenario

in comparison of workloads that run on a shared MapReduce cluster (scenarios 7 and 9) or on shared physical machines (scenario 10).

To conclude, the main factor to increase the performance is the physical isolation of workloads. Thus, the decrease of the performance on shared MapReduce cluster or on shared physical machines must be due to contention on memory channels and/or on processor caches.

Consolidation

We use two metrics to evaluate the consolidation of the overall Cloud platform. The first one is the amount of VMs per physical machine. The Cloud Provider wants to maximize the number of VMs per physical machine. Indeed, if he can run more VMs on the same platform, he can increase the amount of money he earns. The second metric is the amount of hosts used by a user. The Cloud Provider want to balance its platforms. Therefore, if the users can be spread equally between all the physical machines, it is easier to do the balance.

Figure 18 shows the number of VMs per physical machines for each physical machine under each scenario. As one can see by comparing the scenarios 1, 2, 3 and the scenarios 4, 5, 6, our modified version of Haizea (with a bestfit) is doing better than the default OpenNebula scheduler. As predicted, the isolation requirements impact the number of VMs per physical machine as some VMs can not be placed on all the physical machines. The worst case is the scenario 11 where each user has dedicated physical machines. When one user has dedicated PMs (scenarios 7, 9, 10), it is not as bad as for the scenario 11 but the overall consolidation decreases in comparison of the scenario where there is no isolation requirement. The scenario 8 is a special case as it is the only one where there is isolation requirements but one of the users can share physical machines with the other two. In this case, the consolidation is the same than under the scenario where there is no isolation requirement.

Figure 19 allows to understand the decrease of consolidation. Indeed, when there is no isolation requirement, all the VMs of each user are balanced between the 5 physical

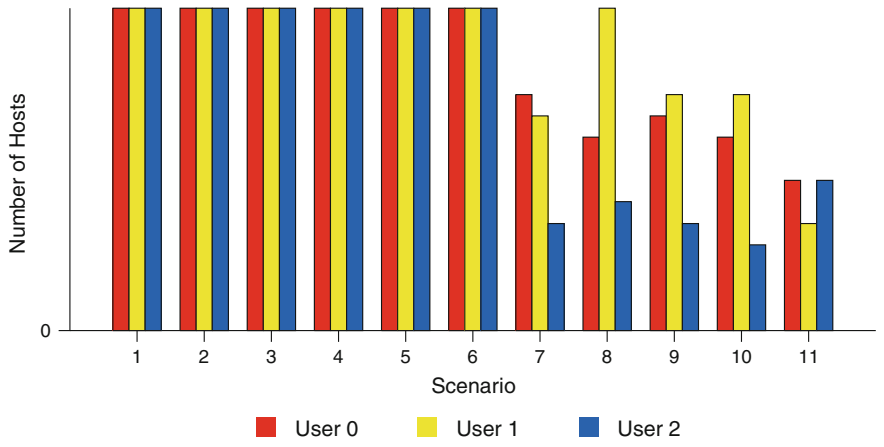


Fig. 19 Number of hosts used per user for each scenario

machines. But with isolation requirement, this balance is reduced to between 1 and 3 physical machines. The only exception is for the scenario 8 where the user 1 can shared physical machines with the other two. In this case, the user 1 has VMs on all physical machines.

To conclude, as we predict earlier, the isolation requirements have an impact on the overall consolidation of the platforms. Moreover, it is more interesting to have small groups of adversary users than users that want dedicated physical machines. With a large number of users with only few adversary users, the consolidation of the platform can remain good.

5.4 Analysis

Based on our MapReduce experimentation, we have demonstrated that the isolation requirements must be billed to the users by the IaaS provider. Moreover, the billing must be per adversary. Each time, a user declares another user as an adversary, he must paid an extra fee. Furthermore, when a user wants dedicated physical machines, he must be billed for all the resources he blocks for other users and not only for the resources he actually uses.

For MRaaS provider, it is more interesting to isolate his users on different VMs than on different physical machines. Indeed, it will cost more for him to isolate his users at the physical level as the IaaS provider will billed for that. But the increase of performance when using dedicated physical machine can be interesting for some users who are ready to pay more for better performance.

6 Conclusion

In this chapter, we have presented how new micro-architectural components and networks can be used to leak information from and to degrade performance of others users on a Cloud. Even if it is not presented here, the same is true for storage [29]. Covert channels and noisy neighbors attacks are due to the sharing of resources between users and the lack of proper isolation between resources. Protection methods exist but they bring a large overhead or only partially protect users against such attack. Accordingly, a general purpose approach to regulate the sharing of resources is required. Furthermore, as there is a trade-off between security, performance and consolidation, we present an approach that gives the power of selecting the proper level of resource sharing to the users. Isolation requirements give such capacity to the user. The selection of proper quality of isolation could be guided by an application that has an in-depth knowledge of the trade-off between the different kinds of isolation, performance and consolidation. For example, given a budget, an application and a SLA, *e.g.* a throughput, it would be possible to select the proper quality of isolation.

First, we have motivated the introduction of new models to formalize both Cloud infrastructures and applications running on it. The Cloud infrastructure model is divided into two models: one for the inner architecture of each physical machine and one for the distributed system. Furthermore, we show that both are hierarchical platforms and thus provides models that are able to take into account this hierarchy. Our model does not focus on the physical hierarchy, it is also possible to formalize the functional hierarchy, *e.g.* the geographical location. The IaaS Clouds provide three types of resources: compute, storage, and network. The application model we have introduced allows to formalize any application using a composition of these resources. Moreover, the model allows to put isolation requirements on a subset or all resources that compose an application.

Using both models, we have shown how a classic resource allocation algorithm can be extended to support these isolation requirements. In this chapter, we focused our presentation on how to modify computing resource allocation algorithms. We showed that taking into account these requirements has an impact on performance and consolidation. Therefore, the study of this trade-off must be extended to extract a proper model representing it. The isolation must be billed depending of the level at which the user requests it. Indeed, requesting to be isolated alone at the cluster level and only using one physical machine will left all the other physical machines idle. Such waste of computing power must be billed to the user. Therefore, a billing model that takes into account isolation requirements must be proposed. To be fair, this model must be based on extended studies of the trade-off and reflects the real loss of consolidation.

In this chapter, we only presented how to modify computing algorithms. But network and storage allocation algorithms must also be modified in the same way to provide an end-to-end integration of isolation requirements within the Cloud infrastructure. As for computing resources, different levels of isolation and trade-offs exist and must be studied. But to improve the isolation and avoid to lose too much

consolidation, compute, storage, and network, co-allocation algorithms must be proposed. Indeed by allocating all resources through a single algorithm, it will be possible to improve the sharing of resources and the way they are allocated.

We have focused on taking into account hardware isolation requirements. But software isolation requirements such as protection mechanisms [1] exists. By expressing the availability of such mechanism through the functional aspects of the distributed system model, our model can easily represent them and resource allocation algorithms can be modified to take them into account. Moreover, the formalization of requirements in the application model is flexible enough to express these kind of requirements. Accordingly, there is no fundamental change to do to support new requirements. We will have a large base of isolation mechanisms (hardware and software). Accordingly, we will need to classify them based on their quality of isolation and the related trade-offs. This classification is required to help the user select the proper isolation mechanisms.

As we have shown through resource allocation, it is possible to increase the quality of isolation between users. Such isolation does not require any configuration of the hardware resources. The allocation process is enough. By taking into account software isolation for the compute, storage, and network isolation, a configuration process will be required. It will be automatically configured based on the isolation requirements. For example, if a VLAN isolation is required, all the VMs of a user must be put into this VLAN. Therefore, the hypervisor must create a virtual network card that takes into account such requirement. Accordingly, the resource allocation algorithms must collaborate with self-configuration mechanisms that will automatically configure isolation mechanisms. Our models already support the expression of isolation requirements that go further than hardware isolation. The next step will be to create self-configuration mechanisms that are able to to take into account software isolation requirements.

For example, the virtual cluster application presented in Sect. 3.2 can be modified to include network isolation requirements. The modified application model is presented in Fig. 20. There are two compute isolation requirements: one hardware and one software. The hardware one states that all VMs must never share NUMA with VMs from others users. This requirement does not require configuration. The software one states that all the hypervisors where the VMs are deployed must have Mandatory Access Control mechanisms such as [1]. As all access control mechanisms, MAC requires policy. Accordingly, a proper policy must be sent to all the hypervisors selected and loaded by the MAC mechanism before the deployment of the VMs. The network *private* IP is connected to all the frontal, compute, and storage VMs. This network has a network isolation requirement: overlay with SSL. Therefore, at deployment time, a suitable overlay mechanism must be selected and configured for each VM. This overlay must be able to create encrypted tunnels between all the VMs. IPOP [23] is a good candidate. For the network *private storage* IP, the isolation requirement asks for VLAN isolation. Accordingly, an available VLAN id must be selected and configured for each storage VM. If required, network applicances must also be modified to configure the VLAN.

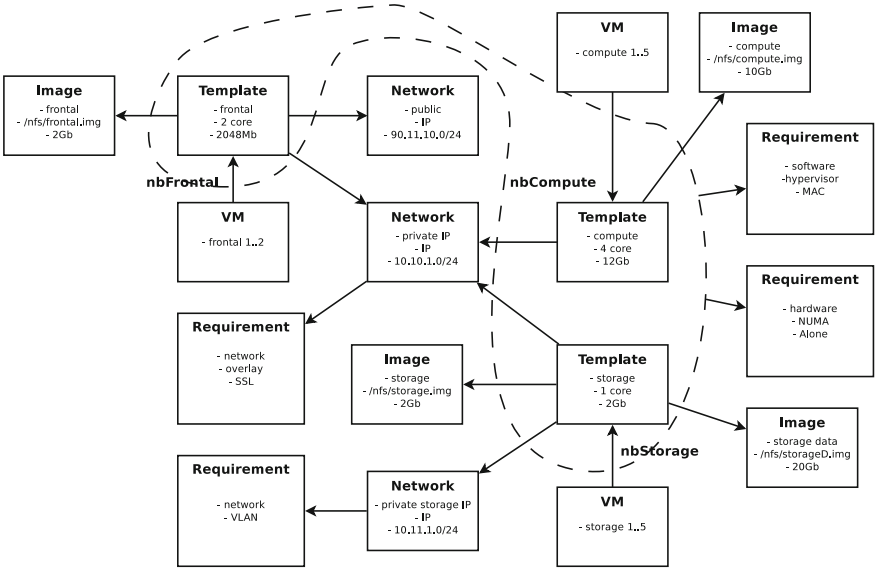


Fig. 20 Virtual cluster model with compute and network isolation requirements

In this chapter, we have focused our study on Infrastructure as a Service Clouds and application using them. But Platform as a Service (PaaS) and Software as a Service (SaaS) platforms should also benefit from enhanced isolation mechanisms. Running PaaS or SaaS on top of an IaaS Cloud that supports isolation requirements will be a first step but it is not enough. Taking isolation requirements directly at the PaaS and SaaS level should be better. At the IaaS level, the isolation requirements presented in this chapter should be used. At the PaaS level, Operating System isolation could be used as the one presented in [17]. For example, Java Virtual Machines are known to have isolation weakness [61]. By isolating each user in a dedicated JVM and controling interaction between them using a MAC mechanism, it could be possible to enhance the isolation. Others mechanisms could be used as a modified JVM with MAC mechanism to control interaction inside a JVM. At the SaaS layer, a mechanism, such as the one presented in [31], must be included into the software to control the interaction between the users and their data. Futhermore, all these mechanisms must collaborate to provide an in-depth security spanning through the 3 layers. The same kind of modification of resource allocation and studies of trade-off must be done. The goal would be an integration of isolation requirements in the three layers.

By proposing a general purpose mechanism that is able to take into account isolation requirements for resource allocation and configuration and that is able to manage the three layers of Clouds and make them collaborate, we will be able to propose an in-depth and end-to-end isolation. We plan to apply the roadmap we highlighted in this section and thus provide such functionality in a near future. Furthermore, thanks to our general purpose models, other can take into account and

implement isolation requirements (or other security requirements) in their Cloud stacks. Therefore, our approach can be the foundation of an eco-system of Clouds with enhanced and easy to use security.

References

1. Afoulki Z, Bousquet A, Briffaut J, Rouzaud-Cornabas J, Toinard C (2012) MAC protection of the openNebula cloud environment. In: International conference on high performance computing and simulation (HPCS), pp 85–90
2. Al-Fares M, Loukissas A, Vahdat A (2008) A scalable, commodity data center network architecture. SIGCOMM Comput Commun Rev 38(4):63–74
3. Ali Q, Kiriansky V, Simons J, Zaroo P (2012) Performance evaluation of HPC benchmarks on VMware's ESXi server. In: Proceedings of the 2011 international conference on parallel processing, Euro-Par'11. Springer, Berlin, pp 213–222
4. Andreozzi S, Burke S, Ehm F, Field L, Galang G, Konya B, Litmaath M, Millar P, Navarro J (2009) GLUE Specification v. 2.0
5. Ballani H, Costa P, Karagiannis T, Rowstron A (2011) Towards predictable datacenter networks. SIGCOMM Comput Commun Rev 41(4):242–253
6. Barabash K, Cohen R, Hadas D, Jain V, Recio R, Rochwerger B (2011) A case for overlays in DCN virtualization. In: Proceedings of the 3rd workshop on data center—converged and virtual ethernet switching, DC-CaVES'11, ITCP, pp 30–37
7. Barham P, Dragovic B, Fraser K, Hand S, Harris T, Ho A, Neugebauer R, Pratt I, Warfield A (2003) Xen and the art of virtualization. SIGOPS Oper Syst Rev 37(5):164–177
8. Barker SK, Shenoy P (2010) Empirical evaluation of latency-sensitive application performance in the cloud. In: Proceedings of the first annual ACM SIGMM conference on multimedia systems, MMSys'10. ACM, New York, NY, USA, pp 35–46
9. Bates A, Mood B, Pletcher J, Pruse H, Valafar M, Butler K (2012) Detecting co-residency with active traffic analysis techniques. In: Proceedings of the 2012 ACM workshop on cloud computing security workshop, CCSW'12. ACM, New York, NY, USA, pp 1–12
10. Bhadauria M, McKee SA (2010) An approach to resource-aware co-scheduling for CMPs. In: Proceedings of the 24th ACM international conference on supercomputing, ICS'10. ACM, New York, NY, USA, pp 189–199
11. Biran O, Corradi A, Fanelli M, Foschini L, Nus A, Raz D, Silvera E (2012) A stable network-aware VM placement for cloud systems. In: Proceedings of the 2012 12th IEEE/ACM international symposium on cluster, cloud and grid computing (ccgrid 2012), CCGRID'12. IEEE Computer Society, Washington, DC, USA, pp 498–506
12. Bleikertz S, Groß T (2011) A virtualization assurance language for isolation and deployment. IEEE Policy, VALID
13. Brandtzaeg E, Mohagheghi P, Mosser S (2012) Towards a domain-specific language to deploy applications in the clouds. In: CLOUD COMPUTING 2012, the third international conference on cloud computing, GRIDs, and virtualization, pp 213–218
14. Breitgand D, Epstein A (2012) Improving consolidation of virtual machines with risk-aware bandwidth oversubscription in compute clouds. In: Proceedings on IEEE INFOCOM, pp 2861–2865
15. Broquedis F, Clet-Ortega J, Moreaud S, Furmento N, Goglin B, Mercier G, Thibault S, Namyst R (2010) hwloc: a generic framework for managing hardware affinities in HPC applications. In: 18th Euromicro international conference on parallel, distributed and network-based processing (PDP), pp 180–186
16. Cappello F, Caron E, Dayde M, Desprez F, Jegou Y, Primet P, Jeannot E, Lanteri S, Leduc J, Melab N, Mornet G, Namyst R, Quetier B, Richard O (2005) Grid'5000: a large scale and highly

reconfigurable grid experimental testbed. In: Proceedings of the 6th IEEE/ACM international workshop on grid computing, GRID'05. IEEE Computer Society, Washington, DC, USA, pp 99–106
17. Clemente P, Rouzaud-Cornabas J, Toinard C (2010) From a generic framework for expressing integrity properties to a dynamic mac enforcement for operating Ssystems. In: Gavrilova M, Tan C, Moreno E (eds) Transactions on computational science XI, vol 6480 of Lecture Notes in Computer Science. Springer Berlin, pp 131–161
18. Fan P, Chen Z, Wang J, Zheng Z, Lyu MR (2012) Topology-aware deployment of scientific applications in cloud computing. In: 2012 IEEE fifth international conference on cloud computing, pp 319–326
19. Fedorova A, Seltzer M, Smith MD (2007) Improving performance isolation on chip multiprocessors via an operating system scheduler. In: Proceedings of the 16th international conference on parallel architecture and compilation techniques, PACT'07. IEEE Computer Society, Washington, DC, USA, pp 25–38
20. Feller E, Rilling L, Morin C (2011) Energy-aware ant colony based workload placement in clouds. In: The 12th IEEE/ACM international conference on grid computing (GRID-2011), Lyon, France
21. Feller E, Rilling L, Morin C, Lottiaux R, Leprince D (2010) Snooze: a scalable, fault-tolerant and distributed consolidation manager for large-scale clusters. In: Green computing and communications (GreenCom), 2010 IEEE/ACM Int'l conference on Iint'l conference on cyber, physical and social computing (CPSCom), pp 125–132
22. Galán F, Sampaio A, Rodero-Merino L, Loy I, Gil V, Vaquero LM (2009) Service specification in cloud environments based on extensions to open standards. In: Proceedings of the fourth international ICST Conference on cOMmunication System softWAre and middlewaRE, COMSWARE'09, vol 19. ACM, New York, NY, USA, pp 1–19, 12
23. Ganguly A, Agrawal A, Boykin P, Figueiredo R (2006) IP over P2P: enabling self-configuring virtual IP networks for grid computing. In: 20th International conferece on parallel and distributed processing symposium, IPDPS 2006, p 10
24. Goglin B, Moreaud S, (2011). Dodging non-uniform I/O access in hierarchical collective operations for multicore clusters. In: CASS the 1st workshop on communication architecture for scalable systems, held in conjunction with IPDPS 2011. IEEE Computer Society Press, Anchorage, AK
25. Gonçalves G, Endo P, Santos M, Sadok D, Kelner J, Melander B, Mangs J (2011) CloudML: an integrated language for resource, service and request description for D-clouds. In: 2011 IEEE third international conference on cloud computing technology and science (CloudCom). IEEE, pp 399–406
26. Greenberg A, Hamilton JR, Jain N, Kandula S, Kim C, Lahiri P, Maltz DA, Patel P, Sengupta S (2009) VL2: a scalable and flexible data center network. SIGCOMM Comput Commun Rev 39(4):51–62
27. Gude N, Koponen T, Pettit J, Pfaff B, Casado M, McKeown N, Shenker S (2008) NOX: towards an operating system for networks. SIGCOMM Comput Commun Rev 38(3):105–110
28. Gulati A, Merchant A, Varman PJ (2010) mClock: handling throughput variability for hypervisor IO scheduling. In: Proceedings of the 9th USENIX conference on operating systems design and implementation, OSDI'10. USENIX Association Berkeley, CA, USA, pp 1–7
29. Harnik D, Pinkas B, Shulman-Peleg A (2010) Side channels in cloud services: deduplication in cloud storage. IEEE Secur Priv 8(6):40–47
30. Hayashi Y, Itsumi H, Yamamoto M (2011) Improving fairness of quantized congestion notification for data center ethernet networks. In: Proceedings of the 2011 31st international conference on distributed computing systems workshops, ICDCSW'11. IEEE Computer Society, Washington, DC, USA, pp 20–25
31. Hicks B, Rueda S, King D, Moyer T, Schiffman J, Sreenivasan Y, McDaniel P, Jaeger T (2010) An architecture for enforcing end-to-end access control over web applications. In: Proceedings of the 15th ACM symposium on access control models and technologies, SACMAT'10. ACM, New York, NY, USA, pp 163–172

32. Huber N, von Quast M, Hauck M, Kounev S (2011) Evaluating and modeling virtualization performance overhead for cloud environments. In: Proceedings of the 1st international conference on cloud Ccomputing and services science (CLOSER 2011), Noordwijkerhout, The Netherlands, 7–9 May. SciTePress, pp 563–573. Acceptance Rate: 18/164 = 10.9 %, Best Paper Award
33. Jayasinghe D, Pu C, Eilam T, Steinder M, Whally I, Snible E (2011) Improving performance and availability of services hosted on IaaS clouds with structural constraint-aware virtual machine placement. In: IEEE international conference on services computing (SCC), pp 72–79
34. Jiang X, Xu D (2004) VIOLIN: virtual internetworking on overlay infrastructure. In: Proceedings of the second international conference on parallel and distributed processing and applications, ISPA'04. Springer, Berlin, pp 937–946
35. Keller E, Szefer J, Rexford J, Lee RB (2010) NoHype: virtualized cloud infrastructure without the virtualization. SIGARCH Comput Archit News 38(3):350–361
36. Kim G, Park H, Yu J, Lee W (2012) Virtual machines placement for network isolation in clouds. In: Proceedings of the 2012 ACM research in applied computation symposium, RACS'12, New York, NY, USA, pp 243–248
37. Kortchinsky K (2009) Hacking 3D (and Breaking out of VMWare). BlackHat USA
38. Lacour S, Perez C, Priol T (2004) A network topology description model for grid application deployment. In: Proceedings of the 5th IEEE/ACM international workshop on grid computing, GRID '04. IEEE Computer Society, Washington, DC, USA, pp 61–68
39. Landau A, Hadas D, Ben-Yehuda M (2010) Plugging the hypervisor abstraction leaks caused by virtual networking. In: Proceedings of the 3rd annual Haifa experimental systems conference, SYSTOR'10. ACM, New York, NY, USA, pp 16:1–16:9
40. Li J, Qiu M, Niu J, Gao W, Zong Z, Qin X (2010) Feedback dynamic algorithms for preemptable job scheduling in cloud systems. In: Proceedings of the 2010 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology, vol 01, WI-IAT'10. IEEE Computer Society, Washington, DC, USA, pp 561–564
41. Macdonell C, Lu P (2007) Pragmatics of virtual machines for high-performance computing: a quantitative study of basic overheads. In: High performance computing and simulation conference
42. Marshall A, Howard M, Bugher G, Harden B, Kaufman C, Rues M, Bertocci V (2010) Security best practices for developing windows azure applications. Microsoft Corp
43. McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J (2008) OpenFlow: enabling innovation in campus networks. SIGCOMM Comput Commun Rev 38(2):69–74
44. Meng X, Pappas V, Zhang L (2010) Improving the scalability of data center networks with traffic-aware virtual machine placement. In: Proceedings of the 29th conference on information communications, INFOCOM'10. IEEE Press, Piscataway, NJ, USA, pp 1154–1162
45. Merkel A, Stoess J, Bellosa F (2010) Resource-conscious scheduling for energy efficiency on multicore processors. In: Proceedings of the 5th European conference on computer systems, EuroSys'10. ACM, New York, NY, USA, pp 153–166
46. Mills K, Filliben J, Dabrowski C (2011) Comparing VM-placement algorithms for on-demand clouds. In: Proceedings of the 2011 IEEE conference cloudCom
47. Mirkovic J, Faber T, Hsieh P, Malaiyandisamy G, Malaviya R (2010) DADL: distributed application description language. USC/ISI Technical Report# ISI-TR-664
48. Moscibroda T, Mutlu O (2007) Memory performance attacks: Denial of memory service in multi-core systems. In: Proceedings of 16th USENIX security symposium on USENIX security symposium, SS'07. USENIX Association, Berkeley, CA, USA, pp 18:1–18:18
49. Murakami J (2008) A hypervisor IPS based on hardware assisted virtualization technology. Black Hat USA
50. Nathani A, Chaudhary S, Somani G (2012) Policy based resource allocation in IaaS cloud. Future Gener Comput Sys 28(1):94–103
51. Nguyen Van H, Dang Tran F, Menaud J-M (2009) Autonomic virtual resource management for service hosting platforms. In: Proceedings of the 2009 ICSE workshop on software engineering

challenges of cloud computing, CLOUD '09. IEEE Computer Society, Washington, DC, USA, pp 1–8
52. Okamura K, Oyama Y (2010) Load-based covert channels between xen virtual machines. In: Proceedings of the 2010 ACM symposium on applied computing, SAC'10, ACM, New York, NY, USA, pp 173–180
53. Onoue K, Matsuoka N, Tanaka J (2012) Host-based multi-tenant technology for scalable data center networks. In: Proceedings of the eighth ACM/IEEE symposium on Architectures for networking and communications systems, ANCS'12. ACM, New York, NY, USA. pp 87–98
54. Osvik DA, Shamir A, Tromer E (2006) Cache attacks and countermeasures: the case of AES. In: Proceedings of the 2006 the cryptographers' track at the RSA conference on topics in cryptology, CT-RSA'06. Springer, Berlin, pp 1–20
55. Open Virtualization Format Specification. Version: 1.0.0d. Distributed Management Task Force, Inc. (DMTF).
56. Page D (2005) Partitioned cache architecture as a side-channel defence mechanism. In: Technical report 2005/280, IACR eprint archive. Cryptography ePrint archive
57. Percival C (2005) Cache missing for fun and profit, BSDCan
58. Pu X, Liu L, Mei Y, Sivathanu S, Koh Y, Pu C (2010) Understanding performance interference of I/O workload in virtualized cloud environments. In: 2010 IEEE 3rd international conference on cloud computing (CLOUD), pp 51–58
59. Raj H, Nathuji R, Singh A (2009) Resource management for isolation enhanced cloud services. In: CCSW'09 proceedings of the 2009 ACM workshop on cloud computing security, p 77
60. Ristenpart T, Tromer E, Shacham H, Savage S (2009) Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proceedings of the 16th ACM conference on computer and communications security, CCS'09, New York, NY, USA, pp 199–212
61. Rodero-Merino L, Vaquero LM, Caron E, Muresan A, Desprez F (2012) Building safe PaaS clouds: A survey on security in multitenant software platforms. Comput Secur 31(1):96–108
62. Schad J, Dittrich J, Quiané-Ruiz J-A (2010) Runtime measurements in the cloud: observing, analyzing, and reducing variance. Proc VLDB Endow 3(1–2):460–471
63. Shieh A, Kandula S, Greenberg A, Kim C (2010) Seawall: performance isolation for cloud datacenter networks. In: Proceedings of the 2nd USENIX conference on hot topics in cloud computing, HotCloud'10, Berkeley, CA, USA, p 1
64. Simarro JLL, Moreno-Vozmediano R, Montero RS, Llorente IM (2011) Dynamic placement of virtual machines for cost optimization in multi-cloud environments. In: International conference on high performance computing and simulation (HPCS), pp 1–7
65. Sotomayor B, Keahey K, Foster I (2008) Combining batch execution and leasing using virtual machines. In: Proceedings of the 17th international symposium on high performance distributed computing, HPDC'08, New York, NY, USA, pp 87–96
66. Sotomayor B, Montero RS, Llorente IM, Foster I (2009) Virtual infrastructure management in private and hybrid clouds. IEEE Intern Comput 13(5):14–22
67. Srikantaiah S, Kansal A, Zhao F (2008) Energy aware consolidation for cloud computing. In: Proceedings of the 2008 conference on power aware computing and systems, HotPower'08. USENIX Association, Berkeley, CA, USA, p 10
68. Stabler G, Rosen A, Goasguen S, Wang K-C (2012) Elastic IP and security groups implementation using openFlow. In: Proceedings of the 6th international workshop on virtualization technologies in distributed computing date, VTDC'12. ACM, New York, NY, USA, pp 53–60
69. Stillwell M, Schanzenbach D, Vivien F, Casanova H (2009) Resource allocation using virtual clusters. In: Proceedings of the 2009 9th IEEE/ACM international symposium on cluster computing and the grid, CCGRID '09. IEEE Computer Society, Washington, DC, USA, pp 260–267
70. Sundararaj AI, Dinda PA (2004) Towards virtual networks for virtual machine grid computing. In: Proceedings of the 3rd conference on virtual machine research and technology symposium, vol 3, VM'04. USENIX Association, Berkeley, CA, USA, pp 14–14
71. Szefer J, Keller E, Lee R (2011) Eliminating the hypervisor attack surface for a more secure cloud. In: ACM conference on computer and communications security

72. Taesoo K, Peinado M, Mainar-Ruiz G (2012) System-level protection against cache-based side channel attacks in the cloud. In: Proceedings of the 21st Usenix Security symposium, USENIX Security'12. USENIX Association, Berkeley, CA, USA, pp 1–16
73. Tickoo O, Iyer R, Illikkal R, Newell D (2010) Modeling virtual machine performance: challenges and approaches. SIGMETRICS Perform Eval Rev 37(3):55–60
74. Tordsson J, Montero RS, Moreno-Vozmediano R, Llorente IM (2012) Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. Futur Gener Comput Sys 28(2):358–367
75. Tsugawa M, Fortes JAB (2006) A virtual network (ViNe) architecture for grid computing. In: Proceedings of the 20th international conference on Parallel and distributed processing, IPDPS'06. IEEE Computer Society, Washington, DC, USA, pp 148–148
76. Varadarajan V, Kooburat T, Farley B, Ristenpart T, Swift MM (2012) Resource-freeing attacks: improve your cloud performance (at your neighbor's expense). In: Proceedings of the 2012 ACM conference on computer and communications security, CCS'12. ACM, New York, NY, USA, pp 281–292
77. Verghese B, Gupta A, Rosenblum M (1998) Performance isolation: sharing and isolation in shared-memory multiprocessors. SIGOPS Oper Syst Rev 32(5):181–192
78. Wang G, Ng T (2010) The impact of virtualization on network performance of amazon EC2 data center. In 2010 Proceedings IEEE INFOCOM, pp 1–9
79. Wang M, Meng X, Zhang L (2011) Consolidating virtual machines with dynamic bandwidth demand in data centers. In: 2011 proceedings on IEEE INFOCOM, pp 71–75
80. Wojtczuk R (2008) Subverting the Xen hypervisor. Black Hat USA
81. Wu Z, Xu Z, Wang H (2012) Whispers in the hyper-space: high-speed covert channel attacks in the cloud. In: the 21st USENIX security symposium (Security'12)
82. Xia L, Cui Z, Lange JR, Tang Y, Dinda PA, Bridges PG (2012) VNET/P: Bridging the cloud and high performance computing through fast overlay networking. In: Proceedings of the 21st international symposium on high-performance parallel and distributed computing, HPDC'12. ACM, New York, NY, USA, pp 259–270
83. Xu Y, Bailey M, Jahanian F, Joshi K, Hiltunen M, Schlichting R (2011) An exploration of L2 cache covert channels in virtualized environments. In: Proceedings of the 3rd ACM workshop on cloud computing security workshop, CCSW'11. ACM, New York, NY, USA, pp 29–40
84. Zhang Y, Juels A, Oprea A, Reiter M (2011) HomeAlone: Co-residency Detection in the Cloud via Side-Channel Analysis. In: IEEE symposium on security and privacy (SP), pp 313–328
85. Zhang Y, Juels A, Reiter MK, Ristenpart T (2012) Cross-VM side channels and their use to extract private keys. In: Proceedings of the 2012 ACM conference on computer and communications security, CCS'12. ACM, New York, NY, USA, pp 305–316
86. Zhuravlev S, Blagodurov S, Fedorova A (2010) Addressing shared resource contention in multicore processors via scheduling. SIGARCH Comput Archit News 38(1):129–142

Mandatory Access Protection Within Cloud Systems

M. Blanc, A. Bousquet, J. Briffaut, L. Clevy, D. Gros, A. Lefray, J. Rouzaud-Cornabas, C. Toinard and B. Venelle

1 Introduction

In order to guarantee security properties, such as confidentiality and integrity, cryptographic mechanisms provide encryption and signature of data, but protection is required to control the data accesses. The recent attacks on Facebook and

A. Bousquet · J. Briffaut · C. Toinard
ENSI-LIFO,88 bld Lahitolle, 18020 Bourges, France
e-mail: aline.bousquet@ensi-bourges.fr

J. Briffaut
e-mail: jeremy.briffaut@ensi-bourges.fr

C. Toinard (✉)
e-mail: christian.toinard@ensi-bourges.fr

M. Blanc · D. Gros
CEA, DAM, DIF,91297 Arpajon, France
e-mail: Mathieu.Blanc@cea.fr

D. Gros
e-mail: Damien.Gros@cea.fr

L. Clevy · B. Venelle
Alcatel-Lucent Bell Labs,7 route de Villejust, 91620 Nozay, France
e-mail: laurent.clevy@alcatel-lucent.com

B. Venelle
e-mail: benjamin.venelle@alcatel-lucent.com

A. Lefray · J. Rouzaud-Cornabas
ENS Lyon-LIP-Avalon,46 Allee d'Italie, 69364 Lyon Cedex 07, France
e-mail: arnaud.Lefray@ens-lyon.fr

J. Rouzaud-Cornabas
e-mail: Jonathan.Rouzaud-Cornabas@ens-lyon.fr

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_5,

Twitter[1] show that the protection must not be limited to the infrastructure i.e. the hosts and the guest virtual machines. Indeed, those attacks rely on a poor control of the information flows within the application i.e. the flows between the Java objects sharing the same Java Virtual Machine. Indeed, Java privilege escalations are achieved through intermediate objects leading to indirect flows within the JVM that violate the confidentiality or the integrity of the system.

Discretionary Access Control (DAC) is fragile since the end users have the responsibility to define the permissions on their own resources. In contrast with the DAC approach, the Mandatory Access Control (MAC) can guarantee security properties. A MAC policy is outside the scope of the end-users. Thus, even the root privilege does not permit to compromise the security. In practice, a reference monitor enforces a MAC policy by capturing the interactions, e.g. the system or method calls, and deciding if they satisfy the policy. However, the MAC approaches are currently poorly supported for the IaaS, the PaaS and the SaaS Cloud systems. Indeed, several hard problems face the security officer aiming at using MAC approaches. First, the security officer has to use different languages for defining the MAC policies at the different levels. Second, the majority of existing MAC solutions poorly control the indirect flows and do not support an advanced property mixing several direct and indirect flows. Third, the MAC solutions are complex to manage since generally millions of rules are required to control the flows at the different levels of the Cloud environments. Finally, the MAC solutions poorly support the heterogeneity of systems e.g. MS Windows and Linux operating systems.

This chapter presents the PIGA-Cloud solution facing the different problems of the MAC approach. PIGA-Cloud guaranties advanced properties associated with multiple direct and indirect flows. An in-depth MAC protection controls the flows between the guest virtual machines and the host, the internal flows of a guest but also the flows between Java objects and the network flows. This chapter shows how to integrate PIGA in various environments such as Unix or MS Windows machines, Java applications and Clouds. PIGA-Cloud extends the direct access MAC policies of SELinux [1] and sVirt [2] to the Java Virtual Machine and to MS Windows providing thus an advanced protection of IaaS, PaaS and SaaS Clouds. Our approach simplifies the administration of the direct access policies while preventing against millions of remaining vulnerabilities.

In contrast with PIGA-OS and PIGA-Virt [3] that only consider the operating system level, PIGA-Cloud goes further since it supports dynamic management of virtual machines, monitoring of the information flows within the applications and heterogeneity of the system. Thus, PIGA-Cloud enforces the security of both IaaS Clouds using virtual operating systems and PaaS or SaaS Clouds, including the application level. PIGA-Cloud is the first extensible in-depth end-to-end MAC protection that can be used for securing various environments. That chapter demonstrates that it is possible to integrate PIGA in every Node/Virtual Machine/Java Application of a Cloud as well as in every Linux or MS Windows Client. Obviously, the PIGA

[1] http://www.forbes.com/sites/andygreenberg/2013/02/15/facebook-hacked-via-java-vulnerability-claims-no-user-data-compromised/

approach can be easily extended to other components such as a PHP virtual machine, .NET framework and hardware.

The PIGA approach is integrated in the OpenNebula [4] infrastructure but it supports any Cloud infrastructure since it is independent of the virtualization mechanisms. First, the chapter explains how PIGA-Cloud deploys the VMs and supports their migration through consistent security policies controlling the direct flows. PIGA-Cloud extends the notion of direct MAC policies available in SELinux to the Java and MS Windows systems. First, Security Enhanced Java (SEJava) is able to monitor the direct flows between Java objects. Second, Security Enhanced Windows (SEWindows) controls the direct flows of a MS Windows Operating Systems. Thanks to the PIGA-Shared reference monitor, PIGA-Cloud can guarantee a large range of security properties inside and between the guests (Linux or MS Windows guests), but also between the objects of a Java application. Thus, an end-to-end protection is provided from a MS Windows/Linux Client to (1) IaaS heterogeneous Clouds running Linux and MS Windows VMs and (2) PaaS and SaaS Clouds including Java applications.

PIGA-Cloud simplifies the work of the security officer. Indeed, it provides templates of rules answering a large range of security requirements. The security officer easily creates a PIGA policy by instantiating those templates with the security contexts defined in the existing direct access control policies (SELinux, sVirt, SEJava, SEWindows…). Thus, only a couple of PIGA rules monitors millions of potential vulnerabilities in IaaS, PaaS and SaaS Clouds.

The chapter is organized as follows. The Sect. 2 describes the state of the art related to the protection of Cloud environments. Section 3 presents the global architecture of PIGA-Cloud including the proposed model contrasting the direct and advanced activities. The Sect. 4 describes the three levels of protection controlling the direct activities inside an operating system, between the virtual machines and inside the Java applications. Section 5 presents the PIGA approach with a use-case that addresses the administration of the Cloud through a Web service and proposes advanced protections again covering the operating systems at the nodes and the client sides, virtual machines and Java applications. Section 6 explains the lessons learned and defines the future works. Section 7 concludes the chapter and summarizes the advantages of PIGA-Cloud for controlling advanced activities through a unified and extensible approach.

2 State of the Art and Motivation

Security is a major issue for Clouds [5–8]. Indeed, the security perimeter becomes blurred between Intranet and Internet. In addition, the attack surface increases due to virtualization as shown by the results of the ANR security challenge [9]. The authors of [10] emphasize the importance of controlling access in public Clouds. This control should be mandatory [11] since the discretionary access control under the responsibility of the end user is fragile and can not guarantee confidentiality

or integrity properties. Mandatory control is achieved by a reference monitor [12] placing protection policies away from users and administrators of the machines. In practice, the policy must be defined by the operator of the Cloud infrastructure. However, the difficulty is to address the complexity of the different levels of policy that are required to support an in-depth protection that controls the information flows within operating systems, between the virtual machines and within the applications. For example, [13] proposes an architecture to provide a mandatory access control for Web applications. They use policies for SELinux and Xen Security Modules to enforce a mandatory access control inside and between Virtual Machines. So, multiple MAC policies are requested using millions of rules. However, they can not express advanced security properties including the control of multiple direct and indirect activities. Finally, they do not control the flows within the Web application.

Thus, a dedicated language is required to express advanced security properties dealing with confidentiality and integrity. That language must be extensible and be able to cover three different levels of systems for controlling the flows within the operating systems and the applications but also between the virtual machines. That chapter reuses the PIGA language [3] to support advanced controls of the information flows for (1) the operating systems of both the nodes of the Clouds and the Clients, (2) the different guest Virtual Machines sharing the same node and (3) the objects running inside an application such as a Java application. That common language addresses the problem of complexity of the different MAC policies since it enforces through a couple of rules the prevention of millions of malicious activities. Moreover, the protection supports the heterogeneity of the different levels. For example, the same properties can support heteregeneous operating systems such as e.g. MS Windows and GNU/Linux.

3 PIGA-Cloud Description

3.1 Global Architecture

PIGA-Cloud offers a mandatory protection through three protection levels:

- Operating systems: protection of the resources of an operating system, either the host's or the Virtual Machine's (VM) resources
- Virtual Machines: protection of a host against the VMs and of the VMs against one another
- Java applications: protection inside a Java Virtual Machine (JVM)

PIGA-Cloud architecture, presented in Fig. 1, uses *security contexts* to identify the various entities/resources (process, file, virtual machine, Java object…) and a shared reference monitor (PIGA-Shared) to control the operations between these security contexts (SELinux, sVirt, SEJava, SEWindows).

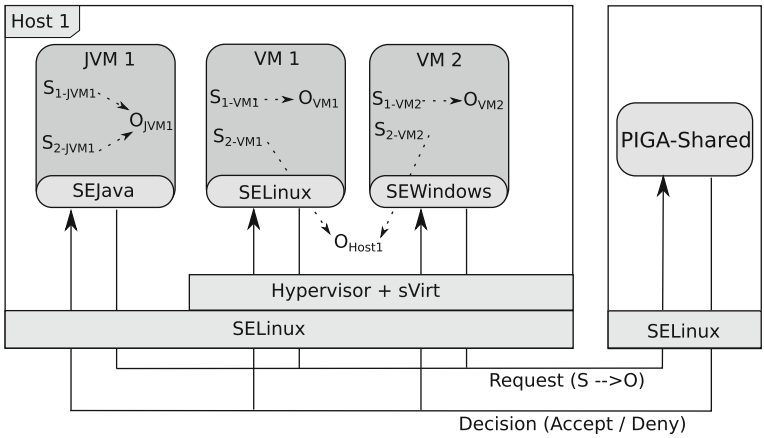


Fig. 1 PIGA-Cloud

The objective of SELinux, sVirt, SEJava and SEWindows is to control a direct flow corresponding to a source security context accessing a target security context $sc_s \rightarrow sc_t$. Let us give an informal definition of a direct flow: if a source sc_s writes the target sc_t, then there is a direct flow $sc_s > sc_t$. If a source sc_s reads a target sc_t, then there is a direct flow $sc_s < sc_t$.

Let us now define informally an indirect flow. If there is a causal closure of two direct flows $sc_{s1} > sc_t$ and $sc_t > sc_{s2}$, i.e. there is a shared context sc_t and the first flow $sc_{s1} > sc_t$ starts before the second flow $sc_t > sc_{s2}$ ends, then there is an indirect flow $sc_{s1} >> sc_{s2}$, see [14] for more details. PIGA controls indirect flows $sc_{s1} >> sc_{s2}$ but also guarantees advanced properties associated with several direct and indirect flows. Thanks to such advanced properties, PIGA significantly eases the policy administration and helps to improve the security. However, PIGA requires (1) security contexts associated to the controlled entities and (2) direct access control policies between these contexts.

SELinux can protect an operating system, either the one of the host or of a guest virtual machine. SELinux uses direct mandatory policies that minimize the privileges of the processes regarding the system resources. The PIGA-Shared reference monitor adds advanced properties to SELinux.

SELinux/sVirt can secure VMs with direct access policies, while being independent of the hypervisor (KVM/Qemu, Xen, VMWare…). In order to cope with virtual machines' migrations, PIGA-Cloud associates a unique security context to each virtual machine and ensures that the contexts are consistent within the Cloud infrastructure. However, sVirt controls only direct information flows between a VM and the host. Thus, PIGA-Shared adds advanced security properties to sVirt. Later sections show how a PIGA property can prevent millions of illicit activities allowed by SELinux/sVirt.

SEJava aims to protect Java applications running in a Cloud environment. SEJava adds the concept of security contexts to Java objects and controls the flows inside the JVM through a direct policy. Nonetheless, just like SELinux, SEJava cannot prevent indirect flows. PIGA-Shared removes this limitation and offers an advanced protection of the Java objects running inside a JVM.

SEWindows controls the direct flows between the processes and the resources available in a Microsoft Windows Operating System. Again, SEWindows adds the concept of security contexts to the MS Windows entities and controls the direct flows inside a MS Windows Operating System. SEWindows, associated with PIGA rules, guarantees an advanced protection of a MS Windows OS.

PIGA-Shared guarantees the advanced security properties for each kind of security context and direct policy (SELinux, sVirt, SEJava, SEWindows). PIGA-Shared receives the requests for the interactions $sc_s \rightarrow sc_t$ coming from each direct reference monitor (SELinux, sVirt, SEJava, SEWindows) and sends back decisions (allow/deny) depending on the PIGA properties. Thus, PIGA-Shared adds advanced controls to any existing direct reference monitor.

3.2 Model

Mandatory Security Contexts

A mandatory security context sc is an association between a label and a resource/entity. Depending on the kind of protection, a mandatory context can represent:

- a virtual machine: a KVM process and its disk images
- a resource on the system: a process or a file
- a resource in the JVM: a Java object

PIGA-Cloud uses the notion of mandatory security context, i.e. a security reference designating an entity. Any kind of notion of MAC security contexts fits. For example, the SELinux or GRsecuritry contexts are supported and several labeling methods are proposed for MS Windows and Java based systems showing the extensibility of the approach. A security context is the label of a resource/entity including a set of attributes. Thanks to this notation, the MAC rules are independent of the name/placement/path of the resources/entities. Theses MAC rules can control a direct or advanced activity.

Direct Activity

A direct activity $sc_s \rightarrow sc_t$ is an operation done by the source security context sc_s on the target context sc_t. An operation can be either a system call (read, write, execute…) or an action inside the JVM (method invocation, field access). A direct

activity is a flow $sc_s > sc_t$ or $sc_s < sc_t$ whether the activity is a write-like or read-like operation.

For instance, the reading of Firefox configuration (`firefox_config_t`) file by the Firefox application (`firefox_app_t`) will be represented as follows :

```
# direct activity
firefox_app_t  -{read}-> firefox_config_t
# read-like flow
firefox_app_t  < firefox_config_t
```

Advanced Activity

An advanced activity is a combination of direct activities. It can be a sequence $sc_1 \Rightarrow sc_n$ that is a transitive sequence of interactions $sc_1 \rightarrow sc_2; \ldots; sc_{n-1} \rightarrow sc_n$. Such a sequence corresponds with an indirect information flow $sc_1 >> sc_n$ when there is a transitive closure of operations causally linked [14] i.e. $sc_1 > sc_2 > \ldots > sc_{n-1} > sc_n$. In practice, an indirect flow corresponds to a sequence of operations leading to information sharing or leaking. For instance, a virtual machine process sc_{s-vm1} can share data with the process of another virtual machine sc_{s-vm2} by using a shared resource $sc_{s-host1}$.

Let us give an example of an advanced activity. The firefox application can first read the Firefox configuration file and then write its content to a web socket. Thus, there is an indirect flow from the Firefox configuration to a web socket when those two direct flows are causally related.

```
# Advanced activity
firefox_app_t  -{read}->firefox_config_t ;
   firefox_app_t -{write}-> web_socket_t
# Indirect information flow associated with two
   causally related direct flows
firefox_config_t > firefox_app_t > web_socket_t
firefox_config_t >> web_socket_t
```

4 Direct Mandatory Protection

This section describes the mechanisms controlling the direct activities for the three considered protection levels. Each level defines (1) a notion of mandatory security context and (2) a model controlling a direct MAC policy. The advanced mandatory protection model, PIGA, is described in Sect. 5.

4.1 Protection Inside an Operating System

Protecting resources, inside an operating system, controls the direct activities occurring on both the host and the guest virtual machine. Fewer works deal with the integration of a MAC protection within the kernel of an operating system. For the Linux kernel, several propositions are available. GRsecurity is a kernel patch that provides a MAC protection. One of the most mature is the SELinux operating system [15]. That solution provides Type Enforcement [16] available at the scale of the whole operating system.

In the context of MS Windows systems, few works deal with the MAC enforcement. First of all, Core Force [17] provides the first implementation of a MAC for the MS Windows XP kernel. MS Windows Vista and MS Windows 7 kernels associate each resource with an integrity level. However, MS Windows 7 kernel does not support Type Enforcement. MS Windows *Mandatory Integrity Control* (MIC) available on Windows 7 is a limited style of MAC protection. PIGA-Cloud proposes SEWindows [18], a MAC mechanism for the MS Windows operating system, with Type Enforcement.

With SELinux or SEWindows, a security context/label identifies each process/file. A SELinux/SEWindows policy defines all the allowed direct interactions, where an interaction is an operation (a system call) between two contexts. The kernel captures the interactions and guarantees that the policy is satisfied. Thus, if a process is compromised, the attacker will not obtain more permissions and the impact of the attack is therefore limited.

SELinux

Security Contexts

A security context is a label including three attributes: a user, a role, and a type. With SELinux, each user is associated to a SELinux user associated with several roles. Each role grants the access to a set of SELinux types. The access control rules define the set of operations (system calls) that are allowed between the SELinux types. The following listing shows some associations between process/files and their contexts:

```
/etc/shadow: system_u:object_r:shadow_t
/usr/bin/apache: system_u:object_r:apache_exec_t
apache process: system_u:system_r:httpd_t
```

In this example, the system services or resources have the `system_u` identity. Passive resources have the `object_r` role. Processes providing system service have the `system_r` role. The type of the file including hashed passwords is `shadow_t`, apache binaries are `apache_exec_t` and web server processes are `httpd_t`.

Direct mandatory protection

A SELinux access control policy defines all the allowed direct activities of an operating system (for instance, the operating system of a VM) resulting in millions of rules. SELinux enforces the *least privilege principle* for each process.

This listing shows a extract of a SELinux policy:

```
#users definition and roles associations
user root roles { sysadm_r staff_r user_r }
user system_u roles { system_r }
#types definition
type passwd_t;
type httpd_t;
[...]
#association between roles and types
role sysadm_r types passwd_t;
role system_r types httpd_t;
[...]
#definition of rules between types
allow passwd_t shadow_t:file { open getattr read }
allow httpd_t apache_exec_t:file { open getattr read
    execute }
[...]
```

This policy allows several direct operations. For example, a process executing a `passwd_t` binary can read the `/etc/shadow_t` file. A `httpd_t` process can execute an `apache_exec_t` binary. However, a compromised web server becoming `root` cannot read `/etc/shadow` since it does not have the `passwd_t` type.

SEWindows

This section describes the principle of SEWindows. A driver captures the system calls in order to control the direct flows between a source and a target. Each entity of the system (process, file, pipe) is associated with a security context. In contrast with SELinux, SEWindows computes dynamically the security contexts in a portable manner. For example, the security context for the `C:\Windows\System32\cmd.exe` file is `system_u:object_r:systemroot|system32|cmd_exec_t`.

Our driver captures the system calls by hooking the *System Service Dispatch Table* (SSDT) containing the references for the system calls. The driver verifies that the system calls satisfy a direct policy. The driver denies a system call that does not satisfy that direct policy. Otherwise, the driver allows the access and sends a request to the PIGA-Shared reference monitor see Sect. 5 before allowing the execution of the corresponding system call.

Further details on SEWindows can be found in [18].

4.2 Protection of Virtual Machines

A virtual machine running on an hypervisor includes a process (a Qemu/KVM instance) and a set of files (configuration files and virtual disks). Most of the research

on security for virtualization [19, 20] focus on hardening the hypervisor but do not provide access control mechanisms between VMs. [21] uses a dedicated hypervisor to encrypt the data and the network transmission. GuardHype [22] and [23] verifies the integrity of the hypervisor itself or the integrity of the kernel and of the critical applications. sHype [24] brings Type Enforcement to control the inter-VM communications. In [25], the authors propose to improve the isolation of virtual machines while minimizing the loss of consolidation. The approach [25] offers a better isolation. To do this, they use memory page coloring. But their approach introduces a very large overhead that does not fit with real world cloud platforms.

In contrast, our approach consists in reusing and extending the MAC sVirt solution in order to isolate the virtual machines from the host system. sVirt is an extension for SELinux that uses categories to monitor direct interactions between virtual machines, so that:

- the virtual machine is isolated from the host system: if a VM is attacked, only a restricted access to the system is granted
- virtual machines are isolated from one another: a VM cannot interact with the files/processes of the other VMs.

The main issue with sVirt is that it doesn't handle the migrations between several hypervisors. Thus, two different virtual machines can have the same category and therefore malicious interactions are allowed.

PIGA-Cloud offers a designation service that provides unique and consistent security contexts for the VMs. Therefore, a VM can migrate from a source host H1 to a target host H2, while keeping its security context.

Security Contexts

PIGA-Cloud uses the static labeling service provided by sVirt to assign a unique context to the process executing the image and to the files holding the image's file system. With the current implementation of sVirt, a context consists of a SELinux label (`user:role:type`) and a category ($c_x : c_y$). The following listing shows some of the SELinux/sVirt labels:

```
# Subjects
system_u:system_r:svirt_t: KVM process
system_u:system_r:virtd_t: libvirtd daemon
# Objects
system_u:object_r:svirt_image_t: read/write image file
system_u:object_r:virt_image_t: read only image file
    (VM not running)
system_u:object_r:virt_content_t: read only image file
    (shared disks)
system_u:object_r:svirt_etc_t: XML configuration files
```

On the other hand, categories can isolate virtual machines from each others: a source process can interact with a target entity only if the target's categories are a

subset of the source's categories. For instance, a process labeled with c_0, c_5 will only be able to access entities labeled with c_0 or c_5, with the c_0, c_5 pair or without any category. sVirt's contexts become as follows:

```
# Subjects
system_u:system_r:svirt_t:s0:c1,c5: VM1's KVM process
system_u:system_r:svirt_t:s0:c2,c10: VM2's KVM process
# Objects
system_u:object_r:svirt_image_t:s0:c1,c5: VM1's image
    file
system_u:object_r:svirt_image_t:s0:c2,c10: VM2's image
     file
system_u:object_r:svirt_image_t:s0: shared image file
```

The PIGA-Cloud designation service has been integrated to OpenNebula using the libvirt library. In the proposed implementation, the OpenNebula's drivers are responsible for choosing the contexts and putting them in libvirt's configuration files. Therefore, the designation preserves the context chosen for the first instance when the VM is suspended or migrated through the XML configuration file of sVirt. The listing 1 shows how the `system_u:system_r:svirt_t:s0:c1,c8` security context of a VM's process is associated to the `system_u:object_r:svirt_image_t:s0:c1,c8` security context of the VM's files.

```
<seclabel type='static' model='selinux'>
  <label>system_u:system_r:svirt_t:s0:c1,c8 </label>
  <imagelabel>system_u:object_r:svirt_image_t:s0:c1,c8
      </imagelabel>
</seclabel>
```

Listing 1 Extract of the configuration used to deploy a VM

The designation service is fully transparent since PIGA-Cloud, e.g. through an integration to OpenNebula, chooses which contexts can be assign to the images through a dedicated database.

This database of the designation allows to know which contexts have already been given and to which VMs they are associated. Thus, when a new VM is launched, the designation assigns an unused context to the VM from the database. Thus, the OpenNebula drivers deploy transparently a consistent labeling all along the life-cycle of a VM.

The following listing shows the *ps* and *ls* commands after a VM deployment: the contexts are properly assigned to both the processes and images.

```
[root@node1 ~]# ps auxZ | grep one-186
system_u:system_r:svirt_t:s0:c1,c8 oneadmin 7901 17.3
    [...]
[root@node1 ~]# ls --scontext /one/var/186/
    images/disk.0
system_u:object_r:virt_image_t:s0:c1,c8 /one/var/186/
    images/disk.0
```

When a VM migrates, the designation maintains the chosen label on the new host. The following result is thus obtained after a migration.

```
[root@node2 ~]# ps auxZ | grep one-186
system_u:system_r:svirt_t:s0:c1,c8 oneadmin 6110 17.3
    [...]
[root@node2 ~]# ls --scontext /one/var/186/images/disk
    .0
system_u:object_r:virt_image_t:s0:c1,c8 /one/var/186/
    images/disk.0
```

Direct Mandatory Protection

Through SELinux/sVirt, several VMs run on the same node while being isolated from each other, but they can be also allowed to share a same image file. For instance with isolation, when a process with an unauthorized label (`system_u:system_r:svirt_t:s0:c1,c3`) attempts a malicious access to (`system_u:object_r:svirt_image_t:s0:c1,c2`), SELinux stops the interaction, as presented in listing 2. Then, OpenNebula informs the user that the VM couldn't be launched (c.f. listing 3). This is the expected behavior since a malicious process tried to get an illegal access to an image file. Thus, SELinux/sVirt protects against both confidentiality and integrity violations of the image file.

```
type=AVC msg=audit(1321711900.859:169931): avc:
    denied { read write }
 for  pid=1796 comm="kvm" name="disk.0" dev=dm-5 ino
    =7471710
 scontext=system_u:system_r:svirt_t:s0:c1,c3
 tcontext=system_u:object_r:virt_image_t:s0:c1,c2
    tclass=file
```

Listing 2 SELinux stops malicious direct accesses

```
error: Failed to create domain from
 /one/var/127/images/deployment.0
error: internal error process exited while connecting
    to
monitor: kvm: -drive file=/one/var/127/images/disk.0
 ,if=none,id=drive-ide0-0-0,format=raw: could not open
 disk image /one/var/127/images/disk.0:  Permission
    denied
```

Listing 3 OpenNebula cannot launch the malicious VM

Thanks to SELinux/sVirt, PIGA-Cloud monitors the direct information flows between the VMs. It is then possible to (1) stop the direct flows between a VM and image files, (2) allow two VMs to share a same image file, and (3) send the SELinux requests to the shared reference monitor PIGA-Shared in order to get a better protection. The Sect. 5 explains how the shared monitor PIGA-Shared guarantees advanced security properties inside a VM and between VMs.

4.3 Protection Inside a JVM

In order to protect PaaS and SaaS solutions, Java applications have to be controlled. However, a satisfying mandatory access control is really missing for the Java Virtual Machine.

Solutions exist to control the flows inside the JVM. JAAS is an API to provide access control based on stack inspection (SBAC). However this API must be called explicitly by the developer in the application, limiting thus its use and effectiveness [26, 27]. A mandatory access control inside the JVM is developed in [28, 29]. However, they use coloration that limits the supported security properties and affects the performances. [30] proposes a similar approach with the same limitations.

Security Enhanced Java (SEJava) is a novel mandatory model controlling the direct interactions between the Java objects sharing the JVM. SEJava provides a reference monitor to intercept all the operations occurring inside the JVM. With SEJava, the source and target security contexts are Java objects. Java objects interact through method calls and attribute accesses. SEJava uses JVMTI (Java Virtual Machine Tool Interface) in order to intercept the interactions between Java objects. Thus, the SEJava reference monitor controls all the operations on the Java objects. This reference monitor guarantees a direct MAC policy between the Java objects.

Security Contexts

With SEJava, a security context is based on the object type, that is to say on the name of the object class. For instance, $Ljava/io/File;$ is the label of the $java.io.File$ class.

Direct Mandatory Protection

In order to define a direct policy, SEJava needs to define the permissions between the security contexts. To do so, SEJava reuses the operations defined by the Java specifications:

- method calls: invocations
- attribute accesses: read or write

```
package se.java;  // Java's package to which the
    usecase belong
public class UseCase {
    public static void main(String args[]) {
        Guest guest = new Guest();  // An object with
            restricted
                                            privileges
        Admin admin = new Admin();  // An object
            owning a secret
```

```
        Storage f = new Storage();  // An ordinary
            Java object

        // The Admin's "secret" is copied to a shared
            data
        f.Write(admin.GetSecret());

        // The content of this shared data is then
            copied as Guest's
        secret guest.SetSecret(f.Read());

        // From this execution point:
        //  An instance of Guest knows the "secret" of
             an Admin's
             instance
        //  The confidentiality of Admin's secret
            field is thus
             violated.
        }
}
```

Listing 4 Example's source code

The listing 4 shows a basic example using three Java objects:

- *Guest* is an unprivileged user. The class has a private field and *get* and *set* methods to handle it.
- *Admin* is a privileged user, inheriting from *Guest*.
- *Storage* is a Java object with a *read* and a *write* method.

```
# [allow/deny] signature --{ permission }--> signature
# Allows Usecase to create new instances of Guest,
    Admin and Storage
allow Usecase --{ main invoke <init> }--> Admin
allow Usecase --{ main invoke <init> }--> Guest
allow Usecase --{ main invoke <init> }--> Storage
# Main has to be able to invoke methods from Storage
allow Usecase --{ main invoke Read }--> Storage
allow Usecase --{ main invoke Write }--> Storage
# Main is allowed to invoke GetSecret on Admin
# but not SetSecret on Guest (see commented rule)
allow Usecase --{ main invoke GetSecret }--> Admin
allow Usecase --{ main invoke SetSecret }--> Guest
```

Listing 5 Extract of a SEJava direct policy

The listing 5 presents an extract of the SEJava policy associated with this program. This policy limits the operations that can be done by the application. Here, the policy allows three kinds of activity:

- the first three rules allow the *main()* function from class *Usecase* to instantiate the classes *Admin*, *Guest* and *Storage*
- the two next rules allow *main()* to read and write the *Storage* instances
- the two last rules allow *main()* to read the *Admin* secret and write the *Guest* secret

```
# decision (signature, id) --{ permission }-->
    (signature, id)
 ALLOW (Usecase, 552) --{ main invoke <init> }-->
    (Guest, 480)
 ALLOW (Guest, 480) --{ <init> invoke <init> }-->
    (Guest, 480)
 ALLOW (Usecase, 552) --{ main invoke <init> }-->
    (Admin, 056)
 ALLOW (Admin, 056) --{ <init> invoke <init> }-->
    (Admin, 056)
 ALLOW (Admin, 056) --{ <init> invoke <init> }-->
    (Admin, 056)
 ALLOW (Usecase, 552) --{ main invoke <init> }-->
    (Storage, 912)
 ALLOW (Storage, 912) --{ <init> invoke <init> }-->
    (Storage, 912)

 ALLOW (Usecase, 552) --{ main invoke GetSecret }-->
    (Admin, 056)
 ALLOW (Usecase, 552) --{ main invoke write }-->
    (Storage, 912)

 ALLOW (Usecase, 552) --{ main invoke read }-->
    (Storage, 912)
 ALLOW (Usecase, 552) --{ main invoke SetSecret }-->
    (Guest, 480)
```

Listing 6 SEJava's decision

The listing 6 shows the SEJava logs generated during a program execution. The three objects are properly instantiated, due to the first three rules of listing 5. Likewise, the *Admin* secret can be written in *Storage* and *Guest* can read the *Storage*.

SEJava can stop an abnormal program execution. Indeed, any behavior unspecified in the policy (for instance the writing of *Admin*'s secret) would be denied. This mechanism can block real program flaws, such as the CVE-2012-1723 flaw.[2]

However, an indirect flow between *Admin* and *Guest*, going through *Storage*, can be noticed. This flow violates the confidentiality of the *Admin*'s secret and should therefore be forbidden. SEJava cannot stop the indirect flows. PIGA-Shared can prevent such a threat as described in the next section.

5 Advanced Mandatory Protection

Mandatory Access Control solutions with monitoring of direct activities require intricate configurations. Moreover, these solutions cannot control advanced activities such as indirect flows or combined activities. The configuration of a full system can entail millions of rules. Therefore, guaranteeing security objectives, e.g. integrity

[2] http://www.symantec.com/connect/blogs/examination-java-vulnerability-cve-2012-1723

or confidentiality of particular resources, is very difficult. In Cloud Computing, this problem becomes even harder due to shared resources and possible migrations of machines, resources or services.

The PIGA approach, developed in [14] and [31], eases the mandatory protection approach while covering a large set of security properties.

PIGA reuses existing direct access control policies, like SELinux, SEWindows, sVirt or SEJava. MAC policies become simpler, i.e. less precise, due to PIGA guaranteeing advanced properties by itself. Besides, PIGA allows to check indirect flows with intermediate contexts and, even more, monitoring all kind of malicious activities including associations of various direct and indirect activities. In practice, PIGA is able to prevent millions of potential vulnerabilities that a direct mandatory policy cannot block. A PIGA policy requires few instantiations of security properties in practice. For example, PIGA-OS [9] significantly strengthens an operating system with a dozen of rules only.

Different reference monitors already control direct flows. Thus, PIGA just adds other mandatory controls and is not an alternative to neither SELinux/SEWindows nor SEJava. It improves and simplifies existing mandatory protections. Therefore, the PIGA approach allows us to reach our objectives, i.e. simplify the administration of mandatory policies while guaranteeing a large set of security properties.

5.1 PIGA Properties

In order to satisfy advanced properties, PIGA uses (*templates*) of advanced security properties expressed using the dedicated *Security Property Language* (SPL). In practice, a security officer simply provides the right MAC security contexts for these templates in order to define instances of advanced security properties. Let us present only three examples of template while more than ten templates are supported and newer ones can easily be defined to cover specific security needings.

A first template guaranties the integrity for a set sc_1 of target security contexts regarding a set sc_2 of source security contexts. It is an advanced property since it can prevent any direct $>$ or indirect flow $>>$.

```
define integrity( sc1 in SCS, sc2 in SCC )
[       ¬(sc1 > sc2)
        AND
        ¬(sc1 >> sc2) ];
```

A second template guaranties the confidentiality for a set sc_1 of target security contexts regarding a set sc_2 of source security contexts. This advanced property prevents any direct or indirect reading of the target security contexts.

```
define confidentiality( sc1 in SCS, sc2 in SCC )
[       ¬(sc2 > sc1)
        AND
        ¬(sc2 >> sc1) ];
```

A third template was designated specifically during the security challenge [9] to cover a large range of threats associated with interpreted executions of downloaded scripts. That advanced property prevents a source security context sc_1 from writing into a target security context sc_2, e.g. a downloaded script, then executing a target security context sc_3, e.g. a binary shell interpreter, in order to read the created file sc_2, e.g. executes the shell script. It is a very powerful property since it only prevents the latter interaction, i.e. the reading, when it is suspicious, i.e. temporal relationships $-then->$ define the malicious activity.

```
define dutiesseparationbash( sc1 in SC )
[       Foreach sc2 in SCC, Foreach sc3 in SC,
        ¬ ( (sc1 →write sc2) -then-> (sc1 →execute sc3) -then-> (
            sc3 →read sc2) ) ];
```

Once the required templates have been properly designed, the security officer simply provides the satisfying parameters using the relevant MAC security contexts. Thus, he easily defines an efficient protection using a couple of advanced properties as shown in the sequel.

5.2 PIGA Global Architecture

PIGA-Cloud uses the mandatory access control policies of each Cloud component in order to control advanced activities. Figure 2 describes the global process used to analyze the security properties and the deployment of PIGA policies. The PIGA policies are required for the control of advanced activities. [32] proposes a distributed access control architecture. Their goals are the same as ours but they do not demonstrate that their approach is applicable on real Cloud environments. Furthermore, their approach is limited to express classical access control rules and can not express advanced security policies as supported with our language. Other works such as [33] only apply these concepts to web services.

The security administrator defines, with the SPL language, the set of PIGA security properties that must be enforced on the three levels of protection. A use case and some examples of security properties are given in the following sections.

In order to analyze the respect of these security properties, the PIGA compiler uses the set of direct mandatory access control policies of each Node/Virtual Machine/-Java software and it computes a set of activity graphs. In an activity graph, a node corresponds to a security context and an edge corresponds to the set of allowed operations. Thus, an edge in the graph is associated to a direct activity, and a path in the graph is associated to an indirect activity (a particular case of advanced activity). A combination of several edges and paths corresponds to an advanced activity.

The compiler analyzes the graphs and enumerates the set of advanced activities that violate the required security properties. Each forbidden activity is added to a database of malicious activities. This database is deployed on the PIGA-Shared monitor in order to control the activities and prevent a malicious activity on each

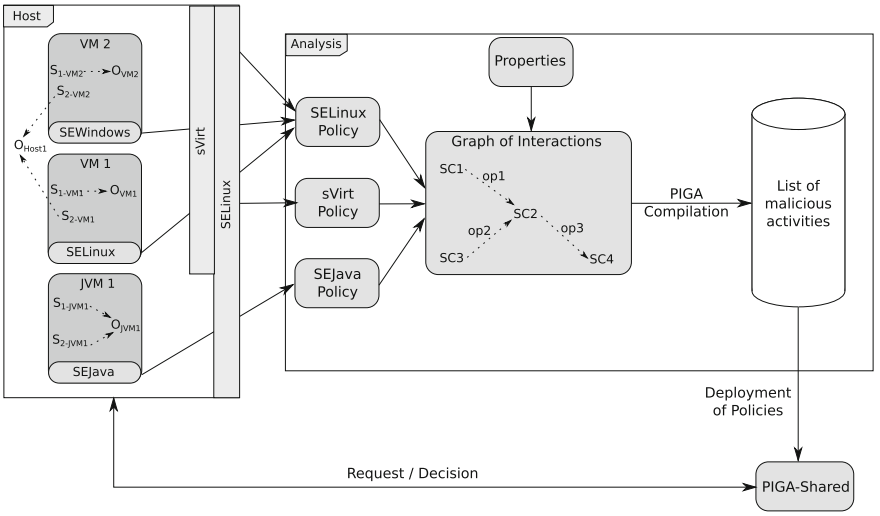


Fig. 2 PIGA-Cloud : analysis and deployment of security properties

level of the Cloud. As shown in Fig. 1, this reference monitor, written in Java, is executed on a dedicated host and it is himself protected with SELinux.

During the execution of the Cloud, PIGA-Shared receives activity requests from:

- SELinux hosts
- Virtual Machines, i.e. each Linux VM that includes SELinux or each MS Windows VM that includes SEWindows [34]
- Java Virtual Machines

PIGA-Shared allows an operation when it does not end any malicious activity of the database. Otherwise, this operation is denied. Thus, PIGA-Shared ensures that the components of the Cloud do not violate any of the required properties.

5.3 Use-Case

Figure 3 shows the use case explained in the following sections. In this example, a Cloud administrator connects to a web-service, running on a Linux node, through Firefox running on a MS Windows node. SEWindows is installed on the MS Windows node and SELinux on each Linux nodes. The web-service is a Java application protected with SEJava. This web-service allows the administrator to update the `/etc/shadow` file of each node of the Cloud. The administrator can add/modify/delete user accounts on the host running the web-service and the web-service can deploy the update on each node. In this architecture, each node is a Virtual Machine running in a QEMU/KVM process and being protected with SELinux/sVirt.

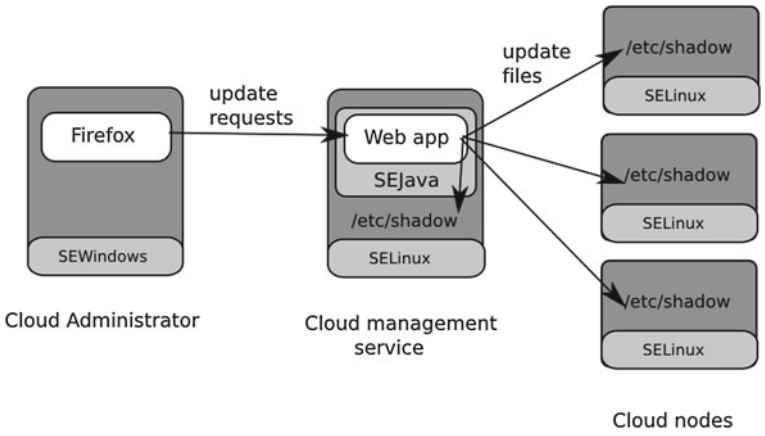


Fig. 3 PIGA-Cloud:use case example

5.4 Advanced Protection Inside an Operating System

The first objective of PIGA-Shared is to control the activities for the operating systems of the hosts or the operating systems running in the VMs. By using our property templates with SELinux contexts, it is possible to control advanced activities and thus provide advanced system protection.

The following listing gives an example of instantiation of the property templates for host systems and virtual machines:

```
integrity( $sc1:="user_u:user_r:user.*_t", $sc2
    :=".*:.*:.*_exec_t" );
integrity( $sc1:="user_u:user_r:user.*_t", $sc2
    :=".*:.*:.*etc_t" );

confidentiality( $sc1:="user_u:user_r:user.*_t", $sc2
    :=system_u:
     object_r:shadow_t );
confidentiality( $sc1:="user_u:user_r:user.*_t", $sc2
    :=system_u:
     object_r:memory_device_t );

dutiesseparationbash( "user_u:user_r:user.*_t" );
```

The first two rules ensure the integrity of executables and system configuration files against the user. The two following rules ensure the confidentiality of the `/etc/shadow` file and the `/dev/mem` device (that allows a direct access to the physical memory). Finally, the last rule prevents a user process to download and run a script interpreter that reads the downloaded script in order to execute it. In our use case, those two properties prevent an illegal reading of `/etc/shadow` or `/dev/mem` for ordinary users. Thus, only the required system services, e.g. the Cloud management service, can read the confidential data.

This last property is very interesting because it does not prevent the download of files, or the execution of interpreters. This rule only prevents the interpretation of scripts that have been downloaded. Of course, it is possible to define a less restrictive property. For example using only the Firefox web browser context as parameter will only limit the execution of downloaded scripts for Firefox. With that later property, the Cloud administrator would be protected only against the vulnerability of his web browser.

5.5 *Advanced Protection for the Microsoft Windows Operating System*

Let us give two examples of the `confidentiality` template in order to protect the Cloud officer using a MS Windows Client.

```
confidentiality( $sc1:=".*:.*:opera_t",$sc2:=".*:.*:
    firefox_t" );
confidentiality( $sc1:=".*:.*:firefox_t",$sc2:=".*:.*:
    opera_t" );
```

Let us imagine that the Cloud officer uses Opera to access the Cloud management service and Firefox to access social networks. The Cloud officer wants to forbid information flows between these two browsers. The first property prevents any flow from Firefox to Opera. The second one prevents any flow from Opera to Firefox.

Starting from a direct policy, PIGA pre-computed respectively 111 and 99 illegal activities for the first and second properties.

Let us give an example of those pre-computed illegal activities.

```
system_u:system_r:firefox_t -( file { create write } )
    ->
system_u:object_r:systemroot_file_t ; system_u:
    system_r:firefox_t -(
file { execute } )-> system_u:object_r:adobearm_exec_t
    ;
system_u:system_r:adobearm_t -( file { execute getattr
    read }  )->
system_u:object_r:systemroot_file_t; system_u:system_r
    :adobearm_t -(
file { create setattr write } )-> user_u:object_r:
    user_home_opera_t
; system_u:system_r:opera_t -( file { execute getattr
    read } )->
system_u:object_r:user_home_opera_t ;
```

Listing 7 Illegal indirect flow from Firefox to Opera

Firefox writes data on the MS Window filesystem. Then, Firefox executes Adobearm. The Adobearm process reads the same filesystem and copies data to the `user_home_opera_t` directory. Finally, Opera reads or even executes data coming from the `user_home_opera_t`.

The direct policy enables each direct interaction. However, it cannot prevent the indirect flow `firefox_t >> opera_t` associated with the causal closure of the direct flows:

```
firefox_t > systemroot_dir_t > adobearm_t >
    user_home_opera_t > opera_t
```

Listing 8 Causal closure of the direct flows

Obviously, that causal closure could be prevented simply by forbidding some of the direct interactions through a modification of the direct policy, e.g. one can prevent Adobearm from writing the user home directory through the direct policy. However, such a modification of the direct policy can prevent the applications from running or even from securing the system, e.g. the Adobe process cannot update the Opera directory in order to correct some security issues associated with the Adobe tools. In the considered indirect flow, the risk is associated with some corrupted data coming from Firefox being transmitted to Opera. Such threats are classical since they could be associated with Firefox executing a JavaScript exploit attempting to compromise indirectly other components, e.g. Opera reading the corrupted data.

PIGA-Shared

In practice, the PIGA monitor gets the requests, i.e. the traces of the requested system calls, from the driver in order to compute on-line the system activities. If a system activity matches with a pre-computed illegal activity, then the PIGA monitor sends a deny for the corresponding system call.

Let us give an example for two requests associated with the beginning of the previously considered illegal activity.

```
type=AVC msg=audit(begin=[129648732304525985],
end=[,]) avc : granted { write } for pid=3884
com="%programfiles%\firefox\firefox.exe" ppid=1960
    path="%systemroot%"
scontext=system_u:system_r:firefox_t
 tcontext=system_u:object_r:systemroot_dir_t tclass=
     file
```

Listing 9 Firefox requesting to write into the system root directory

```
type=AVC msg=audit(begin=[.............41343],
end=[]) avc : granted { read execute getattr } for pid
    =1880
com="%programfiles%\adobe\adobearm.exe" ppid=160 path
    ="%systemroot%"
scontext=system_u:system_r:adobearm_t
tcontext=system_u:object_r:systemroot_dir_t tclass=
    file
```

Listing 10 Adobearm requesting to read into the system root directory

```
type=AVC msg=audit(begin=[.............25985],
end=[.............87541]) avc : granted { write  } for
    pid=3884
com="%programfiles%\firefox\firefox.exe" ppid=1960
   path="%systemroot%"
scontext=system_u:system_r:firefox_t
tcontext=system_u:object_r:systemroot_dir_t tclass=
   file
```

Listing 11 End of Firefox writing into the system root directory

```
type=AVC msg=audit(begin=[.............41343],
end=[.............98312]) avc : granted { read execute
    getattr }
    for pid=1880
com="%programfiles%\adobe\adobearm.exe" ppid=160 path
   =""%systemroot%"
scontext=system_u:system_r:adobearm_t
tcontext=system_u:object_r:systemroot_dir_t tclass=
   file
```

Listing 12 End of Adobearm reading into the system root directory

The two conditions of a causal relationship between two interactions are satisfied: (1) `systemroot_dir_t` 0.9 and 0.10 is a shared context between the Firefox and the Adobearm interactions and (2) the beginning of the first interaction (.....25985) 0.9 is lower than the end of the second interaction (.....98312) 0.12. Thus, PIGA-Monitor detects a causal relationship between the two interactions corresponding to the system activity `firefox_t` > `systemroot_dir_t` > `adobearm_t`. So, PIGA monitor starts the reconstruction of the considered illegal activity.

When the latest interaction occurs, i.e. `opera_t` requesting to read a file in `user_home_opera_t` written by `adobearm_t`, PIGA monitor denies the corresponding system call. It is an efficient property since Opera can read safe files from `user_home_opera_t` but is prevented from reading files written by Firefox.

5.6 Advanced Protection of Virtual Machines

One of the PIGA-Cloud objectives is to control flows between a VM and its host, but also between VMs.

In [13], the authors propose an architecture for protecting web applications. They use a mandatory access control within and between VMs (Virtual Machines). However, they are limited to one node and can not express advanced properties that would include processing an information flow corresponding to a causal closure of direct flows crossing intermediate resources. Rueda et al. [35] analyze multiple SELinux policies. But again, as in [13], they have only one node hosting multiple VMs and do not take into account the deployment or migration of VMs. Finally, the approach is

limited to the analysis and does not protect the system. In [36], the authors decompose a complex policy in several layers. They control the interactions within and between VMs. However, they do not offer a language to express advanced properties and again cannot control the indirect flows.

In order to control flows between a VM and a host, properties are proposed using security contexts associated to the virtual machines. It is then possible to control the advanced activities between the KVM processes (the VMs) and the host. The following rule can prevent a QEMU/KVM process to execute a binary that is not the QEMU binary, the system libraries or the QEMU libraries:

```
trustedpathexecution( $sc1:=system_u:system_r:svirt_t
    , $TPE:= {
     ".*:.*:.*qemu_exec_t",   ".*:.*:.*lib_t",
         ".*:.*:.*lib_qemu_t"
     } )
```

This rule can prevent accessing the host binaries from the running KVM process, for instance, a binary shell interpreter. Additional rules guaranteeing the host confidentiality and integrity have already been defined.

It can sometimes be necessary to share disks or resources between several virtual machines. To monitor the activities occurring between VMs, a new attribute has been added to the VMs' SELinux contexts, so that it becomes possible to identify the machines. Contexts can therefore be represented as follows: `user_u:role_r:type_t:vm_i`, where `vm_i` is the VM's id. Thus, the following rule ensures the confidentiality for the */etc* files of VM1 towards the users of VM2:

```
confidentiality(user_u:user_r:user.*_t:vm_2, system_u:
    object_r:
     etc_t:vm_1);
```

The following listing shows a sequence of activities that would violate the previous property using a NFS sharing:

```
system_u:object_r:etc_t:vm_1 -( { write } )-> system_u
    :object_r:
    nfs_t ; system_u:object_r:nfs_t -( { read } )->
       system_u:
    object_r:user_t:vm_2
```

In order to guarantee the confidentiality property, PIGA will deny the last operation of this sequence. PIGA-Shared can therefore authorize legitimate sharing while preventing malicious ones.

5.7 Advanced Protection Inside a JVM

PIGA-Shared also allows a more efficient protection of the Java applications.

In the example depicted in Sect. 4.3, an indirect activity, leading to a confidentiality violation of the secret object `Admin` regarding the object `Guest`, was computed

by PIGA. This activity being composed of several interactions, it is not possible to block it using SEJava. However, thanks to PIGA-Shared, a confidentiality property can be instantiated to block this flow:

```
confidentiality(system_u:java_r:guest_j, system_u:
    java_r:admin_j);
```

By using this property, the secret of `Admin` will be thus written in the shared object `Storage`, but PIGA-Virt will prevent the copy of the secret from the shared `Storage` by `Guest`. The confidentiality of the `Admin` secret will be then satisfied.

A similar rule guaranties the confidentiality of *Guest*:

```
confidentiality(system_u:java_r:admin_j, system_u:
    java_r:guest_j);
```

These two rules allow PIGA to find four malicious activities that could violate the confidentiality of *Admin* and of *Guest*. These four indirect activities are depicted in the following listing:

```
#Sequences leading to confidentiality violation of
    Admin
system_u:java_r:usecase_j -( getSecret { invoke } )->
    system_u:
    java_r:admin_j ; system_u:java_r:usecase_j -(
        addSecret {
    invoke } )-> system_u:java_r:guest_j
system_u:java_r:admin_j -( addSecret { invoke } )->
    system_u:java_r
    :storage_j ; system_u:java_r:guest_j -( getSecret {
         invoke } )
    -> system_u:java_r:storage_j
#Sequences leading to the confidentiality violation of
     Guest
system_u:java_r:usecase_j -( getSecret { invoke } )->
    system_u:
    java_r:guest_j ; system_u:java_r:usecase_j -(
        addSecret {
    invoke } )-> system_u:java_r:admin_j
system_u:java_r:guest_j -( addSecret { invoke } )->
    system_u:java_r
    :storage_j ; system_u:java_r:admin_j -( getSecret {
         invoke } )
    -> system_u:java_r:storage_j
```

This protection prevents the threats within that Java application used as a Web gateway for the Cloud management service. Indeed, that Java application collects all the passwords into the shared storage. Thus, a guest user can access the shared storage only when the admin secret is not available into the shared storage. That advanced property prevents the exploits within the Java objects. Thus, a malicious user logged as guest into the Cloud management service cannot get the secrets of the Cloud administrator.

Table 1 Properties analysis result

Name		Host Centos/SELinux/sVirt	VM Gentoo/SElinux	Java JVM 7
Graph of interactions	Nb contexts	2897	577	210
	Nb operations	1 879 063	17684	960
Properties	`integrity`	0	0	
	`integrity`	74	30	
	`confidentiality`	145 718	86 268	
	`confidentiality`	100 468	65 648	
PIGA	`dutiesseparationbash`	103 747 632	14 629 680	
	`trustedpathexecution`	8 715		
	`confidentiality`			4

5.8 Policy Analysis

This subsection outlines the results of the PIGA policy for the three protection layers, the policy is composed of the rules previously presented:

```
#Properties at hosts and virtual machines level (level
    1)
integrity( $sc1:="user_u:user_r:user.*_t", $sc2
    :=".*:.*:.*_exec_t" );
integrity( $sc1:="user_u:user_r:user.*_t", $sc2
    :=".*:.*:.*etc_t" );
confidentiality( $sc1:="user_u:user_r:user.*_t", $sc2
    :=system_u:
     object_r:shadow_t );
confidentiality( $sc1:="user_u:user_r:user.*_t", $sc2
    :=system_u:
     object_r:memory_device_t );
dutiesseparationbash( "user_u:user_r:user.*_t" );

#Property at hosts level (level 2)
trustedpathexecution( $sc1:=system_u:system_r:svirt_t
    , $TPE:= {
     ".*:.*:.*qemu_exec_t",  ".*:.*:.*lib_t",
         ".*:.*:.*lib_qemu_t"
     } )

#Property at JVM level (level 3)
confidentiality(system_u:java_r:guest_j, system_u:
    java_r:admin_j);
```

Table 1 shows for each property the number of illegal interactions / sequences / compositions of sequences. PIGA will block all these illegal activities.

Thus, with only one rule, PIGA can detect and forbid millions of illegal activities. For example, the *dutiesseparationbash* property for a Linux host prevents around

100 millions of malicious activities. For the JVM, few illicit activities are prevented due to the simplicity of the considered Java application.

6 Lessons Learned and Future Works

More and more vulnerabilities affecting systems are presented in the news. Indeed, the number of levels in the information is increasing. For example, Cloud environments add new levels, such as virtualization of operating systems, that need dedicated protections.

For instance, Java was the target of multiple attacks in the previous months. For instance, the CVE-2012-1723[3] is a recent flaw that has been exploited in July 2012. Companies such as Facebook and Twitter were also victims of attacks based on Java vulnerabilities.

Moreover, the ANR security challenge [9] demonstrated that the PIGA approach is more efficient that other solutions based on virtualization. Indeed, the other competitors presented solutions based on virtualization that were unable to protect a whole system as well as PIGA-OS.

Consequently, the ANR security challenge proved that only an in-depth MAC protection is truly efficient to protect a system. This result can be extended to Cloud environments. Indeed, a Cloud does not reduce the attack surface but instead makes it larger since all the levels involved in the Cloud architecture need to be protected. Therefore, all the physical hosts present in the Cloud need to be protected with an appropriated MAC protection. The guest virtual machines running operating systems also need protection. Finally, the application level, for instance the Java objects running inside a Java VM, has to be monitored.

However, the experiments done during the ANR security challenge show that the classic MAC implementations, such as SELinux, SEWindows or SEJava cannot efficiently protect a system, since they are only able to control direct activities. Furthermore, these solutions are complex to use since they need very large policies that are difficult to formalize.

The results obtained with PIGA prove that it is possible to simplify the definition of the security requirements for a system. PIGA-Cloud extends the approach to cover all the levels of a Cloud environment, but also to provide an end-to-end guarantee, thus simplifying the definition of a security policy.

In practice, PIGA-Cloud protects the system against millions of vulnerabilities remaining in classical direct MAC policies.

Future works deals with the adaptation of SEJava for the Dalvik virtual machine. Thus, a secure PIGA-Android Client will be proposed controlling the Linux system and Java applications since advanced protections are really missing for Android. Therefore, an end-to-end security could be offered for PIGA-Android clients accessing PIGA-Cloud services.

[3] http://www.symantec.com/connect/blogs/examination-java-vulnerability-cve-2012-1723

Currently, the weakness of the solution is that the approach does not enable to combine different security systems, such as Discretionary Access Control, Mandatory Access Control and methods using cryptography. The Seed4C project[4] aims at addressing these issues. For this purpose, the security language will be extended to support the specificity of DAC, cryptography and network security tools. A dedicated middleware will be proposed for deploying the security requirements regarding the Cloud cartography and the available security tools.

7 Conclusion

To summarize, an in-depth mandatory access control can improve the protection of Cloud environments. Activities are controlled at different levels, i.e. at the host and guest levels for an IaaS Cloud and at the application level, e.g. Java, for PaaS or SaaS Clouds. PIGA-Cloud ensures a consistent labeling of the guest virtual machines during their life cycle including migrations and suspensions. The PIGA approach eases the definition of advanced properties monitoring several direct and indirect flows while supporting a large set of confidentiality/integrity properties. PIGA-Cloud reuses existing direct mandatory policies, e.g. SELinux and sVirt. Moreover, the SEWindows and SEJava approaches allow to control direct flows within a MS Windows host and between Java objects. Through its unified language PIGA-Cloud eases the administration of an in-depth protection facilitating thus the Cloud security administration. Albeit our approach is independent from the Cloud environment, PIGA-Cloud has been integrated and evaluated for OpenNebula. A use case shows the advanced protections supported by PIGA-Cloud. The flows between the guest virtual machines are efficiently controlled. Advanced properties within a MS Windows or Linux guest are proposed. Our use case shows also that the advanced controls of Java objects and MS Windows Clients allow a safe end-to-end administration of the Cloud through a classical Web browser and Java gateway.

References

1. Smalley S, Vance C, Salamon W (2001) Implementing selinux as a linux security module. NAI Labs Report 1:43
2. Morris J (2009) sVirt: Hardening linux virtualization with mandatory access control. Linux.conf.au conference, In
3. Briffaut J, Lefebvre E, Rouzaud-Cornabas J, Toinard C, (2011) Piga-virt: an advanced distributed macprotection of virtual systems. In: VHPC, (2011) 6th workshop on virtualization and high-performance cloud computing. Bordeaux, France 2011
4. Sotomayor B, Montero RS, Llorente IM, Foster I (2009) Virtual infrastructure management in private and hybrid clouds. IEEE Internet Comput 13(5):14–22
5. Pearson S, Benameur A (2010) Privacy, security and trust issues arising from cloud computing. In Proceedings of the 2010 IEEE second international conference on cloud computing technol-

[4] http://celticplus-seed4c.org/

ogy and science, CLOUDCOM '10, Washington, DC, USA, 2010. IEEE Computer Society, pp 693–702.
6. Jaeger T, Schiffman J (2010) Outlook: cloudy with a chance of security challenges and improvements. IEEE Secu Priv Mag 8(1):77–80
7. Vaquero LM, Rodero-Merino L, Morán D (2011) Locking the sky: a survey on iaas cloud security. Computing 91:93–118
8. Sandhu R, Boppana R, Krishnan R, Reich J, Wolff T, Zachry J (2010) Towards a discipline of mission-aware cloud computing. In: Proceedings of the 2010 ACM workshop on Cloud computing security workshop, CCSW '10, New York, NY, USA, 2010. ACM, pp 13–18.
9. Briffaut J, Perès M, Rouzaud-Cornabas J, Solanki TC, Venelle B (2011) Piga-os: Retour sur le système d'exploitation vainqueur du défi sécurité. In 8ième Conférence Francophone sur les Systèmes d'Exploitation, 2011.
10. Takabi H, Joshi JBD, Ahn G (2010) Security and privacy challenges in cloud computing environments. IEEE Secur Priv 8(6):24–31
11. Harrison MA, Ruzzo WL, Ullman JD (1976) Protection in operating systems. Commun ACM 19(8):461–471
12. Lampson BW (1971) Protection. In: The 5th symposium on information sciences and systems, Princeton University, March 1971, pp 437–443.
13. Hicks B, Rueda S, King D, Moyer T, Schiffman J, Sreenivasan Y, McDaniel P, Jaeger T (2010) An architecture for enforcing end-to-end access control over web applications. In Proceedings of the 15th ACM symposium on Access control models and technologies, SACMAT '10, New York, NY, USA, 2010. ACM, pp 163–172.
14. Jérémy B (2007) Formalisation et garantie de propriétés de sécurité système : application à la détection d'intrusions. PhD thesis, Thèse de doctorat en informatique, Université d'Orléans, 13 décembre 2007.
15. Loscocco P, Smalley S (2001) Integrating flexible support for security policies into the linux operating system. In: 2001 USENIX annual technical conference (FREENIX '01), Boston, Massachusets, United-States, 2001. USENIX Association.
16. Boebert WE, Kain RY (1985) A practical alternative to hierarchical integrity policies. In: The 8th national computer security conference, Gaithersburg, MD, USA, October 1985, pp 18–27.
17. Core Labs. Core force user's guide. October 2005, pp 1–2.
18. Gros D, Toinard C, Briffaut J (2012) Contrôle d'accès mandataire pour Windows 7. In: SSTIC 2012, Rennes, France, June 2012, pp 266–291.
19. Keller E, Szefer J, Rexford J, Lee RB (2010) Nohype: virtualized cloud infrastructure without the virtualization. SIGARCH Comput Archit News 38(3):350–361
20. Szefer J, Keller E (2011) Lee RB (2011) Eliminating the hypervisor attack surface for a more secure cloud. ACM conference on computer and communications security, In
21. BitVisor 1.1 Reference Manual. http://www.bitvisor.org/, 2010
22. Carbone M, Zamboni D, Lee W (2008) Taming virtualization. IEEE Secur Priv 6(1):65–67
23. Quynh NA, Takefuji Y (2006) A real-time integrity monitor for xen virtual machine. In: ICNS '06: Proceedings of the international conference on networking and services, Washington, DC, USA, 2006. IEEE computer society, p 90.
24. Sailer R, Jaeger T, Valdez E, Caceres R, Perez R, Berger S, Griffin JL, Van Doorn L, Center IBMTJWR, Hawthorne NY (2005) Building a MAC-based security architecture for the Xen open-source hypervisor. In: Computer security applications conference, 21st Annual, 2005, p 10.
25. Raj H, Nathuji R, Singh A (2009) Resource management for isolation enhanced cloud services. CCSW '09 Proceedings of the 2009 ACM workshop on Cloud computing, security, 2009, p 77.
26. Abadi M, Fournet C (2003) Access control based on execution history. In: Proceedings of the 10th annual network and distributed system security, symposium pp 107–121, 2003.
27. Pistoia M (2007) Beyond stack inspection: a unified access-control and information-flow security model. In: SP: security and privacy. IEEE 2007:149–163

28. Vivek H, Deepak C (2005) Michael F (2005) Dynamic taint propagation for java. Department of Information and Computer Science - University of California, Technical report
29. Vivek H, Deepak C (2005) Michael F (2005) Practical, dynamic information-flow for virtual machines. Department of Information and Computer Science - University of California, September, Technical report
30. Nair S, Simpson P, Crispo B, Tanenbaum A (2008) Trishul: a policy enforcement architecture for java virtual machines. In: Technical, Report IR-CS-045, May 2008.
31. Rouzaud-Cornabas J (2010) Formalisation de propriétés de sécurité pour la protection des systèmes d'exploitation. PhD thesis, Thèse de doctorat en informatique, Université d'Orléans, 2 décembre 2010.
32. Almutairi A, Sarfraz M, Basalamah S, Aref W, Ghafoor A (2012) A distributed access control architecture for cloud computing. IEEE Softw 29(2):36–44
33. Calero JMA, Edwards N, Kirschnick J, Wilcock L, Wray M (2010) Toward a multi-tenancy authorization system for cloud services. IEEE Secur Priv 8(6):48–55
34. Briffaut J, Toinard C, Gros D (2012) Contrôle d'accès mandataire pour windows 7. In: Symposium sur la sécurité des technologies de l'information et des, communications, 2012, pp 266–291.
35. Rueda S, Vijayakumar H, Jaeger T (2009) Analysis of virtual machine system policies. In: Proceedings of the 14th ACM symposium on Access control models and technologies, SACMAT '09, New York, NY, USA, 2009. ACM, pp 227–236.
36. Payne BD, Sailer R, Cáceres R, Perez R, Lee W (2007) A layered approach to simplified access control in virtualized systems. SIGOPS Oper Syst Rev 41:12–19

Part II
Cloud Privacy and Trust

Identity Management in Cloud Systems

Ginés Dólera Tormo, Félix Gómez Mármol and Gregorio Martínez Pérez

1 Introduction

Identity management systems are of paramount importance to provide authentication and authorization based on end user identities trying to preserve privacy, while at the same time enhancing interoperability across multiple domains. Traditional identity management systems allow end users, to some extent, to manage their personal information for accessing certain services.

However, cloud computing brings a different perspective related to the end users' interests. New risks arise, especially due to the fact that the number of devices acting in the system grows exponentially [48]. In this regard, some kind of attacks could be more dangerous and the number of malwares to be considered and malicious users that could potentially join the system increases.

Additionally, end users are more concerned about how their data is managed, where it is located and who can access it. In this sense, cloud computing is changing some of the basic assumptions. As a result, any service in the cloud is exposed to trust, security and privacy threats that can compromise the identity of end users.

Improving the end users experience while offering certain novel identity-related features has been recently achieved by means of applying advanced identity management systems. These systems are designed to deal with authentication and authorization processes, enabling Single Sign-On and methods to exchange end users information between different entities and/or domains. By establishing trust links among

G. Dólera Tormo (✉) · F. Gómez Mármol
NEC Laboratories Europe, Kurfürsten-Anlage 36, 69115 Heidelberg, Germany
e-mail: gines.dolera@neclab.eu

F. Gómez Mármol
e-mail: felix.gomez-marmol@neclab.eu

G. Martínez Pérez
Departamento de Ingeniería de la Información y las Comunicaciones Facultad de Informática, Universidad de Murcia, 30100 Murcia, Spain
e-mail: gregorio@um.es

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_6,

different providers, end users are granted to securely access different resources or services using a single identity, yet preserving end users privacy.

A lot of work has been done in this area, improving and adapting this kind of systems to different needs. A wide variety of identity management systems have been proposed to fulfill different requirements of particular environments or to deal with different challenges that certain contexts pose [18]. However, due to the variety of features offered by the different identity management systems, it is not trivial to determine which approach fits better in a given context.

Based on requirements and threats related to the cloud computing model, our main contribution is to present different identity management approaches, in order to analyze and compare how they fit in the cloud context. The questions these systems leave open and the ongoing work in this regard are described as well afterwards. We also provide a set of recommendations to be taken into consideration when designing or deploying any identity-based service in a cloud environment.

The remainder of the document is organized as follows. Section 2 presents some working groups, standardization activities and on-going international projects aimed to analyze identity-related challenges raised by current and forthcoming technologies. Section 3 presents some representative use cases on the field of identity management for cloud computing in order to help the reader to contextualize subsequent sections. Section 4 describes a set of requirements and threats to be taken into account when working with identity management systems. Section 5 presents a comprehensive survey on the most relevant identity management solutions existing nowadays applicable to cloud computing. Each approach is analyzed, showing its advantages and limitations. Finally, an extensive comparison of all these solutions is provided. Section 6 sketches some foreseen practical and realistic scenarios of application of advanced identity management techniques within the scope of cloud computing, while Sect. 7 extracts the current research challenges to be addressed in order to reach the aforementioned envisioned practical scenarios. To conclude, Sect. 8 presents some final remarks depicting the main findings of our research work.

2 Related Work

This section presents a set of related works in the field of identity management. It describes working groups, standardization activities and international projects whose objective is to identify, analyze, and describe identity management challenges and pending issues.

The OASIS Identity in the Cloud Technical Committee (OASIS IDCloud TC) [41] works to address the security challenges posed by identity management in the context of cloud computing. Its main goal is to collect and harmonize definitions, terminologies and vocabulary of Cloud Computing, and develop profiles of open standards for identity deployment, provisioning and management. Definition of protocols, APIs or implementations is out of scope of this TC.

OASIS IDCloud TC collects use cases to help identify gaps in existing Identity Management standards, and investigate the need for profiles to achieve interoperability within them, with a preference for widely interoperable and modular methods. Additionally, it works with standards bodies to recommend changes to existing standards trying to close current gaps. Use cases categories include identity infrastructure establishment, identity management, authentication, authorization, accountability and attribute management, security tokens, governance and audit and compliance.

The National Strategy for Trusted Identities in Cyberspace (NSTIC) [55] is a White House initiative to work with both public and private sectors in order to improve the privacy, security, and convenience of sensitive online transactions. It offers a collaborative vision to create the standards, policies, guidelines and recommendations needed for interoperable trusted credentials that would dramatically reduce identity theft and fraud online.

NSTIC introduces the so-called Identity Ecosystem, aimed at protecting end users privacy by helping them to verify that the websites they browse are legitimate, avoiding fake sites designed to steal personal information. Furthermore, it enforces service providers to follow a standard set of best practices in order to ensure end users that their personal data will be fairly handled, that they are informed on how their data will be used, and to enable them meaningful choices, while accountability features are deployed. For example, service providers would be required to collect and share the minimum amount of information necessary for authentication.

Additionally, the Kantara Initiative [36] is committed to help in driving policies and technical interoperability in order to verify trust in the identity-based experience of end users, Relying Parties and Federation Operators. Additionally, it works collaboratively to solve harmonization and interoperability challenges among identity-enabled enterprise, Web 2.0 and Web-based applications and services. The goal of this activity is to provide public and private sector organizations with uniform means of relying on digital credentials issued by a variety of identity providers in order to advance trusted identity and facilitate public access to online services and information.

Moreover, the Simple Cloud Identity Management (SCIM) specification suite [28], developed under the IETF, is designed to ease the use of identity management in cloud-based applications and services. SCIM seeks to build upon experience with existing schemas and deployments, placing specific emphasis on simplicity of development and integration, while applying existing authentication, authorization, and privacy models. Its intent is to reduce the cost and complexity of user management operations by providing a common user schema and extension model, as well as binding documents to provide patterns for exchanging this schema using standard protocols.

In turn, Identity Commons [33] is a community of groups working on developing the identity and social layer of the web. Its main purpose is to support, facilitate, and promote the creation of an open identity layer for the Internet, in such a way that control, convenience, and privacy for the individual are improved.

The main objective of the Web Identity Working Group [60], developed by the World Wide Web Consortium (W3C), is to provide Web developers with a secure

and uniform access to elementary cryptographic operations, session state information, and authentication credentials for devices and applications like browsers. Web Identity Working Group aims to produce specifications that have wide deployment amongst end users, adopting, refining and extending existing practices and community-driven draft specifications.

Furthermore, Attribute-based Credentials for Trust (ABC4Trust) [1] is a research project funded by the 7th Research Framework Programme (FP7) of the European Union as part of the Trust and Security Program. The goal of ABC4Trust is to address the federation and interchangeability of technologies that support trustworthy yet privacy-preserving Attribute-based Credentials (ABC). ABC enhances classical trustworthy credentials, which normally do not respect privacy and reveal more information than required by the service. This project defines a common, unified architecture for ABC systems to allow comparing their respective features and combining them on common platforms, in order to deliver open reference implementations of selected ABC systems.

Likewise, PrimeLife [49] is another European Union project funded by its 7th Framework Programme. The main objective of the project is to bring sustainable privacy and identity management to future networks and services. This project addresses challenges related to end users digital interactions over the Internet, which involve leaving a life-long trail of personal data. PrimeLife advances the state of the art in the areas of human computer interfaces, configurable policy languages, web service federations, infrastructures and privacy-enhancing cryptography.

Finally, Secure Widespread Identities for Federated Telecommunications (SWIFT) [53] is also a European Union funded project within the 7th Framework Programme. The project leverages identity technology, building a cross-layer identity management framework as a key to integrate service and transport infrastructures for the benefit of end users and providers. It focuses on extending identity functions and federation to the network while addressing usability and privacy concerns for end users. SWIFT prepares the grounds for a new dimension of business dynamics allowing a fast entry of new players while expanding the business of existing ones.

Table 1 summarizes the related work presented in this section.

3 Identity Management Usecases

This section will present some use cases on the field of identity management for cloud computing. In cloud environments, there could be several use cases identified regarding identity management, ranging from straightforward ones, such as those introducing common SSO concepts, to very complex scenarios, including for instance government provisioning or mobile customers.

The goal of this section is to present those use cases defining representative identity management scenarios which could help the reader to identify requirements, threats, features and challenges that these scenarios raise. This will establish the grounds to better understand the forthcoming sections. The use cases described here are inspired

Table 1 Overview of related work

Identifier	Responsible Organization(s)	Purpose	Type
OASIS IDCloud TC	OASIS	Collect terminologies of Cloud Computing and develop profiles of open standards for identity management in Cloud Systems	Working group
NSTIC	White House	Protect end users privacy by creating standards, policies, guidelines and recommendations needed for interoperable trusted credentials	Working group
Kantara Initiative	Community Members	Stimulate identity community interoperability through the development criteria for operational trust framewors	Working group
SCIM	IETF	Ease the use of identity management in cloud-based applications and handle provisioning of user identity across cloud-based service providers	Specification
Identity Commons	Community Members	Support the creation of an open identity layer for the Internet in order to maximizes control, convenience, and privacy for the individual	Working group
Web Identity Working Group	W3C	Provide web developers secure and uniform access to cryptographic operations and authentication credentials	Working group
ABC4Trust	EU member consortium	Define a common architecture for ABC systems to allow comparing and combining their respective features and delivering an open reference implementations	European Project
PrimeLife	EU member consortium	Address the core privacy and trust issues raised by users' daily interaction over the Internet	European Project
SWIFT	EU member consortium	Building a cross-layer identity management framework, extending identity functions and federation to the network while addressing usability and privacy concerns	European Project

on the Identity in the Cloud Use Cases working draft [51], which provides a set of use cases examining the requirements on identity management functions applied to the cloud.

3.1 UC01: Federated Single Sign-On and Attribute Sharing

There are numerous cloud services in the Internet offered by many different cloud service providers which, in turn, belong to many different domains. It is considered common for end users to have an account for each of these cloud services they want to use, having also different credentials for each of them. For example, they have to create a new account, protected by a specific username/password for accessing some resources or services offered by a given cloud service provider.

Federated Single Sign-On is a process that allows end users to access multiple services having a unique set of credentials. Once end users have been authenticated in their home domain, they do not need re-authentication for accessing different services, even if such services belong to external domains. Additionally, these services

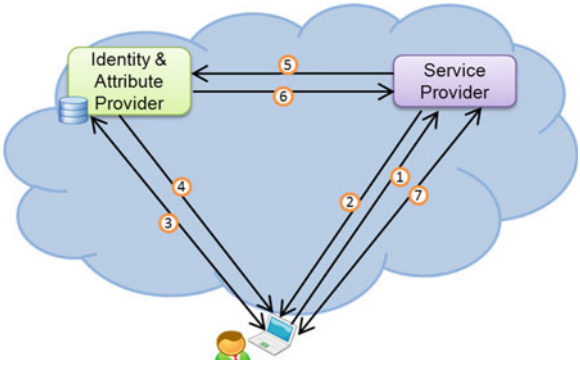


Fig. 1 Graphical representation and process flow of UC01

may also require retrieving users' information to provide the service or to perform some access control process.

Users' information is usually represented as attributes, such as age, country, postal address, etc. Again, to avoid users indicating such attributes for each service they want to use, identity management systems are planned in such a way that the service providers could recover the required attributes from their home domain, i.e., from their unique account, as long as a trust relationship exists between the querying domain and the domain providing such requested information. In this way, authentication and attributes could be asserted if they have been issued by a trusted party, although mechanisms to exchange such kind of information have to be defined (Fig. 1).

3.1.1 Process Flow

1. End user wants to access a service offered by a service provider
2. The service provider requires end users to be authenticated
3. End user is authenticated in her home domain
4. Home domain asserts user authentication
5. The service provider requires user attributes to offer the service
6. Home domain asserts user's attributes
7. End user accesses the service

3.2 UC02: Attribute Aggregation and Operations

End users usually have their attributes spread over multiple information sources, maintained by different providers in different domains. For example, academic information could be managed by their university, while information related to their postal

address could be managed by their city hall and their credit card information is managed by their bank.

Although different information sources could mean different contexts, end users may want to present attributes maintained by different domains to the same cloud service provider at the same time. Furthermore, end users may want to perform some operations over the attributes in order to present just the required information to access a service. In this way, they could present some claims based on the attributes, but no the attributes themselves.

Service providers need to validate the received claims, for instance to check whether they (or the attributes they refer) are still valid. On the other hand, end users may want to present self-asserted attributes, describing some information about them, although such information is not asserted by a relying party (Fig. 2).

3.2.1 Process Flow

1. End user is asked by a service provider to present certain attributes, which could belong to different sources, in order to access a service
2. End user gathers attributes from different sources. Optionally, end user makes operations based on that attributes to claim the required information
3. End user sends the required attributes or the generated claims to the service provider
4. Service provider validates the attributes or claims
5. End user accesses the service

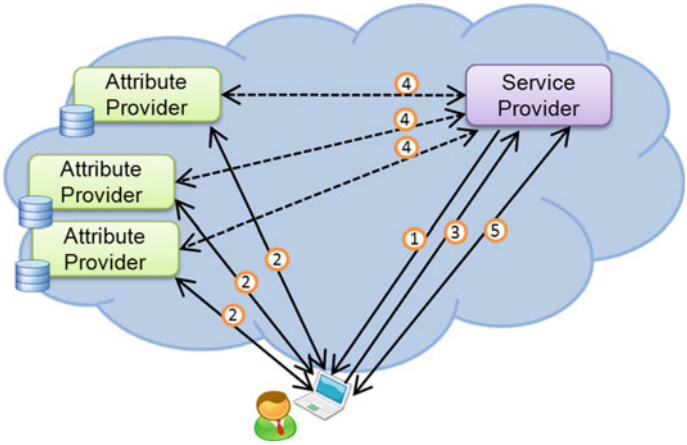


Fig. 2 Graphical representation and process flow of UC02

3.3 UC03: Identity Privacy in Shared Environment

Cloud service providers usually require end users attributes either to provide the cloud service itself (e.g. online shopping services require the postal address to send the purchased items), to perform access control (e.g. on-line film services may require the age of the end user to provide horror movies) or to provide customized services (e.g. a website showing different aspects according to the user language).

However, both end users identities and end users attributes are considered private information, and only reliable parties should gain access to them. In this way, end users identities must be hidden and they may give their explicit consent before any of their attributes is released. Furthermore, end users should release no more information than the strictly required by each cloud service provider, so they may want to choose which attributes will be released for each interaction with the cloud services.

To enable end users to achieve such a process, they do not only need the appropriate tools which allow selecting their attributes, but end users should be also properly informed about the service they want to interact, e.g. level of trustworthiness, fulfillment of privacy policies, etc. In this way, they can take the appropriate decision of allowing or not the service to obtain its attributes (Fig. 3).

3.3.1 Process Flow

1. A service provider requires end user attributes to provide a service
2. End user is informed that the service provider wants to access her attributes. In this step, detailed information of the service provider is shown to the end user
3. End user gives her explicit consent to release her attributes. Additionally, the end user selects the attributes which will be released

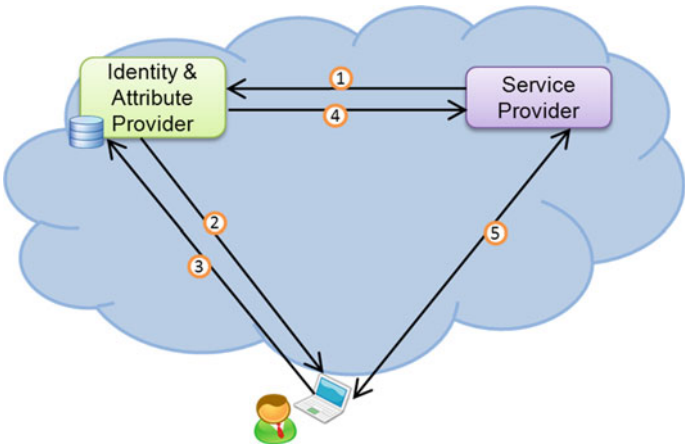


Fig. 3 Graphical representation and process flow of UC03

4. The service provider gets end user attributes
5. The service provider supplies the service based on the obtained attributes

4 Requirements and Threats

In a so demanding context, as is the case of cloud computing, identity management systems need to provide a set of features and give a minimum of guarantee that they properly fulfill the required behavior. Based on the described use cases, this section studies the main functional requirements as well as security threats to be considered when designing and deploying a new identity management system for cloud computing, or when selecting a currently existing one.

4.1 Requirements

Requirements have been grouped into three main categories according to their relation within the context of identity management systems, entitled (1) general requirements, which describes essential functionality that is expected from any identity-related system; (2) user-centric capabilities, which encompasses requirements related to the control offered to the end users for inferring in the interaction between different providers; and (3) information management functionalities, which defines the allowed operation that the end users have when they present information to a third party.

- General requirements:

  R1 **Confidentiality and integrity**: Since any identity management system makes use of sensitive information, they must assure that such information is shared only between appropriate entities. Additionally, they must guarantee that the information remains valid and complete even when is exchanged between different parties. In this way, identity management systems have to use secure communications channels and deploy the appropriate measures in order to ensure confidentiality and integrity of the information.

  R2 **Single Sign-On**: There are multiple services deployed in the Internet, belonging to many different domains, each of them managing their own set of credentials. A key requirement for the usability and security of any identity management system is to allow users of a domain to access applications hosted in another domain using the credentials of the domain they originally belong to. From the end users perspective, it is desirable to benefit from a SSO mechanism, avoiding having an account for each service they want to access.

  R3 **Logging and Auditing**: Logging and auditing discourage attackers since their attempt to subvert the system will be recorded and/or they will leave a trail

which can be further tracked back to them. Moreover, if something unexpected happens, the lack of logs can result in significant difficulties while dealing with the occurred failure. Identity management systems must incorporate an effective logging and auditing mechanism able to trace relevant events happened in the system. This requirement guarantees that an end user or entities cannot deny having performed an operation or initiated a transaction, i.e., non-reputation is provided. To achieve this, the identity management systems may trace sent and received messages and audit (part of) their content, as well as internal operations.

R4 **Strong authentication**: Authentication mechanisms based on shared secrets such as common username-password authentication, do not offer enough protection avoiding impersonation or identity theft. In identity management systems, authentication mechanisms guaranteeing certain level of security need to be deployed, such as those based on biometric techniques or digital certificates, enhancing the level of security of the whole system.

R5 **Justifiable parties**: An identity management system must make its end users aware of the party with whom they are interacting while sharing information, and give certain indications about the level of reliability this party has. In turn, the relying party should also be able to confirm that the information presented by an end user is reliable, for instance if it has been validated by an authority.

- User-centric capabilities

R6 **End user consent**: When an identity provider needs to release some personal information about an end user, for instance when requested by a service provider to access a given service, the end user should be able to explicitly approve whether such information could be released or not. For example, the identity provider should show a confirmation page when some attributes are requested to permit the end users to decide if they want to continue, or not, with the transaction.

R7 **Control of accumulated data**: End users of identity management systems usually release some of their information to other entities, sometimes to enable another entity to manage their information on their behalf. Yet, the end users should be able to control which information each entity has about them, and to know how this information is being secured and protected. This is the case when the attributes of the end user are directly managed by the end user instead of by an identity provider.

R8 **Usability**: One of the main objectives of identity management systems is to ease any identity-related process to end users. This could not be achieved if end users are required to complete complex procedures, manage complicated tools or have advanced technical knowledge in order to interact with services. Instead, identity management systems should provide user-friendly interfaces and intuitive procedures when any identity-related functionality is presented to them.

R9 Off-line mode: Once an end user has authorized a transaction, the exchange of information between entities might be done without the intervention of the end user. Furthermore, if an attribute of the end user changes, the service provider should be able to get the updated value without needing interaction of the end user. For instance, a magazine service provider requires the postal address of their subscribed end users to send a printed version of their magazines and an end user makes use of her town hall identity provider to reveal her address. If the end user changes her address, it would be desirable that the magazine service provider automatically gets the new end user postal address without requiring the end user in such a process.

- Information management functionalities:

R10 Attribute aggregation: End users usually have multiple digital identities depending on the context they are involved. These identities could belong to different identity providers, each of them managing different kind of information. For instance, academic information of a given end user could be managed by the identity provider of her university, while information about her credit card is managed by the identity provider of her bank. Any user-centric identity management system should allow the end users to aggregate attributes from their different identities in order to present a combined set of claims to a given service provider at once.

R11 Attribute revocation: Some of the end users' attributes are not permanent but they can change throughout time or have an expiration time. Furthermore, the attributes could be revoked either by the identity provider which issued them or by the end user, for instance if she wants to cancel an account. When a service provider gets an end user's attribute, it should be able to check whether such attribute is still valid or not.

R12 Self-asserted attributes: End users' attributes usually need to be issued by an authority such as a trustworthy identity provider, in order to prove its validity. However, in some cases may be necessary to allow the end users to issue their own attributes if proving the validity of them is not mandatory. For instance, some service providers could require some non-critical attributes, such as hobbies, language or country, just to provide a customized service.

R13 Minimal disclosure information: The end users should be able to selectively reveal as less information as possible in the credentials presented to the service provider. For example, if an end user wants to make use of her driver license to prove she is older than 21, she should be able to extract a claim just related to her birth date without including the rest of information. Otherwise the service provider could gain all the other information contained in the driver license. Furthermore, the end users should be able to generate new valid claims based on others valid claims in order to prove that they fulfill a policy without revealing attributes. For instance, an end user should be able to prove that she is older than 21 making a verifiable claim based on her birth date but without actually revealing her birth date.

4.2 Threats

Identity management systems are exposed to a number of threats that can compromise its behavior when malicious users or entities try to subvert the system. We classify the threats into the next three categories:

- **Trust threats**: Identity management systems are aimed to simplify the end user experience by creating considerable trust complexity for both service providers and identity providers. They require an infrastructure where all the involved parties must be trusted for specific purposes depending on their role. However, if one of the parties acts maliciously, then the rest of the participants could be exposed to different risks. Identity management systems need to deploy mechanisms to allow entities to trust each other although in some scenarios the trust conditions could introduce some threats [17, 24, 25] if they have not been properly taken into account.
- **Security threats**: Any communication system is exposed to different risks which can compromise the security of the whole system. Malicious users are constantly coming up with new ways to attack any system, focusing their efforts on exploiting vulnerabilities of those systems. This is especially relevant for systems managing identity-related information, due to the fact that they potentially manage sensitive information [39]. Identity management systems need to avoid any threat which allows an attacker to affect negatively the system, from stealing information of the end users or acting on their behalf, to interfere in the communication or interrupt services.
- **Privacythreats**: Privacy is a desired feature of any communication system. End users usually want to keep the information of their digital identities secret. However, having information about the end users is increasingly being considered more and more valuable [38]. Furthermore, some organizations do not need to know the real identities of their end users, but they want to collect the behavior of each of them. Identity management systems have to deploy mechanisms to preserve end users' privacy. That includes (1) anonymity, where a service cannot know the real identity of an end user, (2) unlinkability, where a service provider cannot link different end user's accesses and (3) untraceability, where an identity provider cannot know the services that one of its end users has accessed.

5 Evaluation of Identity Management Approaches

In this section we introduce identity management standards, technologies and solutions which allow end users to manage their personal attributes required for accessing certain services. We analyze these approaches highlighting benefits and drawbacks of each in regards to the previously presented requirements. Finally, we summarize our analysis with a comparative table.

5.1 SAML

SAML [44], short for Security Assertion Markup Language, is an XML-based open standard which defines a framework for exchanging authentication, entitlement and attribute information between entities. In general terms, it allows entities, usually identity providers, to make assertions regarding the identity of end users to other entities, usually a service provider.

The first version of SAML (SAML 1.0) was adopted as an OASIS standard in November 2002. Since then, a minor revision (SAML 1.1) was made in November 2003, and a major revision (SAML 2.0) was made in March 2005, which is the most widespread version. Several organizations are offering support to SAML 2.0, being Shibboleth [20] the reference solution for this standard.

In the common workflow of SAML, as shown in Fig. 4, an end user wants to access a service from a service provider, but this service provider needs to authenticate the end user and obtain some attributes about her. The authentication process, instead of being performed by the service provider, is delegated to the identity provider, which is in charge of managing the user's identity.

To this end, the service provider redirects the end user to her identity provider along with a SAML Authentication Request. The identity provider asks the end user for her credentials, for instance with the usual username/password form, although other authentication methods could be used. Once the identity provider validates the authentication, it redirects the end user back to the service provider along with a SAML Assertion indicating that the end user has been authenticated.

At this point, the service provider may request some attributes of the end user sending SAML Attribute Query messages directly to the identity provider. Since the end users do not have to manage their attributes but they are managed by the identity provider, this solution is usually easy to use and no technical knowledge is required from the end users.

Nevertheless, end users are not aware of which attributes are being released. In fact, it is hard for them to control the information that a given identity provider accumulates about them. Even though some implementations allow defining some

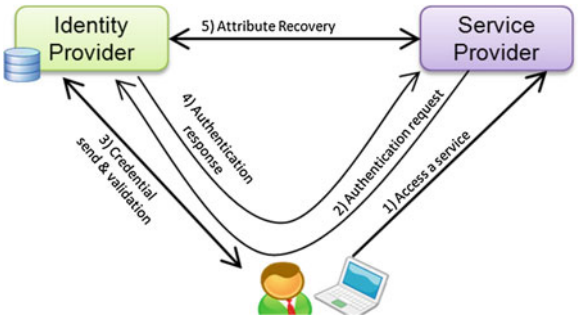


Fig. 4 SAML general workflow

attribute release policies, they are also managed by the identity provider. Additionally, end users are not asked for consent before releasing their attributes.

Each identity provider is also in charge of managing the trust relationships with the service providers. The assertions indicating authentication and attributes are digitally signed by the identity provider, in such a way that their validity could be verified by the service provider. However, there is no option for the end users to present self-asserted attributes to a service provider. Furthermore, even though some SAML identity providers allow obtaining attributes from different information storages, an end user could not present assertions from different identity providers at the same time.

The issued assertions make use of pseudonyms preserving the end user's privacy. That is, the service providers do not know the end user's real identity. However, the identity provider could trace all end users accesses since it has to generate an assertion each time they need to access a service provider. Additionally, end users should initialize the transaction to allow the service provider get attributes, making the offline mode requirement hard to achieve.

One of the main purposes of SAML is providing a SSO mechanism for accessing different service providers with a unique account [15]. Hence, if the unique password of an end user is stolen, the thief could gain access to such services providers on behalf of the end user. Furthermore, a malicious service provider could redirect the end users to a fake identity provider, presenting a similar aspect to the original one, asking for inserting username and password, in order to steal passwords if they do not realize it is a malicious website (the so called "*phishing attack*" [34]).

5.2 OpenID

OpenID [50] is an open technology standard which defines a decentralized authentication protocol in order to allow end users to sign in to multiple websites with the same account. The original OpenID authentication protocol was developed in May 2005, and its current 2.0 version is maintained by the OpenID community since 2007.

With over one billion OpenID enabled end user accounts and over 50,000 websites accepting OpenID, this standard is nowadays widespread in the Internet. It is supported from several large organizations such as AOL, Google, Microsoft, VeriSign and Yahoo!, among many others [46]. Yet, OpenID is not owned by anyone, but the OpenID Foundation in in charge of assisting the community.

When end users create an account in an OpenID provider (i.e. identity provider) they receive an identifier as a URL or XRI. Then, when they access a website which requires authentication and supports OpenID (i.e. relying party or service provider) they may enter their identifier in order to be redirected to their OpenID provider, as shown in Fig. 5. It is worth mentioning that this identifier is usually unique for each end user (e.g. alice.myopenid.net). Therefore, the service provider could trace the end user accesses, since they always use the same identifier.

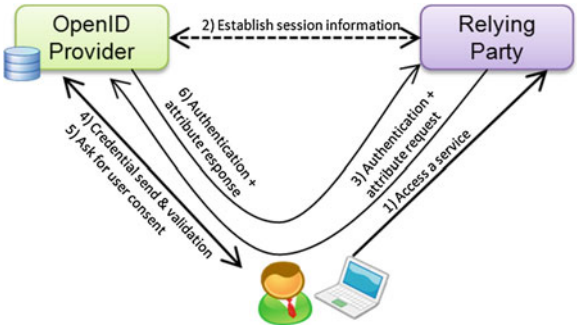


Fig. 5 OpenID general workflow

Being in the OpenID provider, end users could make use of their unique username/password to perform the authentication process. After checking the end user's credentials, the OpenID provider shows a confirmation page, where the end user could verify and select the information which will be released to the service provider. That is to say, end users can check or uncheck the attributes to be released; yet, they cannot make claims based on such attributes.

Finally, end users are redirected back to the service provider, which now can have the requested information about the end users, but neither their password nor their real identity. As commented, the end users' information is managed by the OpenID provider, which makes the system easy to use although requires the user to be online to perform a transaction. Additionally the users can hardly control the information that the OpenID provider gain about them. Furthermore, the OpenID provider could trace end users' accesses since it establishes direct communication with the service provider each time the end user needs to provide an authentication assertion or release some attributes [30].

Even though the attributes are issued by an OpenID provider, the service provider cannot determine the reliability of such an OpenID provider. Since the framework is aimed to distributed environments, where no Certification Authority should be implied, any entity could become an OpenID provider, just implementing the OpenID protocol [26]. The end users can therefore present self-asserted attributes whose validity does not have to be validated by the service provider. However, end users cannot aggregate attributes from different OpenID providers.

The OpenID protocol also presents some security issues [16]. Similarly to the previous standard, if the password of an end user is stolen, the thief could access all the services accepting OpenID on behalf of the end user. Furthermore, a malicious service provider could redirect end users to a fake OpenID provider in the authentication process simulating the real end users' OpenID provider to steal their password.

5.3 OAuth

OAuth [29] is an IETF specification which defines a protocol in order for clients to access server resources on behalf of a resource owner. It provides a process for end users to authorize third-party accesses to their server resources without sharing their credentials. The first specification of the OAuth protocol dates from December 2007, although that version was updated on June 2009 to address the session fixation attack [37] and published as RFC 5849 [27] in April 2010.

Currently, there is a working draft in progress of OAuth 2.0 which is being supported by several companies, such as Facebook, Google and Microsoft. OAuth 2.0 is not backward compatible with OAuth 1.0, although the latter is also extendedly supported by several services providers such as LinkedIn, MySpace, Twitter and Yahoo! [45].

In the common workflow of the protocol, as depicted in Fig. 6, an end user wants to share some of their private resources which are maintained in a server (i.e. identity provider), like photos or documents, with a client (i.e. service provider) without revealing any password to such client.

To achieve this process, end users access the client website, which requests the end users' resources. Since the client supports OAuth, the end users may select the identity provider where the resources will be obtained from. At this point, the client requests a set of temporary credentials from the identity provider, and once received the end user is redirected to the identity provider with those temporary credentials.

End users can now see their identity provider website, where they could perform the common username/password authentication process without revealing their credentials to the client. After performing the authentication process, the end user is asked to grant (or deny) access to the client for getting some of their resources.

Despite its several advantages, this solution does not allow either attribute aggregation between different sources, nor making claims based on attributes, but just releasing them. In fact, the end users cannot choose which attributes will be released, just permit or deny the access to a set of them. Furthermore, the granularity of the set

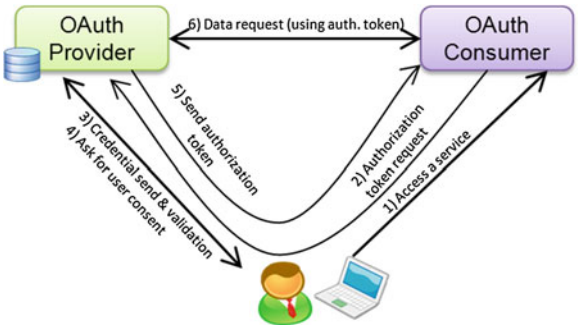


Fig. 6 OAuth general workflow

of attributes or resources to be released depends on the OAuth server. For example, an end user might need to share a whole photo album even if she just wants to share one picture from it.

If the end users approve the request, they are redirected back to the client website, indicating the temporary credentials identifier, informing that they have completed the authorization process. Then, using its temporary credentials, the client requests a set of token credentials to the OAuth server, which will be used for requesting the resources.

Once the client has the set of token credentials, the communication is directly done with the OAuth server without requiring the user to be online. Using this process, the OAuth server hides the real identity of the end user to the client. However, the OAuth server can trace end user's accesses since it communicates with all the services the user wants to make use of.

By establishing a direct communication between OAuth server and client, this solution lets the OAuth server revoke attributes if they are not valid any more, but on the other hand it is difficult for the users to invalidate the granted authorization if they do not want it any more. At most, users could establish some expire period when they confirm the authorization.

Additionally, the client does not know how much can trust in an OAuth server and the validity of the provided attributes. This protocol makes no attempt to verify the authenticity of the server. This solution is mainly focused on allowing access to resources, where the validity of the resources does not have to be validated by an authority. In this sense, this solution allows self-asserted attributes, although it depends on the functionality offered by the OAuth server.

Similar to the previous solutions, this approach is easy to use, since it just shows some user-friendly web pages, although controlling the accumulated data is hard to achieve since it is managed by the OAuth server. It also has similar security problems regarding stolen passwords, since the same account is used for accessing different services. In the same line a malicious client could redirect the end users to a fake OAuth server in the authentication process trying to steal their passwords. Additionally, OAuth 2.0 tokens are not signed, which tends to simplify the protocol although it must rely on SSL/TLS channels to establish a secure communication, making this protocol vulnerable to Man-in-the-middle attacks [8].

5.4 Cardspace

Windows CardSpace [13, 40], also known as its codename InfoCard, is the Microsoft client or Identity Selector for the Identity Metasystem [35]. Although in February 2011 Microsoft decided not to ship this project any more, it is worth describing this solution since it provides the basis for future technologies such as U-Prove [47].

Taking into consideration that the end users may have different identities depending on the context where they are interacting, the challenge of this approach is to allow the end users to create, use, and manage their diverse digital identities in an

understandable and effective way. For instance, at work end users might have a digital identity assigned by their employer while they maintain a private digital identity in MySpace.com to share some music content.

The idea behind Windows CardSpace is that end users could manage their digital identities, and their related attributes, in a similar way that they manage their cards in their wallets. In this sense, when end users are requested to present some information about them, they open their Identity Selector (their wallet), select one of their cards which contains the requested information, and present it to the requester. The Information Cards are usually issued by trustworthy entities, in order for the requester to verify the validity of the information contained.

To achieve such a process, as summarized in Fig. 7, the requester (i.e. relying party or service provider) supplies some requirements to the end user, such as which claims it expects. The Identity Selector shows the cards which fit the requirements to the end user. Once the end users select a card, the Identity Selector requests to the issuer of this card (identity provider) a security token containing the needed claims. Security tokens are signed to check the identity of the identity provider, establishing this way trust relationships. Finally, the Identity Selector sends the security token to the requester, which could be validated making use of the signature.

Windows CardSpace is entirely agnostic about the format of the security token [4]. For this reason, CardSpace can work with any digital identity system, using any type of security token, including X.509 certificates, Kerberos tickets, SAML assertions, etc.

In contrast to the previous solutions, the information of end users is managed by the end users themselves. The user could see all the information contained in an Information card which eases controlling the accumulated data. Furthermore, since the end user selects the information card to be sent, there is an explicit user consent, which also means that the interaction with the user is needed.

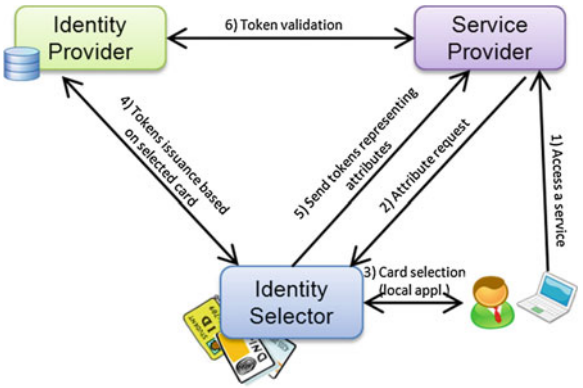


Fig. 7 CardSpace general workflow

However, end users need to manually hold their cards, for instance by installing the cards in their devices. Hence, if the users move to another device they need to carry their cards with them in order to install such cards in the new device in use. This presents further difficulties if the end users want to make use of their cards in a public computer. Additionally, the implementation of reference is integrated with Microsoft Windows, which makes hard to be used in others operating systems, or devices, such as mobile phones.

Although Cardspace could make recommendations about the card to make use of, it is the user who needs to check the information to be released and decide how trustworthy the service provider is. In addition to that, the end user can neither select which attributes inside a card will be released, nor aggregate attributes from different cards, nor make claims based on attributes. The whole card is presented instead.

This solution allows the end user to create personal cards, as if they were issued by a self-identity provider. They usually include personal information such as name, addresses or phone numbers, which does not have to be verified. However, in this case a service provider could trace users' accesses, since the self-issued identity provider uses the same private key for each service provider to sign the tokens.

Moreover, the identity provider generates the user claims each time the user accesses a service in order to form the security tokens. In this way, the identity provider could not directly know the service the user is accessing, but it could trace which identity cards are being used and when. Furthermore, even though by default the service provider's identity is not revealed by the Identity Selector to the identity provider, when a token is requested, the identity provider might require knowledge of the service provider's identity before issuing the requested token (e.g. for creating a Kerberos ticket for that specific service) [2].

When some attributes need to be revoked, the identity provider just has to stop honoring request for security tokens made with this card. If the users want to revoke a card (e.g. from a stolen laptop) they should contact the identity provider. However, self-issued cards could not be revoked, and users should contact each service provider asking for not accepting the self-issued cards.

In order to avoid impersonation, the username/password mechanism for authentication is replaced by using the information card. However, how to acquire information cards is not defined, but it depends on the identity provider. In this sense, if the identity provider does not provide the appropriate measures, the information cards could be stolen and the user could be impersonated [21].

On the other hand, the Information card does not have to contain sensitive data, such as credit card number, but it is maintained in the identity provider, and is released in the security token instead. Additionally, CardSpace improves how websites prove their identity to the users by introducing higher-assurance certificates. These certificates also enable a way for those users to learn the level of assurance a site is offering, which could help them to take decisions about whether to trust a given website.

5.5 Higgins

Higgins is an open source identity framework designed to enhance the end user experience, by integrating identity profiles and social relationships information across multiple sites. The firsts versions of Higgins [19] (Higgins 1.0 and Higgins 1.1), released on 2008, offer an Identity Selector service for the Identity Metasystem. The latest version, Higgins 2.0, is still under development and is planning to implement a Personal Data Service (PDS).

A PDS is a cloud-based service that works on behalf of the end users, giving them a central point for controlling their personal information. It not only manages end users' attributes, but it also manages data flows to external businesses and to other end users' PDS.

As shown in Fig. 8, the functionality and workflow of Higgins is similar to Microsoft CardSpace, but in contrast to it, the cards in Higgins are maintained by a hosted service, outside the devices of the end users. Hence, the Identity Selector of Higgins is mainly a thin client that only implements the end user interface, while the core functionality is performed by the hosted service. In this way, the end users could make use of different devices to access their cards without having to carry them.

However, the end users need a specific piece of software installed in their device as a client application, such as a plugin in the web browser, to make use of the Identity Selector. There are implementations for different operating systems and some intuitive card selectors available, but since the end users need to directly manage their cards, using this solution is not trivial for inexpert users.

Higgins introduces a new kind of Information Card, namely the relationship cards (r-cards) [56]. These cards allow an end user to establish a data sharing relationship between an identity provider and a relying party. In this way, the relying party could directly request end user's attributes to an identity provider without interacting with the end user, but the end users control the authorization to their data.

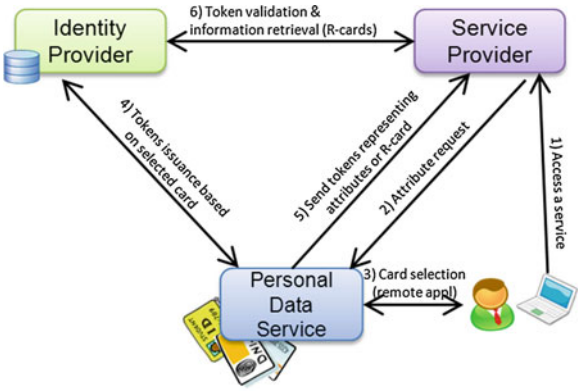


Fig. 8 Higgins general workflow

Either issuing specific credentials for a service provider as CardSpace does, or interacting directly with the service provider through a relationship card, the identity provider could trace users' accesses. Furthermore, the hosted Identity Selector should be placed in a server that the end users really trust, since it not only can trace all their interactions with the different relying parties, but also it is in charge of managing and storing all their identities information.

Regarding user consent, minimal information disclosure, attribute aggregation and revocation requirements, this solution presents similar issues to the CardSpace solution previously presented, since both are based on Information cards.

5.6 U-Prove

U-Prove [47, 57] is a cryptographic technology which presents a type of credential or token to encode end users' attributes in such a way that the issuance and the presentation of such tokens remains unlinkable. U-Prove was developed by Credentica in 2006, acquired and maintained by Microsoft since 2008.

The U-Prove technology makes use of Zero-knowledge proof methods [22, 23] to issue the tokens. Zero-knowledge proof is a way for an end user to prove possession of a certain piece of information without revealing it. That is, an end user can provide an assertion containing a set of attributes revealing nothing beyond the validity of such an assertion and the attributes. In this sense, this method offers the same level of security as X.509 certificates, with additional privacy protecting features.

In a similar way than end users manage Information cards, they may manage U-Prove tokens, as depicted in Fig. 9. These tokens are obtained from different identity providers, which prove the validity of such tokens. Therefore, when a service provider requires some end users' attributes, the end users could present one of their tokens, with the peculiarity that the identity provider is not involved in this process.

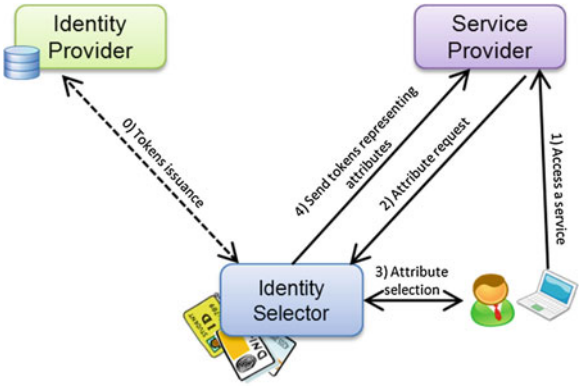


Fig. 9 U-Prove general workflow

That is, the identity provider and the service provider do not have to establish any communication between them.

Furthermore, even if the service provider is the same entity than the identity provider, the identity of the end user presenting the token could not be revealed, due to the fact that the token does not provide information regarding the issuance process which could be traced.

In this solution, the users control their information by themselves, which raises some disadvantages in terms of usability as previously commented when presenting other solutions (i.e. CardSpace and Higgins). On the other hand, end users could decide which of the attributes contained in a token will be released, without presenting the whole token, achieving part of the minimal disclosure information requirement.

However, it does not allow making claims about an attribute without revealing the attribute, for instance in order to prove that the value of an attribute is within a certain range. Although some mechanisms have been proposed to solve this issue [6], they are not efficient. In a similar way, combining tokens or attributes issued by different identity providers, in order to present a set of aggregated attributes at the same time to a relying party, is not available in this solution.

Although U-Prove tokens do not reveal the real identity of the end users, the service provider could trace different accesses of a given end user due to the fact that her tokens present the same public key and signature. This could be solved if the identity provider issues many different tokens to the end user with the same attributes [47], but this solution could be impracticable for large scenarios.

Attribute revocation is available for the users if they contact the service provider to invalidate one of their attributes. Nevertheless, revocation from the identity provider side is hard to achieve and it is an open research question for this technology. Due to the fact that the identity providers are not involved when the end users present a token, the user could be presenting a token even though it has been already revoked by the identity provider. Ordinary Certificates Revocation Lists (CRL) cannot be used since they would break the unlinkability capability of this solution. Some solutions based on unlinkable blacklists have been proposed [7] although they are not practical for large blacklists.

5.7 Idemix

Idemix, short for Identity Mixer [32], is an anonymous credential system following the protocols described by Camenisch and Lysyanskaya [9] in order to allow the end users to control the dissemination of personal information and preserving their privacy. Idemix has been developed at IBM Research and the first design and implementation document [10] dates from 2002.

Idemix makes use of Zero-knowledge proof methods to generate credentials. Similar to U-Prove, an end user can obtain credentials containing attested attributes from identity providers, and prove to a service provider the validity of such attributes without revealing any other information. The credentials are maintained by the end

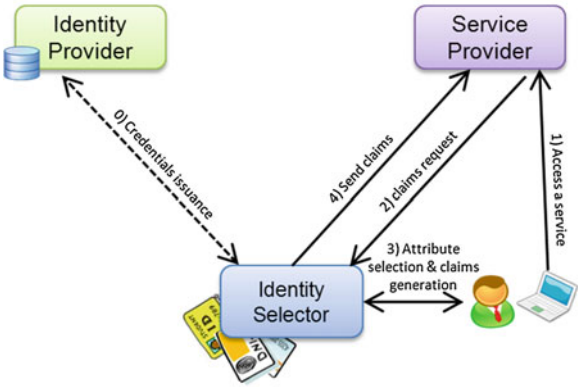


Fig. 10 Idemix general workflow

users in such a way that the identity provider is not required when presenting or validating some users' attributes.

In contrast to U-Prove, Idemix fulfills the selective disclosure of attributes requirement. As shown in Fig. 10, Idemix not only allows the end users to select which attributes will be released to a service provider, but it also has the ability to prove inequalities, such as the value of a birth date attribute being less than today's date minus 21 years without disclosing that birth date attribute value. Idemix could also prove that two attributes have the same value without disclosing that value. However, it neither allows attribute aggregation from different identity providers.

In addition, Idemix tokens are self-blindable. In this way, the end user could transform the token so that they look different each time they need to present and prove some attribute. Therefore, the service provider could not trace end user's accesses. However, this makes the issue of attribute revocation even harder to achieve, which in this case is not available neither for end users, nor for identity providers.

The specification of Idemix presents the usage of short-term claims to replace the revocation behavior. That is, using claims with short expiration time so they need to be renovated every so often. However, although this alternative is valid for certain scenarios, it presents some drawbacks regarding a real revocation mechanism, and some research is currently underway to solve this issue.

5.8 Discussions

This section summarizes the features and limitations of the previously described solutions regarding the requirements presented in Sect. 4.1. Although all solutions fulfill some essential requirements, such as Single Sign-On, confidentiality and integrity requirements, one of the main conclusions is that there is not an ideal approach ful-

filling all the requirements. Instead, selecting the most appropriate solution depends on the features of the scenario and the desired behavior.

In general, solutions which do not allow end users to directly control their information are easier to apply, such as SAML, OpenID and OAuth, since the identity providers are in charge of managing such information on their behalf. However, these solutions are based redirections of the end users whenever authentication is required, and this authentication is usually based on username/password. Hence, they are exposed to impersonation if a malicious service provider redirects the end users to a fake identity provider.

On the other hand, although systems which allow end users to control their information usually present user-friendly interfaces, they often require end users to install and manage some applications, and maintain their credentials manually. On the contrary, they use stronger authentication mechanisms, avoiding impersonation.

Additionally, some of the presented systems could be exposed to other security threats, such as the Man-in-the-middle attack, or session related attacks [39] if they do not deploy the appropriate counter-measures. SAML and OpenID standards indicate that the messages must be digitally signed and uniquely identified, to avoid malicious users to modify or replace an assertion, although OAuth just relies on the SSL/TLS channel to exchange messages so no vulnerability establishing such channel could compromise the security of the whole system. CardSpace and Higgins protect the assertions (i.e. tokens indicating authentication statements or attributes) using signatures and a secure communication channel. However, how to acquire Information cards depends on the identity provider, which could raise some security risks if it does not take possible vulnerabilities into consideration.

Regarding privacy, SAML and OAuth make use of pseudonyms to hide the real identity of the end users, but they do not support minimal disclosure information. In other words, the end users could hardly decide which attributes will be exchanged. Yet, OpenID, CardSpace, Higgins and U-Prove do allow the end users to select which attributes will be released, though service providers could trace end user accesses since they use the same pseudonym to access different services or even to access the same service several times. Idemix has the ability of selecting the attribute to be disclosed [3], while presenting each service provider a different identifier, being really hard for them to trace end users accesses. Furthermore, Idemix could make claims based on attributes, so the attributes are not revealed but information based on them instead.

In order to preserve privacy, some systems avoid direct communication between service providers and identity providers, so the latter could not trace end users' accesses. However, that could result in a tough implementation of other features or requirements. For example, OAuth and Higgins issue authorization tokens to allow the service providers to directly access the user information under certain conditions, instead of sending the information directly into the token. Hence, the identity provider and the service provider could exchange information even if the end users go offline.

Furthermore, in solutions like U-Prove or Idemix, where end users can present attributes without involving the identity provider, attribute revocation is hard to achieve. Additionally, since the identity provider cannot trace end users accesses,

and the end users are completely anonymous to the service provider, it is difficult to provide these systems with accurate audit mechanisms.

Nevertheless, some of the defined requirements are not properly managed by any of the presented systems, like attribute aggregation. Thus for instance, SAML, OpenID and OAuth are focused on having a unique identity provider for managing all identity-related information of the end users, so attribute aggregation is not contemplated. In turn, CardSpace, Higgins, U-Prove and Idemix support credentials and attributes from different identity providers, for instance, having an Information card from each of them. However, they do not allow presenting information asserted by different providers at the same time.

It is also worth mentioning other trust aspects. Identity management systems assume that trust relationships are established, so they usually require the end users attributes to be asserted by a reliable entity (e.g. a trustworthy identity provider). OpenID, CardSpace and Higgins allow end users to assert self-attributes without requiring them to be validated by a trusted party, which could be useful for non-critical scenarios. However, all the presented identity management systems need additional considerations when deployed on more dynamic environments. Additionally, although end users could approve transactions before releasing any private data in some of the systems, they are not informed about the reliability of the service provider. That is to say, whether the service provider is trustworthy enough to obtain their sensitive information or to provide the expected service.

Table 2 presents a comparative of the current identity management solutions, showing how these solutions meet the aforementioned requirements.

6 Visionary Thoughts for Practitioners

This section sketches the envisioned practical and realistic scenarios of application of advanced identity management techniques within the scope of cloud computing.

In the field of cloud computing, new risks are continuously emerging due to the fact that the number of devices acting in the system is growing exponentially. In other words, the more devices deployed in the system, the more harmful some kind of attacks could be. Furthermore, the number of malwares is also dangerously increasing, since more malicious users could join the system, and new kind of attacks arise.

Although we are talking about a “new” technology or concept, most of the challenges related to cloud computing are not actually new. That is, cloud computing is not something really new, but it rather consists of an integration of technologies related to other contexts, such as multi party computation [5], distributed systems [54], etc. Therefore, some challenges, such as privacy, secure data management, network accessibility, etc. can be handled in the same way as they have been managed in other contexts.

Table 2 Comparative of current identity management solutions within the context of cloud computing

Req.	SAML	OpenID	OAuth	CardSpace	Higgins	UProve	Idemix
R1	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved
R2	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved	Successfully achieved
R3	IdPs could log end users accesses	OpenID providers could log end users accesses	OAuth server could log end users accesses	IdPs could log end users accesses	IdPs could log end users accesses	Hard to achieve efficient auditing due to the offered unlinkability properties	Hard to achieve efficient auditing due to the offered unlinkability properties
R4	Authentication mechanism depends on the IdP, although username/password is usually used	Authentication mechanism depends on the IdP, although username/password is usually used	Authentication mechanism depends on the OAuth server, although username/password is usually used	Information cards provide a strong authentication mechanism	Information cards provide a strong authentication mechanism	Authentication based on cryptographic techniques	Authentication based on cryptographic techniques
R5	Static trust relationships are supposed between IdP and SP	RPs do not know the reliability of the IdPs. IdPs and users do not know the reliability of the RPs either	RPs do not know the reliability of the IdPs. IdPs and users do not know the reliability of the RPs either	IdPs cannot prevent the user from sending cards to untrustworthy sites. Users cannot know how reputable a site is	IdPs cannot prevent the user from sending cards to untrustworthy sites. Users cannot know how reputable a site is	IdPs cannot prevent the user from sending cards to untrustworthy sites. Users cannot know how reputable a site is	IdPs cannot prevent the user from sending cards to untrustworthy sites. Users cannot know how reputable a site is
R6	Do not ask for user consent	Explicitly ask for user consent before releasing any user's attribute	Explicit user consent before releasing any user's attribute	The user selects the information card to be sent.	The user selects the card to be sent.	The user selects which attributes will be released	The user selects which attributes will be released
R7	Directly managed by the IdP, not by end users	Directly managed by the IdP, not by end users	Directly managed by the IdP, not by end users	End users store and manage their Information cards by themselves	End users store and manage their Information cards by themselves	End users store and manage their attributes by themselves	End users store and manage their attributes by themselves
R8	Do not require technical knowledge, information is managed by the IdP	Do not require technical knowledge, information is managed by the IdP	Not required technical knowledge. Friendly web pages are usually shown	Requires end users to manually manage their different identities. They even need to make backup of the cards	End user needs to install applications. Similar to Cardspace, it is not trivial for inexpert users	Requires end users to manually manage their attributes. It is not trivial for inexpert users	Requires end users to manually manage their attributes. It is not trivial for inexpert users
R9	Offline mode is not defined in the standard	User interaction is needed tocomplete the information exchange process	Information exchange is directly done between the parties once the authorization	End users interaction is needed toshare their information	Using R-cards the information exchange could be done without end users interaction	End users interaction is needed toshare their information	End users interaction is needed toshare their information

(continued)

Table 2 Continued

			token is issued				
R10	Attribute aggregation between different IdPs is not available	No attribute aggregation between different sources available	Not attribute aggregation between different sources available	Not attribute aggregation availalble. Just one Information card is selected	Not attribute aggregation availalble. Just one Information card is selected	Not available	Not available
R11	When attributes are revoked in the IdP, they are not valid anymore and hence not released to the SPs	When attributes are revoked in the IdP, they are not valid anymore and hence not released to the SPs	When attributes are revoked in the IdP, they are not valid anymore and hence not released to the SPs. However, the user can hardly invalidate an authorization token after being issued	When a card is revoked, the IdP stops grantingrequests for security tokens made with this card. End users should contact their IdPs to revoke cards. Self-issued cards could not be revoked	When a card is revoked, the IdP stops grantingrequests for security tokens made with this card. End users should contact their IdPs to revoke cards. Self-issued cards could not be revoked	End users could revoke tokens,but onlymanually by contacting each SP. Furthermore, for IdPs it is hard to revoke tokens	Neither available for end users nor for issuers due to the unlinkability properties this solution offers
R12	Not self-asserted attributes option available	All the attributes could be self-asserted since no CA should be implied	Depends on the IdP and on the scenario	User could create self-asserted attributes	User could create self-asserted attributes	Available if complemented with CardSpace or Higgins	Available if complemented with CardSpace or Higgins
R13	Once authenticated, the Service Provider is able to request any attribute allowed by the IdP. Hence, the Service Provider could get attributes which are not required	The end users could select the attributes which will be released when asked for the user consent	The user could decide the information which will be released. But the granularity depends on the OAuth server	When the user sends a card, all the information of the card is sent. The user cannot select which attributes inside this card will be released	When the user sends a card, all the information of the card is sent. The user cannot select which attributes inside this card will be released	The end users could select the attributes which will be released	The end users could select the attributes which will be released and claims based on those

Requirement successfully fulfilled Requirement partially fulfilled Requirement not fulfilled

However, cloud computing does bring a different perspective related to the user interests: Can third parties get access to my data? When and why they cannot obtain my data? It is my data protected against intrusion and lost? The distributed architecture of the cloud makes it harder to answer these questions. It is not only a technical issue, it is more related to the "cloud" concept, and the trust that users place in this concept. Cloud computing is changing some of the basic assumptions. The one to one model client-server is no longer conceivable. Now, it is Client-Cloud-Server for legal, contractual, technology, data protection and privacy considerations. Additionally, data can be easily distributed among different countries and jurisdictions requiring the application of different points of view of all these aspects.

Cloud computing leaves lot of questions unresolved, most of them related to the fact that users cannot know where their data is geographically located. They cannot know who to trust, or how secure the data handling is. Privacy implications of cloud computing include: jurisdiction, third party access, security safeguards, limitations on use and retention, and demonstrating/verifying compliance. There is not a universally agreed definition of privacy; privacy is contextual. Perspectives on privacy are influenced by culture, economics, society, politics, religion, history, experience, education, etc. Furthermore, identifying what "personal data" is can be also a hard issue.

There are other issues focused on enterprises. The enterprises' internal data now could be moved outside the bounds of the company, increasing the need for secure collaborations. Additionally cloud-based environments make policies harder to manage, since it is difficult to answer who has access to what. An issue we should not neglect as well is how organizations using cloud computing can ensure compliance with the laws and regulations that they are subject to.

Cloud service providers should be able to assure that tenants' compliance and security policies are consistently managed and enforced. The identities may need to be governed or managed by geographical locations to enforce regional compliance policies. In the same way, every action that affects a resource being governed by a compliance policy must be recorded. The consumers of the cloud are responsible for the security and integrity of their own data, even when it is held by a (cloud) service provider.

In addition to that, industry and government do not always perceive risks in the same way; therefore, solutions are not equally taken into consideration. On the one hand, industry can see risks as business risks, that is, taking security measures into consideration costs money, so the equilibrium point between not taking security measures at all (zero costs, but maximum risk) and covering all the security issues (maximum cost, minimum risks) should be found. On the other hand, government must avoid any kind of risk, since risks could derive in threats for identity, which may affect the national security. Sometimes solving these issues is not a technical question, but rather a legal one. Additionally, governments may establish clear responsibility lines, which is not a trivial issue in cloud environment. Both industry and government should promote security as an integral part of the technology, not as an extra cost.

Standards can help to address these baseline issues, and could establish the basis for systematic assessment of identity management requirements for cloud systems. They separate technical aspects from legal aspects, describing taxonomy of categories that allows an easier understanding by vendors and customers alike, entailing good practice around category requirements. Common categories emerge from regulations across geographies. Within these categories, regional and national governments have their own identity requirements. However, from regulators' point of view, there are so many standards development, so they do not know where to focus their efforts. They also see a lack of appropriate expertise.

Creating standards or specifications is not an easy task. Currently, we see a lot of concurrent specifications with the same objective, for instance securing assessing, certifying and accrediting cloud solutions. As a result, it is difficult not only to know what we should use, but also some of these specifications contradict each other. Consequently, we cannot follow all of them simultaneously, and it could be a nightmare to try to map these specifications.

Another point to take into account regarding standardization is the implementation of these standards. Usually, standards define how the outcome looks like, but no how to reach it. Therefore, sometimes they are difficult to develop. One of the objectives is also to provide a standards-based framework that will help business process engineers, IT analysts, architects, and developers to implement privacy and security policies in their operations. The PMRM (Privacy Management Reference Model)

[42] provides a guideline for developing operational solutions to privacy issues. It also serves as an analytical tool for assessing the completeness of proposed solutions and as the basis for establishing categories and groupings of privacy management controls.

Cloud computing has to consider current technologies too in order to benefit from them. PKI infrastructures have been used in several solutions until now, and it is a model which has been well accepted for authentication and adapted to different requirements. In the same way, this model is desirable in cloud computing. Organizations, governments and citizens will be eager to use their identities for authentication [58]. Actually, many states are deploying national electronic IDs, so the objective now is looking for ways to leverage existing ID infrastructures.

Authentication (you are who you say you are) is not enough to achieve emerging objectives, so it has been complemented with authorization (you are allowed or not to perform a certain action). XACML [43] is a standard for defining access control policies, which allows fine granularity. Furthermore, it defines a protocol to query for authorization decisions, which allows externalizing authorization management.

The distributed nature of cloud computing permits multiple Identity Providers authentication and authorization services, each of one using different identity credentials, representation and formats, and all of this should be integrated in the same "cloud" concept.

7 Future Research Directions

This section will extract the current research challenges to be addressed in order to reach the aforementioned foreseen practical scenarios. Many of the challenges that cloud computing brings are already handled, such as virtualization [14], federated identity [39], remote data storage accesses [31], etc. But others have not been considered so much yet.

There are still so many unresolved questions regarding data ownership and its access. Cloud computing relieves management of distributed data among different organizations, domains or even countries. In this sense, users' privacy, personal data management and users' rights become more important and they should be carefully considered.

Due to the user-centricity and privacy-preserving features offered by identity management systems, they are becoming a key element in cloud computing environments. Current researches are especially focused on systems based on zero-knowledge proofs, due to the multiple possibilities they may offer. However, as shown in previous sections, some essential features or requirements are still missing in these systems.

Furthermore, on systems based on zero-knowledge proofs, audit and privacy seem to be opposed features. A main goal of identity management systems is to preserve users' privacy by hiding their real identity and interactions, but at the same time they should incorporate effective auditing mechanisms to guarantee that end users cannot deny having performed an operation [59]. Hence, it is a current challenge to define

how providers can discourage attackers by recording their actions without tracing end users operations.

There are on-going works taking into consideration this issue [11]. They make use of advanced cryptography techniques extending the anonymous credentials in such a way that the anonymity could be broken under certain conditions. For instance, the identity of a given end user is hidden unless an external and trustworthy inspector (e.g. a government agent) considers that the end user has had a malicious behavior, like committing fraud.

In the same line, since end users could present assertions without requiring involving the identity providers, ordinary Certificates Revocation Lists (CRLs) cannot be used, making efficient attribute revocation hard to achieve. Although some solutions have been already proposed they tend to be impracticable for large systems, and this issue is still an open research question today.

As shown in Sect. 5.8, attribute aggregation is a requirement not properly fulfilled by any of the analyzed solutions. There are some research documents, like [12], proposing solutions for this challenge. Those solutions are usually based on adding an intermediate element in charge of collecting attributes from different providers, but they require end users to manually link their different accounts somehow. Furthermore, how to integrate this kind of solutions within identity management systems based on zero-knowledge proofs, or how they could be adapted to cloud environment has not been still properly faced.

Deployment of cloud computing depends on the features offered to end users to manage their information. End users would be more reluctant to use cloud computing services if they lose the capability of controlling their private information. In this way, user consent is not only a matter of allowing end users to approve whether to continue or not a transaction, but also on defining transitive permission to third parties to access their information. Moreover, identity management in cloud computing is not only authentication any longer, but it also includes authorization. In this sense, existing authorization mechanisms have to be improved or adapted to take into account the dynamicity of cloud computing.

The main difference between traditional systems and cloud computing is that confidentiality based on encryption is hardly possible, and the inexistent user control on the physical level. This affects directly to approaches based on electronic Identity Cards (eID), which now must be “cloud compatible”. For instance, how can cloud fit in projects like STORK [52], which consists of an interoperability framework on top of national eID infrastructure, it is a question that still needs to be answered.

Additionally, identity management systems are usually based on trust relationships between entities or domains. Trust management is an important aspect of identity management, since it defines security boundaries. In this sense, not only service providers could validate end users’ attributes, but also end users (or identity providers acting on their behalf) could determine whether the receiver of their personal information is trustworthy.

However, due to the heterogeneity and dynamicity of cloud computing, trust relationships based on strong contracts, such as SLAs (Service-Level Agreement) establishment, is no longer an option for many scenarios, and more adaptable methods

need to be applied. In this regard, reputation systems propose an efficient alternative to handle this issue, although they need to take into account the risk and threats raised by cloud computing. Reputation systems make use of past experiences to calculate a level of trust for a given service, determining whether the service is asserted to be reliable.

8 Conclusion

Identity management systems have been proved to be secure and efficient in diverse contexts and scenarios. By establishing trust relationships between providers and domains, identity management systems offer a huge range of features both for end users and for organizations regarding controlling and exchanging identity-related information in a privacy-aware way.

Due to the user-centricity and privacy-preserving features offered by identity management systems, they are becoming a key element in cloud computing environments. Cloud computing integrates technologies and concepts from other fields, such as multi party computation, distributed systems, federation, etc. Therefore, some of the raised challenges have been already tackled in other contexts, where identity management systems have been widely accepted.

Nevertheless, cloud computing brings a different perspective related to the end users interests, which results on new risks for end users identities. Additionally, end users are more concerned about how their data is managed, where it is located and who can access it. In this sense, cloud computing is changing some of the basic assumptions.

In this document we extract essential requirements as a result of analyzing different use cases and scenarios related to the cloud. As main contribution we have presented representative identity management standards, technologies and approaches in order to highlight the benefits and drawbacks of each of them with regard to the previously presented requirements. An analysis and comparison have been conducted to describe how each of these systems fits in a cloud computing environment.

We have shown that there is not an ideal approach fulfilling all the requirements, which emphasizes all the unsolved questions these systems leave. In this way, selecting the most appropriate solution depends on the features of the scenario and the desired behavior.

In general, this document draws the envisioned practical and realistic scenarios of application of advanced identity management techniques within the scope of cloud computing. Finally, current research challenges to be addressed and ongoing work are presented, together with a description of working groups, standardization activities and international projects aimed to identify, analyze, and describe identity management challenges and pending issues.

References

1. ABC4Trust. Attribute-based credentials for trust. European union funded project of the 7th framework programme. [Online]. Available: https://abc4trust.eu/
2. Alrodhan WA, Mitchell CJ (2007) Addressing privacy issues in CardSpace. In: Proceedings of the 3rd international symposium on information assurance and security (IAS '07), Manchester, UK, pp 285–291
3. Ates M, Buccafurri F, Fayolle J, Lax G (2012) A warning on how to implement anonymous credential protocols into the information card framework. Int J Inf Secur 11(1):33–40
4. Bertocci V, Serack G, Baker C (2008) Understanding windows CardSpace: an introduction to the concepts and challenges of digital identities. Addison-Wesley, Reading
5. Bogdanov D, Niitsoo M, Toft T, Willemson J (2012) High-performance secure multi-party computation for data mining applications. Int J Inf Secur 11(6):403–418
6. Brands S (2000) Rethinking public key infrastructures and digital certificates: building in privacy. MIT Press, Cambridge
7. Brands S, Demuynck L, De Decker B (2007) A practical system for globally revoking the unlinkable pseudonyms of unknown users. In: Proceedings of the 12th Australasian conference on information security and privacy, ACISP'07. Springer
8. Callegati F, Cerroni W, Ramilli M (2009) Man-in-the-middle attack to the HTTPS protocol. IEEE Secur Priv 7(1):78–81
9. Camenisch J, Lysyanskaya A (2001) An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Birgit Pfitzmann (ed) Proceedings of the international conference on the theory and application of cryptographic techniques: advances in cryptology (EUROCRYPT '01), Springer-Verlag, London, UK, pp 93–118
10. Camenisch J, Van Herreweghen E (2002) Design and implementation of the idemix anonymous credential system. In: Proceedings of the 9th ACM conference on computer and communications, security
11. Camenisch J, Krontiris I, Lehmann A, Neven G, Paquin C, Rannenberg K, Zwingelberg H (2011) D2.1 architecture for attribute-based credential technologies. Deliverable of ABC4Trust European project
12. Chadwick DW, Inman G (2009) Attribute aggregation in federated identity management. IEEE Comput Soc 42(5):33–40
13. Chappell D (2006) Introducing windows CardSpace. MSDN, Available: http://msdn.microsoft.com/en-us/library/aa480189.aspx
14. Christodorescu M, Sailer R, Schales DL, Sgandurra D, Zamboni D (2009) Cloud security is not (just) virtualization security: a short paper. In: Proceedings of the 2009 ACM workshop on cloud computing security (CCSW '09), ACM, New York, NY, USA, pp 97–102
15. Clercq JD (2002) Single sign-on architectures. In InfraSec '02: proceedings of the international conference on infrastructure security, Springer, Bristol, UK, pp 40–58
16. van Delft B, Oostdijk M (2010) A security analysis of OpenID. Policies Res Identity Manag 343:73–84
17. Dólera Tormo G, Gómez Mármol F, Martínez Pérez G (2012) On the application of trust and reputation management and user-centric techniques for identity management systems. XII Spanish meeting on cryptology and information security (RECSI 2012), San Sebastián, Spain
18. Dólera Tormo G, López Millán G, Martínez Pérez G (2013) Definition of an advanced identity management infrastructure. Int J Inf Secur 12(3):173–200
19. Eclipse.org, Higgins 2.0 Personal Data Service. [Online]. Available: http://www.eclipse.org/higgins/
20. Erdos M, Cantor S (2002) Shibboleth architecture DRAFT v05. [Online]. Available: http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-arch-v05.pdf
21. Gajek S, Schwenk J, Steiner M, Xuan C (2009) Risks of the CardSpace protocol. Lect Notes Comput Sci 5735:278–293
22. Goldreich O, Micali S, Wigderson A (1991) Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J ACM (JACM) 38(3):690–728

23. Goldwasser S, Micali S, Rackoff C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18(1):186–208
24. Mármol Gómez F, Martínez Pérez G (2009) Security threats scenarios in trust and reputation models for distributed systems. Comput Secur 28(7):545–556
25. Mármol Gómez F, Girao J, Martínez Pérez G (2010) TRIMS, a privacy-aware trust and reputation model for identity management systems. Comput Netw 54(16):2899–2912
26. Gómez Mármol F, Kuhnen M, Martínez Pérez G (2011) Enhancing OpenID through a reputation framework. In: Proceedings of the 8th international conference on autonomic and trusted, computing ATC11, p 118
27. Hammer-Lahav, E. and Recordon, D., "The OAuth 1.0 Protocol", Internet Engineering Task Force (IETF) RFC 5849, 2010.
28. Harding P, Madsen P, Drake TC, Mortimore C (2012) System for cross-domain identity management: core schema. Internet Draft. draft-ietf-scim-core-schema-00 (SCIM)
29. Hardt D (ed) (2012) The OAuth 2.0 authorization framework. Technical report, IETF. Available: http://tools.ietf.org/html/draft-ietf-oauth-v2-31
30. Herranz J, Iñigo J, Pujol H (2009) Privacy features of authentication systems. In: Proceeding of the first workshop on law and web 2.0, Barcelona, Spain. pp 35–46
31. Hoschek W, Jaen-Martinez J, Samar A, Stockinger H, Stockinger K (2000) Data management in an international data grid project. Lect Notes Comput Sci 1971:77–90
32. IBM Research, Zurich (2010) Specification of the identity mixer cryptographic library
33. Identity Commons. [Online]. Available: http://www.identitycommons.net/
34. Jagatic TN, Johnson NA, Jakobsson M, Menczer F (2007) Social phishing. Commun ACM 50:94–100
35. OASIS Standard (2009) Identity Metasystem Interoperability Version 1.0 (IMI 1.0). Available: http://docs.oasis-open.org/imi/identity/v1.0/identity.html
36. Kantara Initiative. [Online]. Available: http://kantarainitiative.org/
37. Kolšek M (2002) Session fixation vulnerability in web-based applications. ACROS security, Available: http://www.acrossecurity.com/papers/session_fixation.pdf
38. Kontaxis G, Polychronakis M, Markatos EP (2012) Minimizing information disclosure to third parties in social login platforms. Int J Inf Secur 11(5):321–332
39. Maler E, Reed D (2008) The venn of identity: options and issues in federated identity management. IEEE Secur Priv 6:16–23
40. Nanda A, Jones MB (2008) Identity selector interoperability profile v1.5. Microsoft Corp. Available: http://download.microsoft.com/download/1/1/a/11ac6505-e4c0-4e05-987c-6f1d31855cd2/Identity_Selector_Interoperability_Profile_V1.5.pdf
41. OASIS IDCloud TC. OASIS identity in the cloud TC. [Online]. Available: http://wiki.oasis-open.org/id-cloud/
42. OASIS Privacy Management Reference Model (PMRM) TC [Online]. Available: http://www.oasis-open.org/committees/pmrm
43. OASIS Standard. eXtensible access control markup language TC v2.0 (XACML) (2005) Available: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf
44. OASIS Standard: assertions and protocols for the OASIS security assertion markup language (SAML) version 2.0 (2005).
45. OAuth Community. [Online]. Available: http://oauth.net/community/
46. OpenID Community. [Online]. Available: http://openid.net/community/
47. Paquin C, Thompson G (2010) U-prove CTP white paper. Microsoft Tech Rep
48. Pearson S, Benameur A (2010) Privacy, security and trust issues arising from cloud computing. In: Proceedings of the second international conference on cloud computing technology and science (CloudCom), Bristol, UK, pp 693–702
49. PrimeLife. European union funded project of the 7th framework programme. [Online]. Available: http://primelife.ercim.eu/
50. Recordon D, Drummond R (2006) OpenID 2.0: a platform for user-centric identity management. In: Proceedings of the second ACM workshop on digital identity management, Alexandria, VA, USA, pp 11–16

51. Saldhana A, Nadalin A, Rutkowski M (2012) Identity in the cloud use cases version 1.0. Available: http://docs.oasis-open.org/id-cloud/IDCloud-usecases/v1.0/cn01/IDCloud-usecases-v1.0-cn01.html
52. STORK (Secure idenTity acrOss boRders linKed), European Union funded project of the 7th framework programme. [Online]. Available: https://www.eid-stork.eu/
53. SWIFT. Secure widespread identities for federated telecommunications. European Union funded project of the 7th framework programme. [Online]. Available: http://www.ist-swift.org/
54. Tanenbaum AS, Van Steen M (2001) Distributed systems: principles and paradigms. Prentice Hall, Upper Saddle River, NJ
55. The White House. National strategy for trusted identities in cyberspace (NSTIC). [Online]. Available: http://www.nist.gov/
56. Trevithick P. Relationship cards. Higgins report, 19 Sept 2009. Available: http://www.eclipse.org/higgins/documents/relationship-cards.html
57. U-Prove: Microsoft Corporation Technology (2010) [Online]. Available: http://www.microsoft.com/u-prove
58. Ustaoğlu B (2011) Integrating identity-based and certificate-based authenticated key exchange protocols. Int J Inf Secur 10(4):201–212
59. Wang C, Wang Q, Ren K, Lou W (2010) Privacy-preserving public auditing for data storage security in cloud computing. In: Proceedings of the 29th conference on information communications (INFOCOM'10). IEEE Press, Piscataway, pp 525–533
60. Web Identity Working group. [Online]. Available: http://www.w3.org/2011/08/webidentity-charter.html

Data Accountability in Cloud Systems

Ryan K. L. Ko

1 Introduction

This chapter reviews the definitions, existing techniques and standards in the area of data accountability in cloud computing. It also introduces new research for the accountability, traceability and auditability of data provenance and history and discusses the critical problems of cloud security relating to accountability.

In a recent survey by Fujitsu Research Institute [17], it was shown that 88 % of cloud computing customers surveyed want to know "what goes on behind the scenes" in clouds. This finding reveals how (1) cloud customers are unhappy with the current level/lack of accountability and transparency in cloud computing environments, and (2) the relative lack of solutions which enable cloud providers and users to have an awareness of "*who has touched their data, at where and at what time*", whether "*their data has left a desired boundary/ country*", or even, "*how many copies of their files are present in the cloud at a particular time*".

These questions also expose the relative lack of solutions and techniques for tracking the derivation history and life cycle of data in clouds [26, 57]. Solving these holy grails will empower one to truly achieve end-to-end tracking of data in any distributed, virtualized environments. Such solutions will also enable cloud providers to ensure high accountability, auditability and trust.

The nature of cloud computing requires users to deposit their information, sometimes critical information, into systems which they may not own or have control over. This inevitably results in a trust relationship tension between the user and the cloud service providers (CSPs). The users, who are the owners of the data, will have concerns over what is happening to their data behind the scenes.

However, since we have no technology to clearly separate the boundaries of data processing requirements from data ownership and control, we are at this moment,

R. K. L. Ko (✉)
Cyber Security Lab, Department of Computer Science,The University of Waikato, Private Bag 3105, 3240 Hamilton, New Zealand
e-mail: ryan@waikato.ac.nz

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_7,

faced with a very daunting situation: the lack of the ability to achieve accountability of our data in the cloud.

1.1 What is Data Accountability in the Cloud?

From a system design perspective, the notion of trust can be increased via reducing risk when using the cloud. While risk can be greatly mitigated via privacy protection and security measures such as encryption, they are not enough, particularly as full encryption of data in the cloud is at present not a practical solution. There is a need to complement such ***preventative controls*** with equally important ***detective controls*** that promote transparency, governance and accountability of the service providers.

Despite accountability being a crucial component of improving trust and confidence [34, 37], current prominent providers (*e.g. Amazon EC2/ S3* [1, 18], *Microsoft Azure* [8]) are still not providing full transparency or capabilities for the tracking and auditing of the file access history and data provenance [4] of both the physical and virtual servers utilized [17]. Currently, users can at best monitor the virtual hardware performance metrics and system event logs of the services in which they engage.

The cloud computing research community, particularly the Cloud Security Alliance, has recognized this. In its *Top Threats to Cloud Computing Report* [10], it listed seven top threats to cloud computing:

1. *Abuse and nefarious use of cloud computing*
2. *Insecure application programming interfaces*
3. *Malicious insiders*
4. *Shared technology vulnerabilities*
5. *Data loss or leakages*
6. *Account, service and traffic hijacking*
7. *Unknown risk profile.*

Methods increasing the accountability and auditability of cloud service providers, such as tracing of file access histories, will allow service providers and users to reduce five of the above seven threats: 1, 2, 3, 5 and 7. This chapter focuses on such traceability of data, so that one day, we can achieve true accountability of data in clouds.

1.2 Key Concepts

Let us begin by defining the bigger picture: *trust* in the cloud.

In dictionaries, *Trust* is generally related to "*levels of confidence in something or someone*" [33, 55]. Hence, in cloud computing, we can view trust in the cloud as the customers' level of confidence in using the cloud. While this 'level of confidence'

is subjective, researchers should try to 'increase confidence' by mitigating technical and psychological barriers to confidence in the cloud.

1.2.1 Components of Trust in Cloud Computing

Before mitigating barriers to confidence, we need to understand the main components affecting cloud *trust*:

Security [3, 53]—*Mechanisms* (*e.g. encryption*) *which make it extremely difficult or uneconomical for an unauthorised person to access some sensitive information.*

Privacy [34, 37]—*Protection against the exposure or leakage of sensitive data* (*e.g. personally identifiable information* (*PII*)).

Accountability [19, 34]—*Ensuring that all actors and actions performed on the information are being 'accounted for', i.e. recorded as evidence.*

Auditability [1]—*The relative ease of auditing a system or an environment. Poor auditability means that the system has either an absence of, or poorly-maintained logs and systems that enable efficient auditing of processes within the cloud. Auditability is also an enabler of accountability. In simpler words, auditability ensures events are "loggable" while accountability ensures that events deemed important are logged and not missed.*

1.2.2 Preventive Versus Deterrent Measures

Trust components can be also grouped as ***Preventive Measures*** or ***Deterrent Measures***. *Security* and *Privacy* are preventive measures, while *Accountability* and *Auditability* act as deterrent measures. This chapter focuses on the Deterrent measures. Despite the lack of direct ability to stop irregularities from occuring, deterrent measures are very important.

Deterrent measures act as not only psychological obstacles to commit crime in the cloud, and also serves as records for post-mortem investigations should any crime occur. Deterrent measures hence complement preventive measures; both cannot be applied as standalones without consideration of the other.

2 A Trust-Related Data Accountability Scenario Users Fear

Figure 1 shows a typical trust-related scenario which many potential cloud customers **fear** [17]. A customer stores some sensitive data in a file (*see* Fig. 1 *top-left; red icon*) within a virtual machine (VM) hosted by a provider s/he has subscribed to. Upon uploading the data, failsafe mechanisms within the cloud will typically back it up, and perform load balancing by creating redundancies across several virtual servers and physical servers in the service provider's trusted domain.

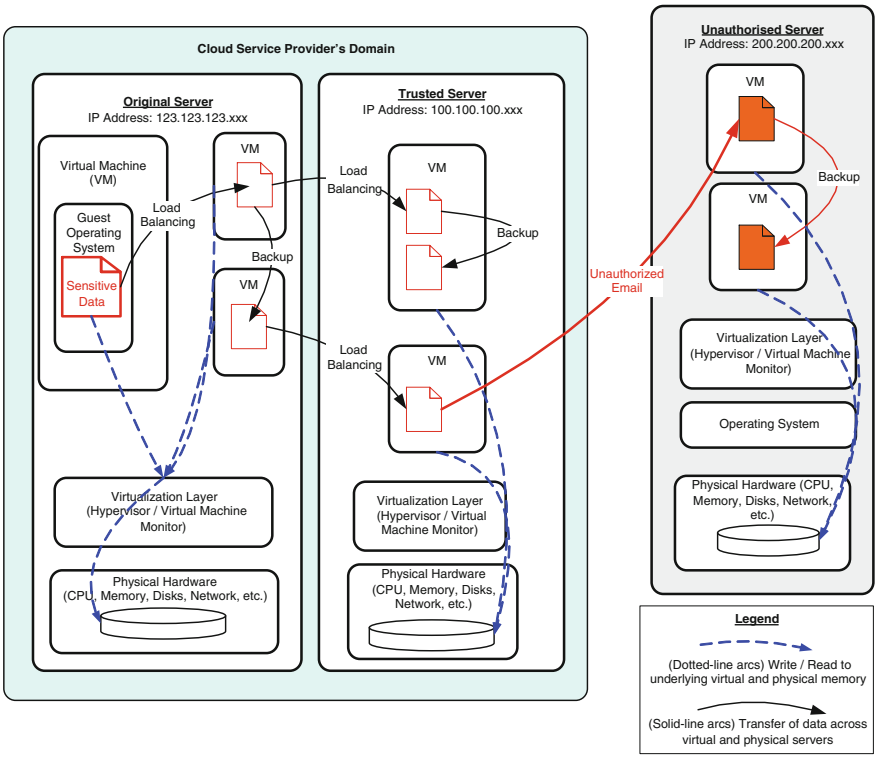


Fig. 1 An example scenario in cloud computing, showing the importance of accountability and auditability

From the file's creation to the backup processes, large numbers of data transfers occur across virtual and physical servers (*solid-line arcs*; Fig. 1), and several memory read/write transactions to both virtual and physical memories are involved (*dotted-line arcs*; Fig. 1).

If all such transactions and the creation of new duplicate files are logged, monitored and accounted for, we would be able to trace the file history and log the access history and content modifications, i.e. achieving cloud accountability and auditability.

Even if a malicious insider of the CSP attempts to transfer the sensitive file/ data to a target outside the cloud (*e.g. in* Fig. 1, *'via email'*), we will be well-equipped to know when, where, how and what was being leaked, and by whom. This empowers **both** the CSP and the consumers, as problematic processes and even insider jobs may be investigated. This also removes some barriers to confidence in the cloud.

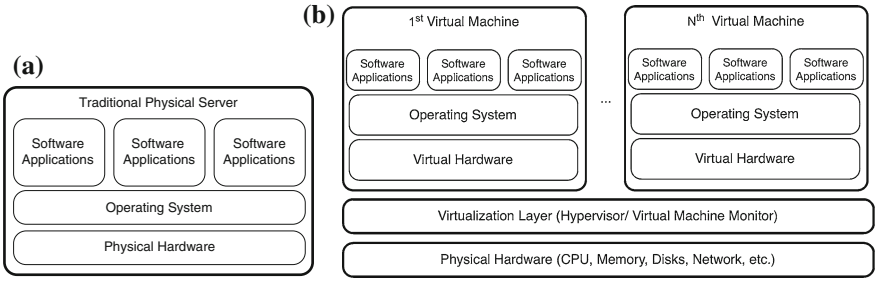


Fig. 2 Logical components of (**a**) traditional physical server environments versus (**b**) confederated virtualised & physical server environments in clouds

3 Complexities Introduced by Cloud's Elasticity

However, with cloud computing's promise of elasticity empowered by virtualization [1, 2], comes several new complexities introduced into the area of accountability.

3.1 Challenges Introduced by Virtualisation

3.1.1 Tracking of Virtual-to-Physical Mapping and Vice Versa

As shown in Fig. 2, clouds are mostly configured like that of Fig. 2b, as compared to the traditional physical server environments typically found within companies (Fig. 2a). With the introduction of virtualization, precious server resources can be utilized more efficiently as they now host more servers, and are able to adapt to peaks and troughs in usage and bandwidth requirements. However, the addition of virtualized layers also means that we would not only need to track the events in each individual virtual server, but also in the physical servers.

Currently, there are only tools (e.g. HyTrust [23]) which are able to log virtual level logs and system health monitoring tools for virtual machines. There is still a lack of transparency of (1) the linkages between the virtual and physical operating systems (OS), (2) relationships between virtual locations and physical static server locations, and (3) how the files are written into both virtual and physical memory addresses. These information are currently not available as a single-point-of-view for the customers. In a recent survey by Fujitsu Research Institute [17], it was mentioned that a whopping 88% potential cloud consumers are worried about who has access to their data, and would like to have more awareness of what "goes on" in the backend physical server. Such surveys have only enhanced the urgency for practitioners and researchers to quickly address such obstacles to trust.

3.1.2 Multiple Operating System Environments

With the ease of choosing a myriad of operating systems for virtual machines, comes the complexity of managing the logging of a very large possibility of operating systems within the cloud. Enforcing a single operating system for all virtual machines would solve this issue, but it would make the provider less competitive.

This means that we cannot just focus on system health logging and existing operating system based logging tools [44], but need a new perspective of logging, as explained in the following section.

3.1.3 Logging from Operating System Perspective Versus Logging from File System Perspective

Current tools focus on operating systems and system health monitoring (e.g. cloudstatus.com, [22], etc), but few emphasize on file-system perspective. By the file system perspective, we mean that we need to trace data and files from the time they are created to the time they are destroyed.

When we log from a file system perspective, [28] we view data and information independent from the environmental constraints. This is reflective of the very elastic nature of cloud computing, and with the transfer of control of data into the providers, the providers have the mandate to ease the minds of consumers by providing them the capabilities to track their data just like that in Fig. 1.

3.2 Scale, Scope and Size of Logging

The elasticity concept also increases the need for efficient logging techniques and a proper definition of scope and scale of logging. By efficient, we mean that the impending exponential increase in log size has to be manageable, and not quickly wipe out memory of servers hosting the cloud logging features.

By scale and scope, we need policies that can help to clearly define the areas which loggers are assigned to log in. For example, a service provider may label its own network as a safe zone, while its suppliers or mirror sites trusted zones, and any other network outside of these are labeled as unsafe zones. Zonal planning will greatly reduce the complexities of network data transfer tracing within a cloud. Another way of reducing complexity will be the classification of the level of data abstraction, e.g. crude data, documents, and on a higher level, workflows.

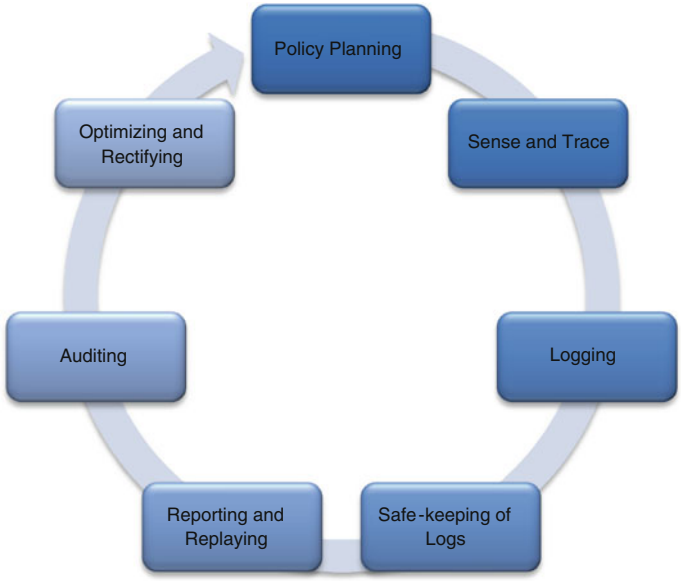


Fig. 3 The cloud data accountability life cycle

4 The Cloud Data Accountability Life Cycle

The discussions earlier and the scenario in Fig. 1 have not only revealed the scale and urgency of the problem but also exposed the need for reduction of complexity. Having an awareness of the key accountability phases will not only simplify the problem, but also allow toolmakers to gauge the comprehensiveness of their tool (i.e. if there are any phases not covered by their product). Phases can help researchers focus on specific research sub-problems of the large cloud accountability problem. Consumers can also understand if the cloud accountability tool has a real coverage of all phases of cloud accountability. These phases are collectively known as the **Cloud Accountability Life Cycle** (CALC). We propose CALC as the following seven phases (see Fig. 3):

- ***Policy Planning***
  In the beginning, CSPs have to decide what information to log and which events to log on-the-fly. It is not the focus of this chapter to claim or provide an exhaustive list of recommended data to be logged. However, in our observation, there are generally four important groups of data that must be logged: (1) Event data—a sequence of activities and relevant information, (2) Actor Data—the person or computer component (e.g. worm) which trigger the event, (3) Timestamp Data—the time and date the event took place, and (4) Location Data—both virtual and physical (network, memory, etc) server addresses at which the event took place.
- ***Sense and Trace***

The main aim of this phase is to act as a sensor and to trigger logging whenever an expected phenomenon occurs in the CSP's cloud (in real time). Accountability tools need to be able to track from the lowest-level system read/write calls all the way to the irregularities of high-level workflows hosted in virtual machines in disparate physical servers and locations. Also, there is a need to trace the routes of the network packets within the cloud.

- ***Logging***
  File-centric perspective logging is performed on **both** virtual and physical layers in the cloud. Considerations include the lifespan of the logs within the cloud, the detail of data to be logged and the location of storage of the logs.
- ***Safe-keeping of Logs***
  After logging is done, we need to protect the integrity of the logs prevent unauthorized access and ensure that they are tamper-free. Encryption may be applied to protect the logs. There should also be mechanisms to ensure proper backing up of logs and prevent loss or corruption of logs. Pseudonymisation of sensitive data within the logs may in some cases be appropriate.
- ***Reporting and Replaying***
  Reporting tools generate from logs file-centric summaries and reports of the audit trails, access history of files and the life cycle of files in the cloud. Suspected irregularities are also flagged to the end-user. Reports cover a large scope: virtual and physical server histories within the cloud; from OS-level read/write operations of sensitive data to high-level workflow audit trails.
- ***Auditing***
  Logs and reports are checked and potential fraud-causing loopholes highlighted. The checking can be performed by auditors or stakeholders. If automated, the process of auditing will become 'enforcement'. Automated enforcement is very feasible for the massive cloud environment, enabling cloud system administrators to detect irregularities more efficiently.
- ***Optimising and Rectifying*** Problem areas and security loopholes in the cloud are removed or rectified and control and governance of the cloud processes are improved.

5 Cloud Data Accountability Layers

Next we address the important question: *what data to log?* The answer ranges from a system-level log to a workflow-level audit trail transactional log. Such a range shows that there are many abstraction layers of data, and a framework is needed to reduce this kind of ambiguity and increase research focus and impact. As such, we propose the following **layers of accountability in a cloud** [26]:

Figure 4 shows the abstraction layers for the type of logs needed for an accountable cloud.

It is important to note that the focus is on the abstraction layers of logs and not on architectural layers. Hence, it is independent of virtual or physical environments. The data and workflow abstraction layers are derived from related works in data and

workflow provenance [4, 5, 49], and the system layer is derived from related works in trusted computing platforms [35, 38] and system logging literature [21, 40].

Such explicit definition of layers in Fig. 4 allows us to efficiently identify the areas of their application and their focus areas. At a glance, the three layers look deceptively simple, but the problem is more complex than it looks. Each layer has a slightly different set of sub-components for each different context. Our model simplifies the problem and makes accountability more achievable. The usefulness of layers is also analogous to OSI. [58] and TCP/IP [48] networking layers.

Let us now discuss the scope and scale of each layer:

5.1 System Layer

The foundation layer is the system layer. The system layer performs file-centric logging within the following three components:

5.1.1 Operating Systems

Operating Systems (OS) system and event logs are the most common type of logs associated with cloud computing at the moment. However, these logs are not the main contributing factor to accountability of **data** in the cloud, but a supporting factor. This is because in traditional physical server environments housed within companies, the emphasis was on health and feedback on system status and ensuring uptime as server resources are limited and expensive to maintain. In cloud computing, resources like servers and memory are '*elastic*', and are no longer limited or expensive [1, 2]. Hence, OS logs, while important, are no longer the top concern of customers. Instead, the customers are more concerned about the integrity, security and management of their data stored in the cloud [9, 17].

5.1.2 File Systems

Even though the file system is technically part of the OS, we explicitly include it as a major component in a file-centric system layer. This is because, in order to know, trace and record the exact file life cycles, we often have to track system read/write

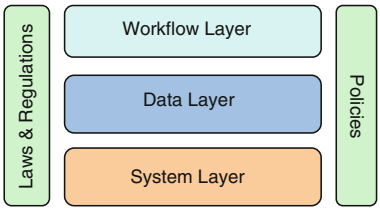


Fig. 4 Abstraction layers of accountability in the cloud

calls to the file system. From the system read/write calls, we can also extract the files' virtual and physical memory locations, providing more information for further forensics. The file-centric perspective [41] is also the area which is less emphasized by current tools.

5.1.3 Cloud's Internal Network

As clouds are vast networks of physical and virtual servers over a large number of locations, we need to also monitor network logs within the cloud. Network logs [46, 47] are logs specific to data being sent and received over the network.

5.1.4 Why System Layer?

One of the key problems of cloud computing environment is the "replay" of a snapshot, i.e. a reproduction of the exact state of the cloud at a particular moment and the machines turned on and off at that instance. With a large number of virtual machines turned on and off at different time periods, and executing several business applications at the same time, it is very difficult to replay the exact same snapshot of the Cloud from the past, e.g. 1 hour ago, so that one can track what actually went wrong [43]. There needs to be an effective and efficient method to do this, and our current work on a file-centric system layer logging mechanism across both virtual machines and physical machines (PMs) fits into the role very well. Such system-layer mechanisms log the resources the VMs use and share when they are turned on. Evidently, such snapshots cannot be captured in the data and workflow layer as they are too high-level and dependent on the on and off status of their hosting machines. The only assured way to track the complete VM changes is actually to track the system layer of the Cloud.

5.2 Data Layer

The data layer supports the data abstraction and facilitates data-centric logging through the following components:

5.2.1 Provenance Logger

In order to enable reasoning about the origins, collection or creation, evolution, and use of data, it is essential to track the history of data, i.e. its provenance. Provenance information constitutes '*the foundation for any reasonable model of privacy and trust*' [20]. It enables validation of the processes involved in generating/obtaining the data; detection of unusual behavior, e.g. whether data was exposed to faulty hardware

or software; formulation of better optimization techniques for data clustering, data evaluation, data caching and data pre-fetching; capturing of temporal and context-related properties; and more.

While these advantages are very promising, corresponding challenges are equally difficult to address/overcome. Common challenges include efficiently and effectively managing the sheer amount of provenance data that has to be maintained; ensuring consistency and completeness of provenance data; detecting malicious users who attempt to falsify provenance data; protecting data owner as well as data providers from exposing sensitive, confidential, proprietary or competitively important information indirectly through provenance logs; enabling efficient querying of provenance data; etc.

The cloud computing domain brings more complexities that require new or evolved solutions. Considering past and current efforts, cloud computing-based provenance logging must fulfill the following criteria: (1) be secure and privacy-aware (to avoid that the logs themselves cannot be tempered with nor are a source for knowledge inference); (2) be (eventually) consistent and complete (similar to the ACID properties known from database transaction processing); (3) be transparent/non-invasive; (4) be scalable; (5) avoid exponential explosion of provenance data (e.g. through the use of summarization techniques); (6) be long-term persistent; (7) allow for multiple tailored views (to permit access based on roles with different access privileges); and (8) be efficiently accessible.

5.2.2 Consistency Logger

While current cloud providers typically support a lesser notion of consistency, i.e. eventual consistency, it is vital to have mechanism to allow for rollback, recovery, replay, backup, and restore features. Such functionality is usually enabled on the basis of operational and/or transactional logs, which assist with ensuring atomicity, consistency, and durability properties. Logs have also been proven useful for media recovery, (time and context-independent) replay, and monitoring of operational anomalies. While these concepts are well established in the database domain, cloud computing's characteristics such as eventual consistency, 'unlimited' scale, and multi-tenancy pose renewed challenges. In addition, secure and privacy-aware mechanisms must be devised not only for consistency logs but also for their backups, which are commonly used for media recovery.

5.3 Workflow Layer

The workflow layer focuses on the audit trails and the audit-related data found in the software services in the cloud. High-level fraudulent risks such as procurement approval routes, decision-making flows and role management in software services run within the cloud has to be monitored and controlled. In a service oriented architecture

[16], services from several sources are composed to perform higher-level, more complex business functions. The accountability of the services and their providers within the clouds also have to be managed.

The workflow layer has to ensure proper governance of cloud applications, empower continuous auditing and manage the accountability of services composed as business processes via the following components:

5.3.1 Governance in the Cloud

When cloud computing experiences an increase in uptake and usage, there will be mandated needs for the auditability, proper prevention and tracking of fraudulent activities, irregularities and control loopholes in the business processes in the cloud. The workflow layer explores how clouds can achieve high auditability via compliance to regulations such as Sarbanes-Oxley (SOX) [42] and Health and Human Services Health Insurance Portability and Accountability Act (HIPAA) (e.g. Title II: Preventing Healthcare Fraud and Abuse) regulations [51], and/ or benchmarking against information security standards such as the ISO 27000 suite [6, 7].

5.3.2 Automated Auditing

With the promise of high performance computing power from cloud architectures, the author envisions the realization of automated auditing of financial and business process transactions in the cloud. Auditability is a prerequisite for such a step. However, achieving auditability via methods such as continuous auditing [39] within a highly virtualized environment is a very difficult and complex task. There needs to be considerations for not only the auditing of the business logic and control flows, but also the applications implementing them.

5.3.3 Patch Management Auditing

There is even a need for auditing of the management of virtual machine image bug fixes, patching and upgrades in a cloud environment [32, 54]. The scale of logging patching and deployment within the cloud environment is massive, and needs to be highly auditable for proper troubleshooting, playbacks and accountability of the technical staff performing these activities.

5.3.4 Accountability of Services

When composing services from existing service components, we also face the problem of trust. With cloud computing, service components are proliferated and their access is virtualized. This makes composition easier and practical. Meanwhile, the

source of services may or may not be trustworthy, which presents a major problem in the cloud platform. This can be explained using the following example.

Let us assume that we are developing a Web portal and we are designing this by integration of the services into a portal. Some of the services may be malicious (i.e. from an untrusted zone/ provider or manipulating data passing through). Therefore, the portal may or may not be a valid software and perform according to the expected design or according to the contractual agreement. In this scenario, the achievement of accountability of services can help us to investigate such occasions. We believe that the logging approach is also applicable to help achieve the accountability of services. Logging should take care of the following concerns on a component (e.g. Web service):

1. *Input or pre-processing*, whether the component takes in more than enough input to perform the required function. It is usually a sign of maliciousness if the input is more than what is needed. Additional information from the user may be used to do something not claimed.
2. *Processing*, whether the component is designed to actually do what is expected? Is there any extra and unexpected processing that has occurred in order to produce the requested result?
3. *Post processing*, whether the component has deleted the input and the intermediary results of the processing. Proper actions need to keep the input and the whole processing confidential and no traces of processing should have been recorded.

Our current logging solution achieves the purpose of deterring the service component providers from making malicious components and encourages the proper behavior and execution of the components.

5.4 Policy, Law and Regulations

Policies and laws may require information to be logged on what data items are processed, accessed, stored or transmitted. We are currently addressing this via asking the questions: *why, when, where, how and by whom*.

What: Data classification is important, as in general there will be different policies and legal rules affecting different classes of data items. Classes to consider might include non-PII data, anonymised data, pseudonymised data, PII, sensitive PII, and PCI-regulated data (This last class is determined by the payment card industry). When a new data item is created (either by a user, or as the result of automated copying or processing of already-existing data) this creation may need to be logged together with the classification of the item and/or the policies associated with it. In addition to records about individual data items, there may be audit needs associated with higher-level information. For example policy changes should be recorded, and there may be audit requirements for process flows within and between organizations in the cloud.

Why: The OECD's Purpose Specification and Use Limitation principles legally restrict the use of PII/sensitive PII to purposes compatible with ones specified before or at the time of data collection, required by law, or consented to by the data subject. Therefore, the purpose of a data processing action, and the purposes for which the processing of a given PII data item is permitted, may need to be recorded.

When: Logs usually include timestamps. Timing information is also necessary for compliance to laws and policies concerned with data retention: it is necessary to have a data retention and destruction plan for all data storage systems (Data retention compliance also requires information to be kept on which records or data items are duplicates, and on the location of backup copies, to ensure that all copies of an item can be destroyed). Timing considerations may also reduce the information that needs to be recorded, as transient data that is only stored for the purpose of the current transaction and then deleted has minimal privacy implications.

Where: Geographical location matters from a legal point of view—different laws may apply depending on where information exists, and there are some restrictions on trans-border data flows. It can be difficult to ascertain within the cloud where data is, and there may be multiple copies. So the physical location of storage and the occurrence of cross-border data transfers (for example to partners, sub-contractors, or cloud service providers) may need to be recorded. A related question is "where from?", that is, the source of data. Was PII data collected directly from the data subject or from a third party provider?

How: Some laws and policies restrict how data is handled. For example, the processing of PCI-regulated data may require encryption and other safeguards. Information on how such data has been handled therefore, needs to be recorded for auditability. The OECD's Collection Limitation principle requires PII to be collected with the knowledge of the data subject where appropriate; if it has been collected without the subject's knowledge, this may need to be logged. Similarly, auditability may require the logging of unplanned data disclosures and the reasons for them (internal requests, e-discovery process, compelled by court order, compelled by law enforcement investigation).

Who: Policies may restrict access to a data item to a particular set of authorized users, identified either as individuals or by role. There may also be a need to record the corporate identity of partners or cloud service providers to which data is transmitted, as part of due diligence about cloud service provisioning, and to assist actions required by policies if a provider goes out of business or is acquired, or has a data breach.

6 Technical Approaches to Increasing Accountability

With the definition of CALC and the abstraction layers of the type of data to log, we are primed to create tools and software that will achieve cloud accountability. Currently, we envision **three possible groups of technical approaches**:

6.1 Central Watchdog/ Event Monitoring

In this approach, a watchdog service/ server monitors a certain set of nodes, and watches over the physical and virtual logs of all layers and stores the logs centrally. While this is more economical and easier to maintain, such a watchdog service would undoubtedly be vulnerable to network routing problems, interference or use of false identities.

6.2 Local File Tracking Embedment

In this approach, we envision that a file is designed to dedicate some of its memory for storage of bite-sized local logs and provenance data. Currently, this is very difficult to achieve in current file extensions as they are usually predefined without much consideration of local logging.

6.3 Domain Segregation

Accountability in cloud computing will be more achievable if there is a clear design of different domains from the perspective of CSPs or customers. Internal Zones can depict the CSP's own network, with Trusted Zones for its Collaborators, and External Zones for networks outside these two zones. If the data leaves authorized zones, the event will be flagged.

7 Data Traceability and Accountability Via Data-Centric Logging

7.1 Most Potential: Centralised Management of Data-Centric Logs

From the three groups of possible approaches, the most promising technique is the centralised watchdog/ event monitoring approach. This approach is critical, as it addresses the System Layer of cloud data accountability. With this foundation, we are primed to achieve the other four layers of cloud data accountability.

Before studying the state-of-the-art, it is important to understand more about current technologies. It is important for the reader to note that these technologies are adopted from legacy computing and server architectures, but that does not address the needs of data accountability direct.

7.2 Current Technologies and Their Loopholes

7.2.1 User Space File System Call Monitor Overlooking the Kernel Space

In traditional one-system or local area network (LAN) environments, it is common to find user-space file monitoring tools (e.g. iNotify [29], swatch [21], FAM [45]) to be widely used for monitoring the single- or multiple-file activities within a single machine. Tools are also available for monitoring the transfer of packets in networks (e.g. snort [40]). With large scales and heavy usage of virtualization technologies in cloud computing, such tools are insufficient to provide an over-arching view for monitoring files across both virtual machines (VMs) and physical machines (PMs). Moreover, these applications are usually housed within the user space, leaving them vulnerable to user space attacks.

7.2.2 File Integrity Checkers/ Intrusion Detectors Not Recording Provenance

File integrity checkers such as TripWire [24] inspect for changes to the files in the systems by checking against a baseline hash-key database which is regularly updated with the latest hash keys of the files within a system. Such an implementation is not scalable for the cloud as there is a high volume of access, i.e. the need to regularly update the key database is not feasible. Furthermore, these tools do not provide a history of the file changes. Hence, while they are able to identify which files have changed, they are unable to explain the history of what actually happened to the files. Such limitation is not desirable for forensics in the context of the Cloud.

7.2.3 Virtual Environment Monitors Only Monitoring Server Utilisation and Health

With the rise in adoption of virtualization technologies especially in private clouds, software such as the HyTrust Appliance [23] are starting to become more prominent. These tools enable administrators to regulate the access rights and to have an overview of the activities and consolidation of common system logs for all virtual machines. However, this visibility of the virtual layer is still not the full transparency requested by end-users [17] surveyed by the Fujitsu Research Institute (recall Introduction).

When there is mention of monitoring, there is a current emphasis of monitoring the server performance in Clouds. Such a focus on system monitoring is not totally aligned to the actual needs of users. Despite having color schemes, visualizations and attractive dashboards, tools such as VMWare vFabric Hyperic [52] and CloudKick [15] are still unable to offer the crucial need of monitoring data movements and transfers in the Cloud.

7.3 New Breed of Loggers Required

It is now evident from observing the limitations of the state-of-the-art that we need the following *necessary requirements* for effective monitoring of data in the Cloud [26]:

- *Transcend VM/ PM*—It must be in kernel space, and must be able to transcend both virtual and physical spaces in the Cloud, providing full transparency of all operations in the Cloud.
- *Provenance*—It must provide a full or a summarized/ concise provenance of data life cycles and transfers in the Cloud. This is also in tandem with the increase in the emphasis of data governance and accountability.
- *Single Auditable View*—It must be able to provide a single consolidated report for inspection.
- *Efficient storage*—It must be efficient in both short-term storage and long-term archival.
- *Analytics* —It must provide auditing features to enable strong analytics and quick observations of footprints of file activities and transfers.

7.4 Flogger: A Breakthrough in Data Accountability's System and Data Layer

7.4.1 Data-Centric Logging from Kernel Space of all Virtual and Physical Machines

One of the most promising breakthroughs in the kernel-space centralised monitoring group is Flogger [26–28, 56, 57] by HP Labs Singapore. Flogger, short for File-Centric Logger, is a novel file-centric logger that can be implemented in both VM and PM kernels in a non-invasive manner within nodes in the cloud. As such, it is able to integate seamlessly with security incident and event management (SIEM) tools such as ArcSight. Flogger is currently deployed in the USA Treasury, USA IRS, and HP customers located in Grenoble and Shanghai.

7.4.2 Flogger Architecture and Design

Part of the TrustCloud project, Flogger addresses the needs of system layer of cloud data accountability. Figure 5 shows Floggers and their accompanying components, and demonstrates the underlying mechanisms capturing file actions and movements from the underlying kernel space (depicted by the numeric sequence in Fig. 5) (Fig. 6). A simple example of the resulting file-centric log (in short, "flog") captured by both a VM and its host PM is shown in Fig. 7. The typical implementation consists of the following components (See Fig. 5):

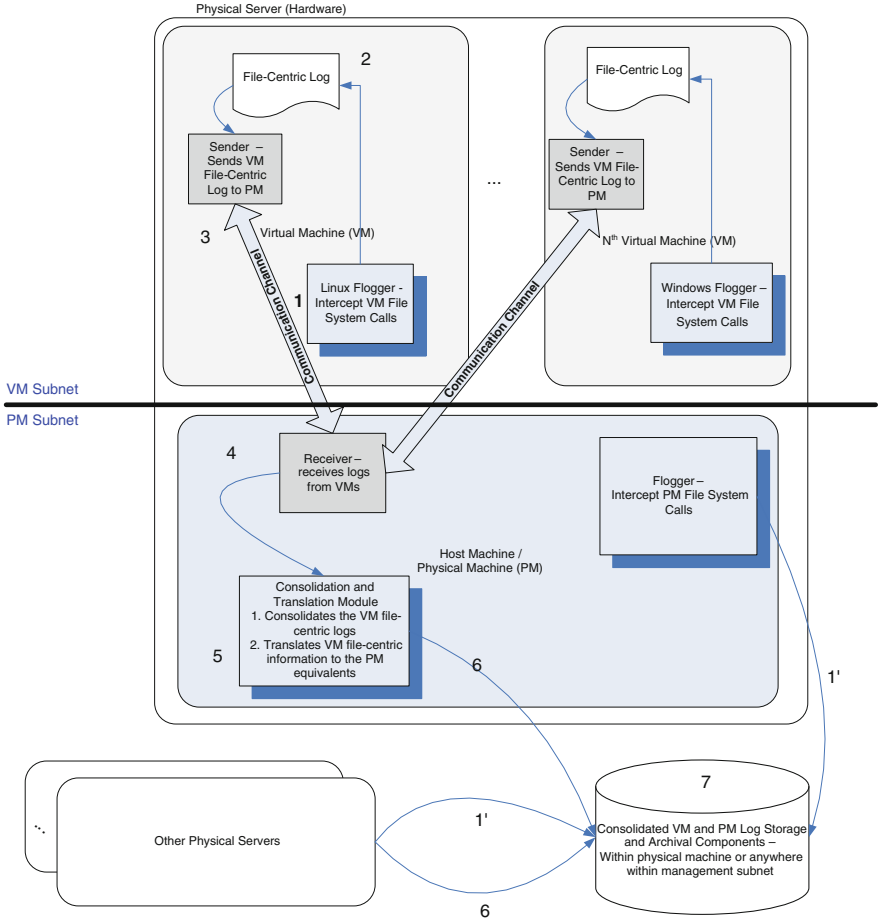


Fig. 5 How flogger collects data-centric provenance for data accountability from kernel space [26]

- Flogger (Linux)—A Linux Loadable Kernel Module (LKM) running on VM that intercepts file and network operations and writes the events as VM flogs.
- Flogger (Windows)—A Windows Device Driver running on PM that intercepts file operations and writes the events as PM flogs.
- Components accompanying Flogger.
  - File Sender Client program running on VM which transfers the VM log files from VM to PM via a direct communication channel.
  - File Sender daemon running on VM, which regularly executes the File Sender Client program.
  - File Sender Server program running on host PMs which receives the VM log files sent by the File Sender Client program.

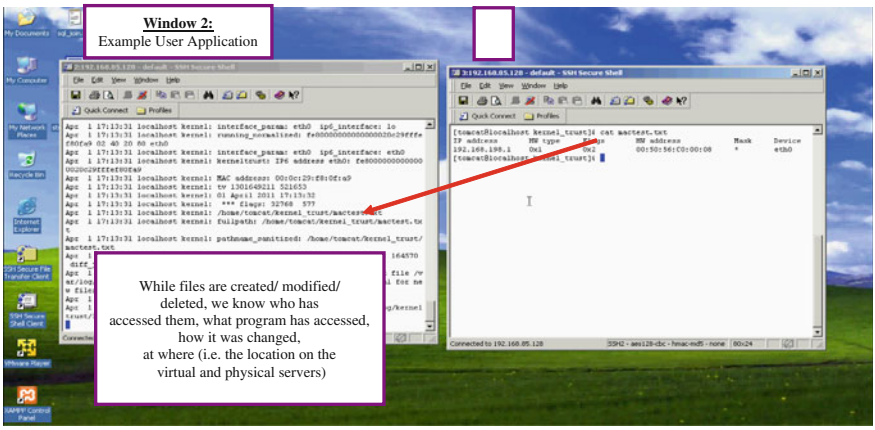


Fig. 6 Screen capture from a cloud virtual machine, showing how flogger logs file activities while processes are run

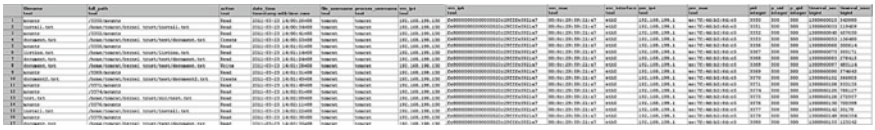


Fig. 7 Sample consolidated file-centric log (flog) extracted from querying the log storage [26]

– Two Database Loader daemons running on PM. The first one regularly loads the VM log files into a remote database server. The second one regularly loads the PM log files into the same remote database server.

With these components, we can then view and analyze the consolidated VM and PM flogs using any database front-end tools or in spreadsheet tools reading CSV/ TSV files.

7.4.3 How Flogger Collects Cloud Data Accountability Information from Kernel Space

Flogger captures file-centric logs (*a.k.a.* flogs) via the following steps (with reference to the labels in Fig. 5):

Step 1′: Linux Flogger/ Windows Flogger intercept every file access in the VMs. The Floggers capture the following information (Flog Subset A) (non-exhaustive list):

- *VM Accessed file name and full path e.g. /home/users/john/docs/sensitive.txt*
- *VM File access date/time*
- *VM IP address*
- *VM MAC address*

- *Machine type i.e. VM or PM*
- *UID of file owner of the accessed file*
- *GID of file owner of the accessed file*
- *UID of process owner who accessed the file*
- *GID of process owner who accessed the file*
- *Action done to accessed file e.g. Create, Read, Write, Network Transfer, Delete*
- *IP address of Destination machine (only applicable to Network Transfer action)*
- *Packet Content to be sent to the Destination machine (only for Network Transfer action).*

It is important to note that the list in Flog Subset A is not exhaustive and more attributes can be added to make the system more robust, e.g. more time stamps.

Step 1: Just like VMs, PMs also have Floggers which intercept the PMs file system calls and then stores them in the Data Store.
Step 2: After the file life-cycle related information are captured, they are sent to the host PM. The VM Flogger directly sends the captured information (Flog Subset A) to PM Receiver Daemon via a Communication Channel between VM and PM. The Communication Channel is special mechanism available on typical hypervisors which enable a serial cable-like communication between VMs and PMs. It does not involve networking transfers. Hence, no VM Flogger transfer Flogs to PM File Sender Servers via network transfers. This increases the security of the transfer of Flogs.
Step 3: VM File Sender Daemon regularly executes the File Sender Client which reads the File Access Details (Flog Subset A) and sends them to the PM via the Communication Channel between VM and PM.
Step 4: PM File Sender Server receives the File Access Details (Flog Subset A) from VM File Sender Client via the Communication Channel between VM and PM. The File Sender Server also extracts VM Data Store Address from the File Access Details (Flog Subset A) and sends it to PM Kernel Module.
Step 5: PM Flogger translates VM Data Store Address to PM Data Store Address and generates other PM information (Flog Subset B), for example (but not limited to): PM IP address, PM MAC address, etc.
Step 6: The PM Flogger sends Subset B to PM File Sender Server. Subsets A & B will give users a consolidated set of information (i.e. Flog) which can pinpoint the VMs and PMs involved in each file's life cycle to enable full accountability of distributed VM and PM architectures, e.g. cloud computing.
Step 7: Within the PM Subnet, the PM Database loader daemons write the joint/consolidated information (both Subset A & Subset B) to a Data Store e.g. database for future data mining and reporting. Note that all the consolidation of the Flogs across PMs into the Data Store take place only in the PM Subnet. Users in the VM Subnet should have no awareness of these behind-the-scenes steps. It is also noteworthy to know that we have not decided on the exact short, medium and long term storage of flogs, as this require another set of I/O experiments against benchmarks and scale.

According to [26], the early prototypes of Flogger were developed and run the implementation of Flogger on the following operating systems: Flogger (Linux) in

the Linux Family (CentOS 5.3, Fedora 15, Ubuntu 11.04) and Flogger (Windows) in the Windows Family (Windows XP Professional SP3, Windows Server 2008 R2). Flogs generated were also pushed into databases via the DB loaders. Experiments were conducted against the prominent open-source row-based relational database PostgreSQL 9.0 and in preparation for data analytical needs over flogs, we also experimented with the column-store MonetDB.

7.4.4 Tracking Data Provenance by Visualising Flogger Logs

The development of Floggers and the successful consolidation of simultaneously-generated VM and PM file-centric logs addressed the need for higher Cloud accountability and transparency. Visualisations were also created for the exploratory discovery and presentation of notable trends and patterns in the flogs. Visualization needs for end-users, administrators and regulators are different. For example, cloud service providers may only offer end-users knowledge about the high-level geography without revealing specific data centers' locations. End-users can still know if their data has violated cross-geography policies of data transfers. On the other hand, regulators may be granted special access accounts to visualize and audit the compliance of full data flows within the cloud.

Figure 8 shows a compelling example of a malicious insider detected by Flogger logs: a cloud system administrator stole and renamed a file, then send it across another instance in the cloud, and finally sending it out of the cloud via email.

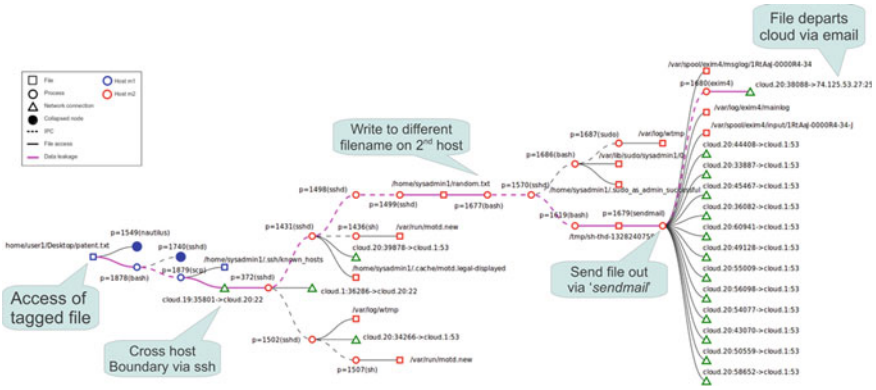


Fig. 8 Visualisation of flogger logs showing data provenance of a file which was stolen by a malicious insider, renamed and then sent out of the cloud by email

8 Other Approaches

While Flogger is the only technique (to our best knowledge) that achieved data accountability needs and is used in the real world, there are several other related efforts from industry, consortia and academia attempting to solve the cloud data accountability problem.

8.1 Industry, Standards and Consortia Works

Governance, Risk Management and Compliance (GRC) Stack of the Cloud Security Alliance

Cloud Security Alliance (CSA) is a non-profit organization formed to promote the use of best practices for providing security assurance within Cloud Computing, and provide education on Cloud Computing uses [12]. Two projects from the CSA's Governance, Risk Management and Compliance (GRC) Stack [13] are very relevant to our paper [11]:

- ***CloudAudit*** [14]—An ongoing API project hosted on Google Code, and aims to provide the technical foundation to enable transparency and trust in private and public cloud systems. It is a very simple, very lightweight and easy-to-implement API without cloud providers having to make a lot of programmatic changes.
- ***Trusted Cloud Initiative*** [10]—An initiative which aims to promote education, research and certification of secure and interoperable identity in the cloud. Most significant and related to our paper will be their movement towards the certification of 'trusted clouds'.
- ***Cloud Trust Protocol:***In mid-2010, at 6th Annual Information Technology Security Automation Conference (ITSAC) hosted by the National Institute of Standards and Technology (NIST), a representative from the technology provider CSC presented the "CloudTrust Protocol (CTP) with Security Content Automation Protocol (SCAP)" [25]. The CTP with SCAP was claimed to offer a simple way to request and receive the fundamental information needed to address cloud transparency. At the time of writing, the CTP is adopted as a CSA Project but there is no public release of the proposed tool.

HyTrust Appliance [23]

Recently in industry, HyTrust, a startup focusing on cloud auditing and accountability, has released a hypervisor consolidated log report and policy enforcement tool *(i.e. HyTrust Appliance)* for VM accountability management in clouds. HyTrust Appliance addresses the *System layer* of accountability in the cloud. Despite this, it focuses only on virtual layers and is not virtual-to-physical complexities.

HP Labs: Cloud and Security Lab

Aside from the TrustCloud project mentioned in Sect. 7.4, Pearson [31, 34, 36, 37] and Mowbray [30, 31] aimed to promote privacy protection via procedural and technical solutions encouraging the increase of accountability in the cloud [34, 37]. Their work on cloud privacy has addressed the high levels of the accountability layers, and the Flogger work by Ko et al. complement their work with the inclusion of the lower system layers.

8.2 Academic Research

University of Pennsylvania

Haeberlen et al. [19] assumed a primitive *AUDIT* with considerations of *agreement*, *service* and *timestamps*. However, *AUDIT* did not have a clear explanation of the scope, scale, phases and abstraction layers of accountability. It is our aim to complement their work. Their team has also proposed an approach for accountable virtual machines [19], and discussed a case study on the application to detect cheats in an online multi-player game Counterstrike. In our opinion, the scenario of a non-cloud based game was not a practical business scenario for cloud accountability.

Traditional Data and Workflow Provenance Research [49]

From the field of databases, data and workflow provenance research focuses on recording histories of derivation of final outputs of data at different levels of abstraction within databases. Provenance research may offer clues to recording logs in the workflow and data layers of cloud accountability.

9 What All Cloud Practitioners must Consider

Data Processor Versus Data Owner

Every practitioner must be mindful that the nature of cloud computing created a situation where users place the data they own into systems which they do not own (i.e. CSP's cloud services). This has caused the trust relationship tension as mentioned earlier in Sect. 1, and merits an in-depth investigation into techniques that can enable the clear and proper allocation of control from the provider and control from the owner.

Data Traceability

It would not be long before we experience cloud-wide malware or botnet attacks which will evade the current monitoring systems. The problem of data traceability must be overcome, and while doing so, the practitioner has to overcome the problem of scale of the logs or information collected. Another potential problem is the conflicting interests of accountability through data log and audit trail collection, and the privacy of the data logs themselves. There may be a need to obfuscate or anonymise certain private data components. For example, the cloud provider may wish to convert certain user information to cryptographic hash keys.

Cross-Boundary/ Trans-National Data Transfers

National boundaries are harder to enforce in a boundary-less cloud. For example, Google optimises its file allocation or placements, and does not consider geographical placements or restrictions for the typical user. This contrasts that of Amazon Web Services, which allocates cloud computing users and servers by zones (e.g. Asia Pacific).

The Laws have to Catch Up

While there are concerns about data accountability, the legislations have not caught up with the realities of the industry. Many companies and users have to resort to mediation in international settings, and lawmakers are finding it harder to define what constitutes data in their country and what does not. There is clearly a need for an innovative internationally accepted cloud computing legislation. Governmental bodies must control and impose penalties on breach of trust in the cloud, much like antitrust and privacy protection laws.

10 Holy Grails in Data Accountability in Cloud Computing

This section introduces the challenging but critical problems that have to be addressed in order to achieve full accountability of data in the cloud.

Data-Centric Source and Identity Tracking Based on Data Audits

While there is a need for the data to be traced, the next step is to quickly identify the identities of the people/ organisation/ malware involved. The main actor involved for each data anomaly or malicious event detected should be quickly identified, despite the scale of the cloud.

Tracking of Data Outside the Cloud

While techniques such as Flogger are able to detect events within the boundaries of the cloud, it is unable to track data which are leaving the cloud. Recent breakthroughs in tracing data outside of clouds have been reported by Tan et al. [50].

International Standards For Data Accountability

While there are standards such as ISO27001/27002 or COBIT, there is no internationally recognised standard which is voted and accepted by all nations. The standards should enable the situations which involve the ownership, liability of the data and services in cloud in cross-national situations must also be covered.

11 Conclusion

In this chapter, we introduced the concept and importance of data accountability in cloud computing. We also discussed about the scope, layers and the life cycle of data accountability. While several techniques are used in today's cloud computing environments, they are not aligned directly to the data centric needs of the cloud users. The cloud computing service provider must only be processors of data, and not indirect owners of data like the scenario today.

Due to the criticality of this trust relationship tension, data accountability in the cloud is a burgeoning research topic. More importantly, it is an important area which is currently lacking in real-life cloud computing environments. While the Flogger/ TrustCloud project has achieved this, many research areas, such as the data-centric identity tracking and tracking of data outside of cloud are still unsolved.

Acknowledgments The author would like to acknowledge the former members of the HP Labs TrustCloud team: Peter Jagadpramana, Chun Hui Suen, Markus Kirchberg, Yu Shyang Tan, Olive Qing Zhang, Aneeth Ahmed, Teck Hooi Lim, Miranda Mowbray. Siani Pearson, Duc Ha, Anurag Singla and Bu Sung Lee.

References

1. Armbrust M et al (2010) A view of cloud computing. Commun ACM 53(4):50–58
2. Baldwin A et al (2008) Auditing in shared distributed virtualized environments. HP Technical Reports
3. Brodkin J (2008) Gartner: seven cloud-computing security risks. Infoworld, 1–3
4. Buneman P et al (2000) Data provenance: some basic issues. FST TCS 2000: foundations of software technology and theoretical computer science. Springer, Berlin, pp 87–93
5. Buneman P et al (2001) Why and where: a characterization of data provenance. Database theory–ICDT 2001. Springer, Berlin, pp 316–330
6. Calder A (2006) Information security based on ISO 27001/ISO 17799: a management guide. The stationery office/Tso
7. Calder A, Watkins S (2008) IT governance: a manager's guide to datasecurity and ISO 27001/ISO 27002. Kogan Page Ltd, London
8. Chappell D (2009) Introducing windows azure.Microsoft, Dec. from http://www.microsoft.com/windowsazure/Whitepapers/IntroducingWindowsAzure/default.aspx
9. Chow R et al(2009) Controlling data in the cloud: outsourcing computation without outsourcing control.In: Proceedings of ACM workshop on cloud computing security (CCSW 2009). IL, ACM, Chicago
10. Cloud Security Alliance (2010) Cloud security alliance governance, risk management and compliance (GRC) stack. From http://www.cloudsecurityalliance.org/grcstack.html
11. Cloud Security Alliance (2010) Cloud security alliance homepage. From http://www.cloudsecurityalliance.org/
12. Cloud Security Alliance (2010) CloudAudit (A6–the automated audit, assertion, assessment, and assurance API). From http://cloudaudit.org/
13. Cloud Security Alliance (2010) Top threats to to cloud computing, Report (Ver.1.0)
14. Cloud Security Alliance (2010) Trusted cloud initiative. From http://www.cloudsecurityalliance.org/trustedcloud.html
15. CloudKick (2011) CloudKick—cloud monitoring and management. From https:// http://www.cloudkick.com/
16. Erl T (2005) Service-oriented architecture: concepts, technology, and design. Prentice Hall PTR, New Jeresy
17. Fujitsu Research Institute (2010) Personal data in the cloud: a global survey of consumer attitudes. From http://www.fujitsu.com/downloads/SOL/fai/reports/fujitsu_personal-data-in-the-cloud.pdf
18. Garfinkel S (2007) An evaluation of Amazon's grid computing services: EC2, S3, and SQS. Technical Report TR-08-07. Center for Research on Computation and Society, Harvard University, Cambridge
19. Haeberlen A (2010) A case for the accountable cloud. ACM SIGOPS Oper Syst Rev 44(2):52–57
20. Halpin H (2009) Provenance: the missing component of the semantic web for privacy and trust. In: Proceedings of the trust and privacy on the social and semantic web (SPOT) workshop at ESWC 2009, Citeseer
21. Hansen S, Atkins E (1993) Automated system monitoring and notification with swatch. In: USENIX association's Proceedings of the 7th systems administration (LISA VII) conference.
22. Hyperic (2010) CloudStatus. From http://www.cloudstatus.com/
23. HyTrust (2010) HyTrust appliance. From http://www.hytrust.com/product/overview/
24. Kim GH, Spafford EH (1994) The design and implementation of tripwire: a file system integrity checker. In: Proceedings of 2nd ACM conference on computer and communications security (CCS '94), ACM
25. Knode R (2010) CloudTrust 2.0. From http://scap.nist.gov/events/2010/itsac/presentations/day2/Security_Automation_for_Cloud_Computing-CloudTrust_2.0.pdf

26. Ko RKL et al (2011) Flogger: a file-centric logger for monitoring file access and transfers within cloud computing environments. In: Proceedings of trust, security and privacy in computing and communications (TrustCom), 2011 IEEE 10th international conference on, IEEE, pp 765–771
27. Ko RKL et al (2011)TrustCloud—a framework for accountability and trust in cloud computing. In: Proceedings of IEEE 2nd cloud forum for practitioners (IEEE ICFP), IEEE computer society, Washington DC. USA
28. Ko RKL et al (2011) From system-centric to data-centric logging-accountability, trust & security in cloud computing. In: Proceedings of defense science research conference and expo (DSR)
29. Love R (2005) Kernel Korner: intro to iNotify. Linux J (139):8
30. Mowbray M, Pearson S (2009) A client-based privacy manager for cloud computing. In: Proceedings of ACM
31. Mowbray M et al (2010) Enhancing privacy in cloud computing via policy-based obfuscation. J Supercomputing 1–25
32. Ning WZP et al (2010) Always up-to-date-scalable offline patching of VM images in a compute cloud. In: Proceedings of IBM technical papers (RC24956)
33. Oxford University Press (2005) Concise oxford english dictionary. Retrieved 5 Dec 2005.
34. Pearson S (2009) Taking account of privacy when designing cloud computing services. In: Proceedings of 2009 ICSE workshop on software engineering challenges of cloud computing, IEEE computer society
35. Pearson S, Balacheff B (2003) Trusted computing platforms: TCPA technology in context. Prentice Hall PTR, New Jeresy
36. Pearson S, Benameur A (2010) Privacy, security and trust issues arising from cloud computing. In: Proceedings of 2nd international conference on cloud computing (2010) IEEE, Indiana
37. Pearson S, Charlesworth A (2009) Accountability as a way forward for privacy protection in the cloud. Cloud Computing. Springer, Berlin, pp 131–144
38. Proudler G (2005) Concepts of trusted computing. In: Mitchell CJ (ed) Trusted computing, IEE professional applications of computing series. The Institute of Electrical Engineers (IEE), London, pp 11–27
39. Rezaee Z et al (2002) Continuous auditing: building automated auditing capability. Auditing 21(1):147–164
40. Roesch M (1999) Snort-lightweight intrusion detection for networks. In: Proceedings of 13th large installation system administration conference (LISA), Seattle, Washington
41. Rosenblum M, Ousterhout J (1992) The design and implementation of a log-structured file system. ACM Trans Comput Syst (TOCS) 10(1):26–52
42. Sarbanes-Oxley Act (2002) Public law no. 107–204. In: Proceedings of 107th US congress. Government Printing Office, Washington DC
43. Shende J (2010) Live forensics and the cloud - part 1. Cloud Comput J. 2011, From http://cloudcomputing.sys-con.com/node/1547944. Accessed on 27 Sep 2010
44. Silberschatz A et al (1991) Operating system concepts. Addison-Wesley, New York
45. Silicon Graphics International Corp (2009) File alteration monitor (FAM) overview. From http://oss.sgi.com/projects/fam/
46. Slagell A et al (2004) Network log anonymization: application of crypto-pan to cisco netflows. In: Proceedings of NSF/AFRL workshop on secure knowledge management (SKM '04), Buffalo
47. Slagell A, Yurcik W (2006) Sharing computer network logs for security and privacy: a motivation for new methodologies of anonymization. In: Proceedings of workshop of the 1st international conference on security and privacy for emerging areas in communication networks, IEEE 2005
48. Stevens W (1994) TCP/IP illustrated vol. I: the protocols. Pearson Education India, India
49. Tan W (2007) Provenance in databases: past, current, and future. IEEE Data Eng 30:3–12
50. Tan YS et al (2012) Tracking of data leaving the cloud. Trust, security and privacy in computing and communications (TrustCom). In: Proceedings of 2012 IEEE 11th international conference on IEEE

51. US Congress (1996) Health insurance portability and accountability Act (HIPAA) of 1996. Public Law 104–191
52. VMWare Hyperic (2011) Performance monitoring for cloud services. From http://www.hyperic.com/products/cloud-status-monitoring
53. Vouk M (2008) Cloud computing–issues, research and implementations. In: Proceedings of 30th international conference on information technology interfaces, 2008 (ITI 2008) IEEE
54. Wei J et al (2009) Managing security of virtual machine images in a cloud environment. In: Proceedings of ACM
55. Woolf H (1974) The Merriam-webster dictionary. Pocket Books, New York
56. Zhang OQ et al (2011) How to track your data: the case for cloud computing provenance. In: Proceedings of Cloud computing technology and science (CloudCom), 2011 IEEE 3rd international conference on IEEE
57. Zhang OQ et al (2012) How to track your data: rule-based data provenance tracing algorithms. In: Proceedings of trust, security and privacy in computing and communications (TrustCom), 2012 IEEE 11th international conference on IEEE
58. Zimmermann H (2002) OSI reference model-The ISO model of architecture for open systems interconnection. Commun IEEE Trans on 28(4):425–432

Privacy Preservation over Big Data in Cloud Systems

Xuyun Zhang, Chang Liu, Surya Nepal, Chi Yang and Jinjun Chen

1 Introduction

Cloud computing and Big Data, two disruptive trends at present, pose significant influence on current IT industry and research communities [6, 9]. Cloud computing provides massive computation power and storage capacity which enable users to deploy applications without infrastructure investment. Coupled with cloud computing, data sets have become so large and complex that it is a considerable challenge for traditional data processing tools to handle the analysis pipeline of these data. Generally, such data sets are often from various sources and of different types (Variety) such as unstructured social media content and half-structured medical records and business transactions, and are of large size (Volume) with fast data in/out (Velocity).The MapReduce framework has been widely adopted by a large number of companies and organizations to process huge-volume data sets [11]. Unlike the traditional one, MapReduce incorporated with cloud computing becomes more flexible, scalable and cost-effective. A typical example is the Amazon Elastic MapReduce (Amazon EMR) service [1]. Users can invoke Amazon EMR to conduct their MapReduce computations based on the powerful infrastructure offered by Amazon Web Services and are

X. Zhang (✉) · C. Liu · C. Yang · J. Chen
Faculty of Engineering and IT, University of Technology Sydney, Sydney, Australia
e-mail: xyzhanggz@gmail.com

C. Liu
e-mail: changliu.it@gmail.com

C. Yang
e-mail: chiyangit@gmail.com

J. Chen
e-mail: jinjun.chen@gmail.com

S. Nepal
ICT Centre, CSIRO, Sydney, Australia
e-mail: Surya.Nepal@csiro.au

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_8,

charged in proportion to the usage of the services. In this way, it is economical and convenient for companies and organizations to capture, store, organize, share and analyze Big Data to gain competitive advantages.

However, privacy concerns in MapReduce platforms are aggravated because the privacy-sensitive information scattered among various data sets can be recovered with more ease when data and computational power are considerably abundant. Although some privacy issues are not new, their importance is amplified by cloud computing and Big Data [9]. With the widely adoption of online cloud services and proliferation of mobile devices, the privacy concern about processing on and sharing of sensitive personal information is increasing. For instance, HealthVault [29], an online health service provided by Microsoft company, is deployed on the Windows Azure cloud platform [28]. Although these data in such cloud services are usually deemed extremely sensitive, they usually offer significant human benefits if analyzed and mined by organizations like disease research centers. As the MapReduce framework is usually adopted to handle the data in scenarios such as hybrid cloud or multiple clouds, solutions to the privacy issues in the framework are urgently desired.

Recently, the research on privacy issues in the MapReduce framework on cloud has commenced. Mechanisms such as encryption [8], access control [35], differential privacy [36] and auditing [42] are exploited to protect the data privacy in the MapReduce framework. These mechanisms are well-know pillars of privacy protection and still have open questions in the context of cloud computing and Big Data [9]. Usually, the data sets uploaded into cloud are not only for simply storage, but also for online cloud applications, i.e., the data sets are dynamical. If we encrypt these data sets, processing on data sets efficiently will be quite a challenging task, because most existing applications only run on unencrypted data sets. Although recent progress has been made in homomorphic encryption which theoretically allows performing computation on encrypted data sets, applying current algorithms are rather expensive due to their inefficiency [17]. Furthermore, data holders and data users in cloud are different parties in most applications, e.g., cloud health service providers and pharmaceutical companies. In such cases, encryption or access control mechanisms alone fail to ensure privacy preservation and data utility exposure. Data anonymization is a promising category of approaches to achieve such a goal [15]. However, the computing infrastructure and paradigm has been moving to the MapReduce framework in order to get scalability, e.g., the newly emerging project Apache Mahout [2]. Thus, how to achieve privacy preservation and high utility of Big Data in the MapReduce framework on cloud for mining or analytic applications is still a challenge problem and needs extensive investigation.

In this chapter, we propose a flexible, scalable, dynamical and cost-effective privacy-preserving framework based on MapReduce on cloud. The framework is built on the top of MapReduce, and functions as a filter to preserve the privacy of data sets before these data sets are accessed and processed by MapReduce. Specifically, the framework provides interfaces to data holders to specify various privacy requirements based on different privacy models. Once privacy requirements are specified, the framework launches anonymization algorithms of MapReduce version to efficiently anonymize data sets for subsequent MapReduce tasks. Anonymous data sets

are retained and reused to avoid re-computation cost. Thus, the privacy-preserving framework handles the dynamical update of data sets as well to maintain the privacy requirements of such data sets. Besides anonymization, the framework also integrates encryption techniques to cost-effectively ensure the privacy of multiples data sets that are independently anonymized in terms of different privacy requirements. Finally, a corresponding prototype system is developed based on our cloud environment to implement the framework with the four features discussed above. We conduct extensive experiments on real-world data sets to evaluation the proposed framework. Empirical evaluation demonstrates that the privacy-preserving framework can anonymize large-scale data sets and mange the anonymous data sets in a highly flexible, scalable, efficient and cost-effective fashion.

The remainder of this chapter is organized as follows. The next section reviews the related work about privacy protection in the MapReduce framework, cloud computing and Big Data. In Sect. 3 we briefly describe the MapReduce framework as background knowledge. Section 4 formulates the details of the proposed privacy-preserving framework based on MapReduce. The prototype design and our cloud environment are presented in Sect. 5. We empirically evaluate the components of the prototype system in Sect. 6. Finally, we conclude this chapter and discuss our future work in Sect. 7.

2 Related Work

We briefly review recent research on privacy protection in the MapReduce framework on cloud.

The Kerberos authentication mechanism [32] is integrated into the MapReduce framework of Hadoop [18] after the 1.0.0 version. Kerberos is a distributed authentication service that allows a process (a client) running on behalf of a principal (a user) to prove its identity to a verifier (an application server, or just server) without sending data across the network that might allow an attacker or the verifier to subsequently impersonate the principal. However, access control fails to preserve privacy because data users can infer the privacy-sensitive information if they access to the unencrypted data. Roy et al. [36] investigated the data privacy problem caused by the MapReduce framework and presented a system named *Airavat* which incorporates mandatory access control with differential privacy [13]. The mandatory access control will be triggered when the privacy leakage exceeds the threshold specified by data providers, so that both privacy preservation and high data utility are ensured. The privacy disclosure degree is specified via the differential privacy model which aims to provide means to maximize the accuracy of queries from statistical databases while minimizing the chances of identifying its records. However, the results produced in this system are mixed with certain noise, which is unsuitable to many applications that need data sets without noise, e.g., medical experiment data mining and analysis.

Encryption is widely adopted as a straightforward approach to ensure data privacy on cloud against malicious users. Yet the data are definitely not encrypted if they are processed by applications in cloud computing. Homomorphic encryptions may be the prospective and promising mitigations to these security problems. A researcher at IBM announced having developed a fully homomorphic encryption scheme that allow data to be processed in encrypted form [17]. Even though homomorphic scheme has eliminated the theoretical obstacle to fully homomorphic encryption, the immense computational cost still deters it from being put into practice. Special operations such as query or search on encrypted data sets stored on cloud has been extensive studied [8, 19, 24], although performing general operations on encrypted data sets is still quite challenging. Instead of encrypting all data sets, Puttaswamy et al. [35] described a set of tools called *Silverline* that can separate all functionally encryptable data from other cloud application data, where the former is encrypted for privacy preservation while the later is left unencrypted for application functions. However, the sensitivity of data is required be labeled in advance. Zhang et al. [47] proposed a privacy leakage upper-bound constraint based approach to preserve privacy of multiple data sets by only encrypting part of data sets on cloud. The encryption and anonymization techniques are combined together in this approach to arrive at cost-effective privacy preservation. Data sets required encrypting are identified first, and the rest are anonymized to achieve privacy requirements, with incurring minimum expense when using these data sets.

Particular to the MapReduce framework, several systems have been proposed to handle privacy concerns of computation and storage in the framework. Blass et al. [5] proposed a privacy-preserving scheme named *PRISM* for the MapReduce framework on cloud to perform parallel word search on over encrypted data sets. PRISM transforms the problem of word search into a set of parallel instances of private information retrieval on small datasets. Each instance on a small dataset is efficiently solved by a node in the cloud during the "Map" phase of MapReduce. Outcomes of map computations are then aggregated during the "Reduce" phase. Nevertheless, many cloud applications require the MapReduce framework to conduct tasks like data mining and analytics over these data sets besides search. Ko et al. [20] proposed the *HybrEx* MapReduce model to provide a way that sensitive and private data are processed within a private cloud while others can be safely extended to public cloud. Similarly, Zhang et al. [46] proposed a system named *Sedic* which partitions MapReduce computing jobs in terms of the security labels of data they work on and then assigns the computation without sensitive data to a public cloud. However, sensitivity of data is also required be acquired in advance in above two systems. [40] proposed a service integrity assurance framework named *SecureMR* for the MapReduce framework. The framework consists of five components and mainly aims at MapReduce service integrity and preventing denial of service (DoS) attack. Meanwhile, it also tries to preserve the simplicity, applicability and scalability of MapReduce. But we mainly focus herein on privacy-preserving issues in our proposed framework. Our privacy-preserving framework attempts to produce anonymized data sets according to data holders' privacy requirements for subsequent MapReduce tasks.

3 Cloud Systems and MapReduce Preliminary

Cloud computing is one of the most hyped IT innovations at present, having sparked plenty of interest in both IT Industry and academia. Recently, IT giants such as Amazon, Google, IBM and Microsoft have invested huge sums of money in building up their public cloud products, and indeed they have developed their own products, e.g., Amazon's Web Services, Google's App Engine and Compute, and Microsoft's Azure. Meanwhile, several corresponding open source cloud computing solutions are also developed, like Hadoop, Eucalyptus, OpenNebula and OpenStack. The cloud computing definition published by the U.S. National Institute of Standards and Technology (NIST) comprehensively covers the commonly agreed aspects of cloud computing [27]. In terms of the definition, the cloud model consists of five essential characteristics, three service delivery models and four deployment models. The five key features encompass on-demand self-service, broad network access, resource pooling (multi-tenancy), rapid elasticity and measured services. The three service delivery models are Cloud Software as a Service (SaaS), e.g., Google Docs, Cloud Platform as a Service (PaaS), e.g., Google App Engine, and Cloud Infrastructure as a Service (IaaS), e.g., Amazon EC2 and S3. The four deployment models include private cloud, community cloud, public cloud and hybrid cloud.

Technically, cloud computing could be regarded as an ingenious combination of a series of developed or developing ideas and technologies, establishing a novel business model by offering IT services using economies of scale. In general, the basic ideas encompass service computing, grid computing, distributed computing, etc. The core technologies that cloud computing principally built on include web service technologies and standards, virtualization, novel distributed programming models like MapReduece, and cryptography. All the participants in the cloud computing can benefit from this new business model. The giant IT enterprises can not only run their own core businesses, but also make profit by delivering the spare infrastructure services to others. Small and medium-size businesses are capable of focusing on their own core businesses via outsourcing the boring and complicated IT management to other cloud service providers, usually at a fairly low cost. Especially, cloud computing facilitates start-ups considerably, enabling them to build up their business with low upfront IT investments as well as cheap ongoing costs. Moreover, due to the flexibility of cloud computing, companies can adapt their business readily and swiftly by enlarging or shrinking the business scale dynamically, without concerns about losing anything.

MapReduce is a scalable and fault-tolerant data processing framework that enables to process huge volume of data in parallel with many low-end commodity computers [11]. MapReduce was first introduced in the year 2004 by Google with similar concepts in functional languages dated as early as 1960s. It has been widely adopted and received extensive attention from both academia and industry due to its promising capability. In the context of cloud computing, the MapReduce framework becomes more scalable and cost-effective because infrastructure resources can be provisioned on demand. Simplicity, scalability and fault-tolerance are three main salient features

of MapReduce framework. Therefore, it is convenient and beneficial for companies and organizations utilize MapReduce services such as Amazon EMR to process Big Data and obtain core competiveness.

Basically, a MapReduce task consists of two primitive functions, Map and Reduce, defined over a data structure named as key-value pair (key, value). Specifically, Map function can be formalized as *Map*: $(k_1, v_1) \rightarrow (k_2, v_2)$, i.e., Map function takes a pair (k_1, v_1) as input and then output another intermediate key-value pair (k_2, v_2). These intermediate pairs will be consumed by Reduce function as input. Formally, Reduce function can be represented as *Reduce*: $(k_2, \text{list}(v_2)) \rightarrow (k_3, v_3)$, i.e., Reduce function takes intermediate k_2 and all its corresponding values $\text{list}(v_2)$ as input and output another pair (k_3, v_3). Usually, (k_3, v_3) list is the results which MapReduce users attempt to get. Both Map and Reduce functions are specified by data users in terms of their specific applications.

To make such a simple programming model work effectively and efficiently, MapReduce implementations provide a variety of fundamental mechanisms such as data replication and data sorting. Besides, distributed file systems like Hadoop distributed file system [37] are substantially crucial to make the MapReduce framework run in a highly scalable and fault-tolerant fashion. Recently, the standard MapReduce has been extensively revised into many variations in order to handle data in different scenarios. For instance, Incoop [4] is proposed for incremental MapReduce computation, which detects changes to the input and automatically updates the output by employing an efficient, fine-grained result reuse mechanism. Several novel techniques: a storage system, a contraction phase for Reduce tasks, and an affinity-based scheduling algorithm are utilized to achieve efficiency without sacrificing transparency. As standard MapReduce framework lack build-in supports for iterative programming which arise naturally in many applications including data mining, web ranking, graph analysis, model fitting, and so on, Haloop [7] and Twister [14] are designed to support iterative MapReduce computation. HaLoop is built on top of Hadoop and extends it with a new programming model and several important optimizations that include: a loop-aware task scheduler, loop-invariant data caching, and caching for efficient fixpoint verification. Twister a distributed in-memory MapReduce runtime optimized for iterative MapReduce computations.

4 Privacy-Preserving Framework Design

4.1 System Overview

To preserve the privacy of data sets processed by data users using MapReduce, we propose a Privacy-Preserving Framework PPF between original data sets and user-specified MapReduce tasks. Figure 1 depicts the privacy-preserving framework based on MapReduce. As shown in Fig. 1, original data sets are stored confidentially on cloud by data owners, and can never be accessed directly by data users. Then

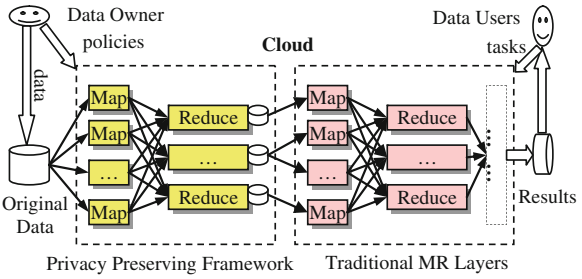


Fig. 1 MapReduce framework with privacy-preservation

data holders specify privacy requirements and submit them to the privacy-preserving framework. The framework is responsible for anonymizing original data sets according to privacy requirements. Certain anonymous data sets are retained to avoid recomputation. Thus, the framework is also responsible for managing the stored data sets and updating the data when new data join. Data users can then specify their application logic in MapReduce jobs and run these jobs on the anonymized data sets. The privacy-preserving framework itself exploits MapReduce jobs to conduct the computation required in data anonymization. This is plausible and necessary to use MapReduce to accomplish the computation for in anonymizing and managing these data sets because data sets are usually of huge volume and complexity in the context of cloud computing and Big Data.

In order to comply with the features specific to cloud computing and Big Data, we identify several system requirements that should be satisfied when designing the privacy-preserving framework as follows.

- ***Flexible***. The framework should provide a user interface through which the data owners can specify various privacy requirements. Usually, data sets will be accessed and processed by different data users for different application.
- ***Scalable***. Scalability is necessary for current privacy-preserving approaches, because the scale of data sets is too large to be processed by existing centralized algorithms. So the privacy-preserving framework should be also scalable to handle data anonymization and managements. Concretely, the data-intensive or computation-intensive operations in the framework should be executed efficiently in parallel.
- ***Dynamical***. In cloud computing, most applications accumulate data sets over time, e.g., cloud health services will receive a large number of information from users. The scale of such data sets becomes larger and larger, forming Big Data. Hence, the privacy-preserving framework should handle dynamical data sets, the privacy and utility of such data sets can still be ensured in time when updates occur.
- ***Cost-effective***. In the pay-as-you-go feature of cloud computing, saving IT cost is one of the core enablers. Thus, it is also desired for the privacy-preserving framework to save the expense of privacy preservation as much as possible while the privacy preservation and data utility can still be ensured.

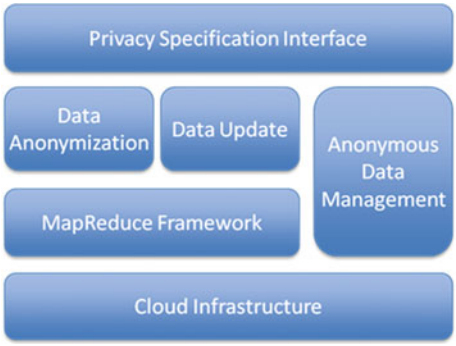


Fig. 2 System structure of the privacy-preserving framework

We design the privacy-preserving framework in terms of the four system requirements formulated above. Accordingly, the framework consists of four main modules, namely, Privacy Specification Interface (PSI), Data Anonymization (DA), Data Update (DU) and Anonymous Data sets Management (ADM). Based on the four modules, the PPF can achieve the four system requirements. The system structure of the framework is depicted in Fig. 2 where the relationships among the above four modules, MapReduce Framework and cloud infrastructure are described.

As shown in Fig. 2, DA, DU and ADM are the three main functional modules. They conduct concrete operations on data sets in terms of the privacy models specified in the PSI module. The DA and DU modules take advantage of the MapReduce framework to anonymize data sets or adapt anonymized data sets when updates occur. The ADM module is responsible for managing anonymized data sets in order to save expense via avoiding re-computation. Thus, ADM utilize cloud infrastructure directly to accomplish the tasks. Unlike traditional MapReduce platforms, the MapReduce platform in our research is deployed on top of the cloud infrastructure to gain high scalability and elasticity. We will discussion the four proposed modules in detail in following sections.

4.2 Privacy Specification Interface

As to privacy models and protection techniques, privacy-preserving data publishing research community has extensively investigated on the issues and made fruitful progress with a variety of privacy models, privacy preserving methods and algorithms [15]. Usually, original data sets can be accessed and process by different data users for different purposes, leading to various privacy risks in these scenarios. Moreover, the privacy requirements of data owners possibly vary over time. As such, a systematic and flexible privacy specification model is proposed to frame privacy requirements. We have the following definition on privacy specification.

Definition 1 (*Privacy Specification*). The privacy requirements specified by a data owner are defined as a Privacy Specification (PS). A privacy specification is formally represented by a vector of parameters, i.e., $PS = <PMN, Thr, AT, Alg, Gra, Uti>$. The parameters in the vector are elaborated subsequently.

PMN represents the name of a privacy model. Out of recent proposed privacy models, we employ three widely adopted privacy models in the privacy-preserving framework, namely, *k*-anonymity [38], *l*-diversity [26] and *t*-closeness [25].The three privacy principles provide different levels of privacy-preserving extent. The privacy principle *k*-anonymity means that the number of the anonymized records that correspond to a quasi-identifier is required to be larger than a threshold. Otherwise, once certain quasi-identifiers are too specific that only a small group of people is linked to them, these individuals are linked to sensitive information with high confidence, resulting privacy breach. Here, quasi-identifiers represent the groups of anonymized data records. Based on *k*-anonymity, *l*-diversity requires that the sensitive values correspond to a quasi-identifier is not less than a threshold and therefore stricter than *k*-anonymity. The strictest is *t*-closeness, requiring the distribution of sensitive values correspond to a quasi-identifier to be close that of original data sets.

Parameter *Thr* is the threshold of the specified privacy model, i.e., *k*, *l* and *t* in the above three privacy principles.

Parameter *AT* denotes application type. Data owners can specify the goals of anonymized data sets, e.g., classification, clustering or general use. If the use of anonymized data sets is known in advance, anonymization algorithms can produce anonymized data sets with higher data utility when the privacy are still preserved. Without knowing the types of applications that consume the anonymized data sets, the DA module produces anonymized data sets for general use. Different information metrics will be utilized for different application types [15].

The anonymization algorithms are indicated by the parameter *Alg*. A variety of algorithms have been developed for different privacy principles and application types. Details will be described in Sect. 4.3.

Parameter *Gra* represents the granularity of the privacy specification. It determines the scope of the privacy preservation. Usually, only part of original data sets is shared with data users, and different data users are possibly interested in different part of Big Data. Moreover, only part of attributes of a data records is considered in the process of anonymization. Thus, the granularity parameter is quite useful.

The data utility parameter *Uti* is an optional one. Data owners can specify how much data utility they allow to expose to data users. In general, privacy and data utility are two roughly opposite aspects of privacy preservation. When privacy thresholds are given, most anonymization algorithms usually expose as much as possible. On the contrary, we can make the data sets anonymous as much as possible if data utility is fixed.

Above all, the privacy-preserving framework systematically and comprehensively provides diverse privacy specifications to achieve the flexibility and diversity of privacy preservation.

4.3 Data Anonymization

Several categories of anonymization techniques have been proposed, including generalization [30], suppression [3] and anatomization [41]. Generalization means that a parent domain value is replaced with its child domain values within its domain taxonomy tree to preserve privacy. Suppression just simply replaces original values in a data record with one pre-given symbol, thereby hiding all the information of the value. Anatomization separates the attribute values and sensitive values without modifying values of attributes, and places them in different locations to enforce privacy preservation. In the PPF, we utilize generalization for the anonymization because it is widely investigated and adopted in existing algorithms. Specifically, four generalization schemes have been proposed, namely, full-domain generalization (FG) [22], sub-tree generalization (SG) [16], multi-dimensional generalization (MG) [23] and cell generalization (CG) [43]. Full-domain generalization makes all domain values in an attribute generalized to the same level of the taxonomy tree. Sub-tree generalization requires that either all child domain values or none of a non-leaf node in a taxonomy tree is generalized. Multidimensional generalization takes multiple attributes together into account when generalizing domain values. The aforementioned schemes are global recoding, i.e., if a domain value is generalized, all its instances are generalized. Cell generalization, on the contrary, is local recoding which generalizes identical instances into different levels of domain values. Roughly speaking, data utility exposed by these four schemes increase in the order: FG $<$ SG $<$ MG $<$ CG, when the same privacy requirement is given. But note that the anonymized data sets produced by MG and CG suffers from data exploration problem, i.e., the anonymized data sets contains inconsistent data. For instance, one original attribute value can be generalized into two different higher-level values. The *Alg* parameter in a privacy specification can indicate which anonymized scheme can be used to anonymize data sets.

However, most existing anonymization algorithms are centralized, meaning that these algorithms fail to handle Big Data. Usually, Big Data are so large that they cannot be fit in the memory in one normal cloud computation node. Hence, they are usually stored across a number of nodes. Therefore, it is a challenging problem to anonymize large-scale data sets cloud for existing anonymization algorithm. Hence, we revise existing algorithms into MapReduce versions, in order to exploit the MapReduce to efficiently anonymize data sets in a scalable and parallel fashion.

The data anonymizing (DA) module consists of a series of anonymized algorithms of MapReduce version. Basically, each anonymized algorithm has a MapReduce driver program and several pairs of Map and Reduce programs. Usually, these Map and Reduce programs, constituting a MapReduce job, will be executed iteratively because the anonymization is an iterative process. The standard MapReduce implementation mainly supports one-pass data processing rather than iterative processing. As such, we take advantage of a recently proposed MapReduce implementation named Haloop [7] to deploy our anonymization algorithms. In this way, the DA module can anonymize data sets efficiently.

4.4 Data Update

The anonymized data sets are retained on cloud for different data users. So, these data sets are persistent every time the anonymized data sets are generated. However, the data sets in applications on cloud are dynamic and increase dramatically over time, resulting in Big Data. Hence, we have to update original data sets and anonymized data sets. A straightforward way is to anonymize the already updated data sets from scratch. From the efficiency perspective, it is usually unacceptable to anonymize all data sets once an update occurs. Furthermore, the privacy preservation cannot be ensured according to the analysis in Ref. [34]. Therefore, a better way is to just anonymize the update part and adjust the already anonymized data sets. For anonymous data sets, anonymization level is used to describe the degree of privacy preservation.

Usually, the anonymization level for the already anonymized data sets satisfies the given privacy requirements. So, new data can be simply anonymized to current anonymization level when updates occur. But the newly anonymized data sets possibly violate the privacy requirements because they are likely to be too specific. In such a case, we have to adjust the anonymization level of the whole anonymized data sets to ensure the privacy preservation for all data. Another aspect of privacy preservation is to produce data utility as much as possible to data users when privacy requirements are satisfied. For data anonymization, an interesting phenomenon is that for a given privacy requirement, the more data are anonymized, the lower anonymization level will be. A lower anonymization level means more data utility can be produced, because the anonymized values are more specific. Consequently, just anonymizing new data to the current anonymization level is not sufficient even though this satisfies the privacy requirements definitely. To expose more data utility, we need to lower the anonymization level of the whole anonymized data sets. As such, three basic operations are provided in the DU module, namely, *update*, *generalization* and *specialization*. *Generalization* is utilized to raise the anonymization level while *specialization* is to lower the anonymization level [48].

4.5 Anonymous Data Management

As described in the last section, anonymous data sets are retained for data sharing, mining and analytics. Another consideration is to save IT expense. In the context of cloud computing, both computation and storage resources will be charged in proportion to their usage in terms of the pay-as-you-go feature. In this sense, it is beneficial to store certain part of intermediate data sets rather than re-compute them repeatedly. Yuan et al. [44, 45] has extensively investigated the trade-offs between data storage and re-computation and proposed a series of strategies. There research demonstrates that retaining a bundle of data sets can significantly reduce overall cost in cloud systems. Based on their research, anonymous data sets are stored to save

cost and managed systematically in the PPF. We exploit data provenance [10] to manage the data generation relationships among these data sets. Data provenance is commonly defined as the origin, source or history of derivation of some objects and data, which can be reckoned as the information upon how data was generated [31]. Re-producibility of data provenance can help to regenerate a dataset from its nearest existing predecessor datasets rather than from scratch.

Retaining a large number of independently anonymous data sets potentially suffers from the privacy breach problems. Since the PSI module provides flexible privacy specifications, a data set can be possibly anonymized into different anonymous data sets. As a result, privacy-sensitive information can be recovered from different anonymous data sets. To address this inference problem in multiple data sets, we have proposed an approach that incorporates encryption to ensure privacy preservation [47]. Basically, we can encrypt all the anonymous data sets and share them to specific users. However, encrypting all data sets will incur expensive overhead because these anonymous data sets are usually accessed or processed frequently by many data users. So, we propose to encrypt part of these data sets to save privacy-preserving cost while privacy preservation can still be ensured.

In the privacy-preserving framework, privacy of all anonymous data sets are quantified according to Zhang et al. [47]. Then we carefully select part of anonymous data sets to encrypt via our proposed approach. In this way, the PPF can achieve cost-effective privacy preservation.

5 Prototype System and Environment

5.1 Prototype System

We have developed a prototype system for the privacy-preserving based on Hadoop, an open-source implementation of MapReduce, and the OpenStatck cloud platform.

For the DA module, we have already developed a variety of anonymization algorithms of MapReduce version. Take the sub-tree generalization scheme as an example. We have developed a driver program (MRTDS Driver) and three pairs of iterative MapReduce jobs (IGPL Initialization Map/Reduce, IGPL Update Map/Reduce, Data Specialization Map/Reduce, respectively). Such MapReduce programs can anonymize large-scale in an efficient and scalable fashion. In the DU module, we have developed three basic operations based on the MapReduce framework, namely, MRUpdate, MRGeneralization and MRSpecialization. We also developed privacy quantification algorithms and heuristic algorithms to determine the anonymous data sets that need to be encrypted with the minimum privacy-preserving cost in the ADM module. Together with the SPI module, these three modules constitute the privacy-preserving framework and achieve the four system requirements as described in Sect. 4.1.

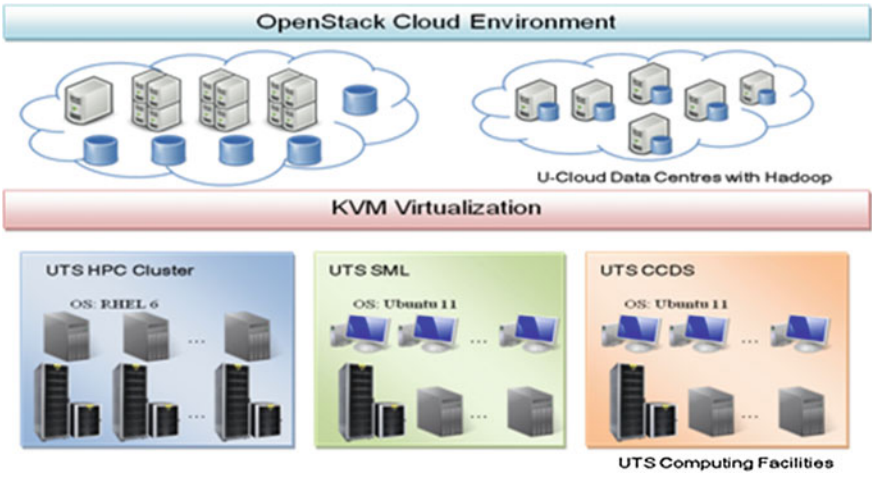


Fig. 3 System overview of U-cloud

5.2 Deployment Environment

We develop and deploy the privacy-preserving framework based on our cloud environment U-cloud. U-Cloud is a cloud computing environment at University of Technology Sydney (UTS). The system overview of the U-Cloud system is depicted in Fig. 3. The computing facilities of this system are located among several labs in the Faculty of Engineering and IT, UTS. On top of hardware and Linux operating system (Ubuntu), we install KVM virtualization software [21] which virtualizes the infrastructure and provides unified computing and storage resources. To create virtualized data centers, we install OpenStack open source cloud environment [33] which is responsible for virtual machine management, resource scheduling, task distribution and interaction with users. Furthermore, Hadoop [18] is installed based on the private cloud built via OpenStack to facilitate MapReduce computing paradigm and massive data processing. We also deploy the recently developed MapReduce implementation Haloop [7] on the cloud to support iterative job execution.

6 Experimental Evaluation

In this section, we empirically evaluate the main components of the privacy-preserving framework via conduction a series of experiments on real-world data sets. Specifically, we compare the performance of the DA, DU, and ADM components with their corresponding existing approaches. The experimental environment is our cloud platform U-cloud as already described in Sect. 5.2. We use Adult data set [39], a public data set commonly used as a de facto benchmark for testing data

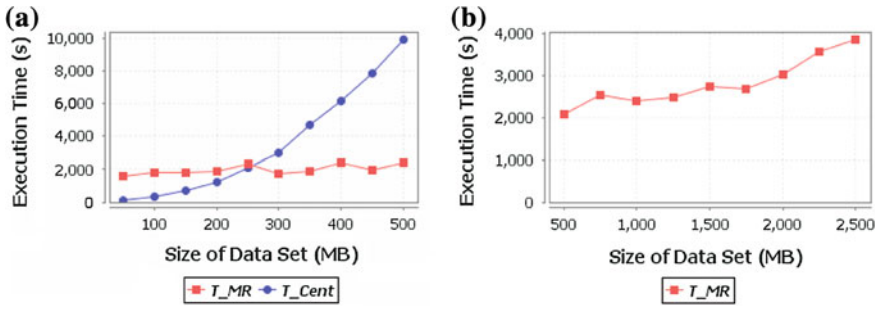


Fig. 4 Change of execution time w.r.t. data size: TPTDS versus CentTDS

anonymization for privacy preservation. We also generated enlarged data sets based on the Adult data set. We describe the experiment results of the three components in the following subsections.

6.1 Experiment results of Data Anonymization

We compare our approach of data anonymization with a state-of-the-art centralized approaches proposed in Refs. [16, 30]. We run both approaches on data sets varying from 50 MB to 2.5 GB. We check whether both approaches can scale over large-scale data sets and measure execution time to evaluate the efficiency. The change of execution time with respect to data set size is depicted in Fig. 4. The execution time of our approach is denoted as T_{MR}, while that of the centralized approach is denoted as T_{Cent}.

From Fig. 4a, it is seen that TCent surges when the data size increases while TMR increases slightly even though it has a higher start value. The centralized approach suffers from memory insufficiency when the data size is larger than 500 MB, while Fig. 4b shows that our approach can still scale over much larger data sets. Hence, the privacy-preserving framework can significantly improve the scalability and efficiency compared with existing state-of-the-art anonymization approaches.

6.2 Experiment Results of Data Update

We compare our approach of anonymous data update with a state-of-the-art approach proposed in Ref. [12]. We run both approach with the number of records in data update batch ranging from 2,000 to 20,000. We measure the update time to evaluate the efficiency. The execution time of our approach is denoted as t_I while that of the existing approach is denoted as t_E. Figure 5 illustrates how the difference between t_1

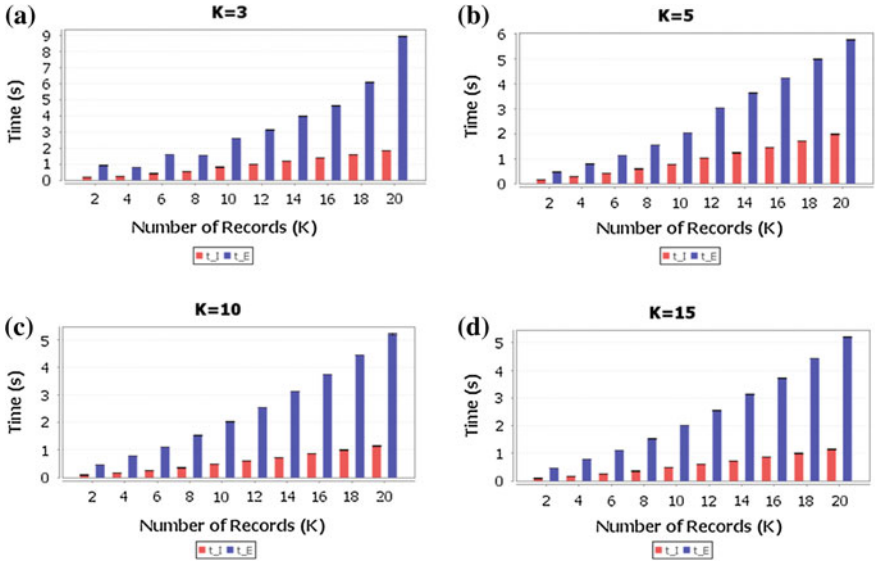


Fig. 5 Change of update time w.r.t. the number of update records

and t_E changes with respect to the number of data records in an update batch when K is fixed, where K is the user-specified k-anonymity parameter.

When K is fixed, it can be seem from Fig. 3 that the difference between t_1 and t_E becomes bigger and bigger when the number of data records increases. This trend demonstrates that the privacy-preserving framework can significantly improve the efficiency of privacy preservation on large-volume incremental data sets over existing approaches.

6.3 Experiment Results of Anonymous Data Management

We compare our approach of retaining anonymous data sets with the existing approach which encrypt all data sets. We run both approaches on data sets with the number of data sets varying from 100 to 1,000. The privacy-preserving monetary cost is measured to evaluate the cost-effectiveness. The costs of our approach and the existing one are denoted as C_{HEU} and C_{ALL}, respectively. The change of the cost with respect to the number of data sets is illustrated in Fig. 6 when ε_d is fixed. The parameter ε_d is a user-specified privacy requirement threshold meaning that degree of privacy disclosure must be under ε_d.

We can see from Fig. 6 that that the difference between C_{ALL} and C_{HEU} becomes bigger and bigger when the number of intermediate data sets increases. That is, more expense can be reduced when the number of data sets becomes larger. Thus, this trend

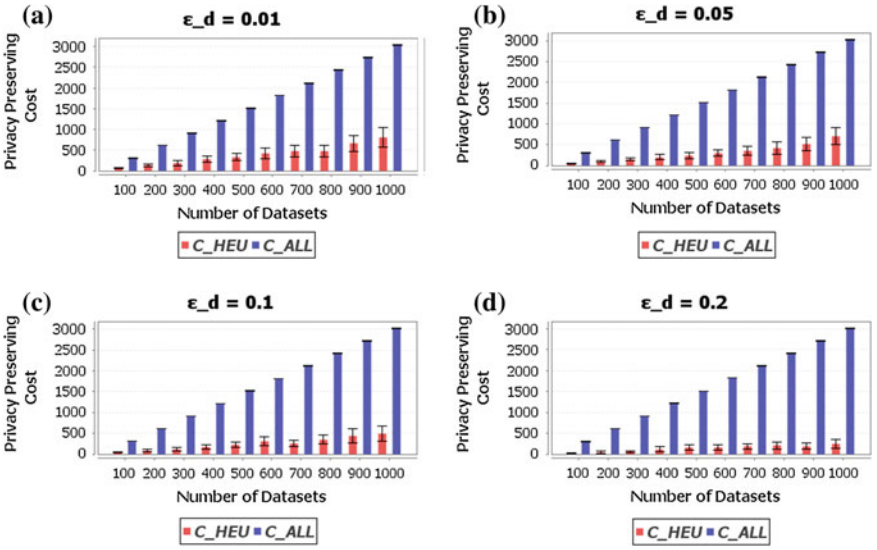


Fig. 6 Change of privacy-preserving cost w.r.t the number of data sets

means the privacy-preserving framework can reduce the privacy-preserving cost of retaining anonymous data sets significantly over the existing approach in real-world Big Data scenarios.

As a conclusion, the evaluation results in the above experiments demonstrate that the proposed privacy-preserving framework can anonymize large-scale data sets and mange the anonymous data sets in a highly scalable, efficient and cost-effective fashion.

7 Conclusions and Future Work

In this chapter, we have proposed a flexible, scalable, dynamical and cost-effective privacy-preserving framework based on MapReduce on cloud. We have formulated four basic system requirements for privacy preservation in the context of cloud computing and Big Data. To achieve the four system requirements, we have designed four modules for the privacy-preserving framework, namely, Privacy Specification Interface (PSI), Data Anonymization (DA), Data Update (DU) and Anonymous Data Management (ADM). We also have developed a prototype system for the framework and deployed it in our cloud environment. Empirical evaluation has demonstrated that the privacy-preserving framework can anonymize large-scale data sets and mange the anonymous data sets in a highly flexible, scalable, efficient and cost-effective fashion.

Privacy concerns of Big Data on cloud have attracted the attention of researchers in different research communities. But ensuring privacy preservation of large-scale data sets still needs extensive investigation. We plan to improve the proposed privacy-preserving framework comprehensively in the future. Besides, we plan to integrate this privacy-preserving framework with other data processing frameworks that employ the MapReduce framework as the computation engine, e.g., the Apache Mahout project that is a data mining library built atop of MapReduce.

References

1. Amazon Web Services (2013) Amazon elastic MapReduce (Amazon EMR). http://aws.amazon.com/elasticmapreduce/. Accessed on 10 Mar 2013
2. Apache (2013) Apache Mahout machine learning library. http://mahout.apache.org/. Accessed on 10 Mar 2013
3. Bayardo RJ, Agrawal R (2005) Data privacy through optimal k-anonymization. In: Proceedings of the 21st IEEE international conference on data, engineering (ICDE'05), pp 217–228
4. Bhatotia P, Wieder A, Rodrigues R, Acar UA, Pasquin R (2011) Incoop: MapReduce for incremental computations. In: Proceedings of the 2nd ACM symposium on cloud, computing (SoCC'11), pp 1–14
5. Blass E-O, Pietro RD, Molva R, Önen M (2012) PRISM–privacy-preserving search in MapReduce. In: Proceedings of the 12th international conference on privacy enhancing technologies (PETS'12), pp 180–200
6. Borkar V, Carey MJ, Li C (2012) Inside "Big Data Management": ogres, onions, or parfaits? In: Proceedings of the 15th international conference on extending database technology (EDBT'12), pp 3–14
7. Bu Y, Howe B, Balazinska M, Ernst MD (2012) The HaLoop approach to large-scale iterative data analysis. VLDB J 21(2):169–190
8. Cao N, Wang C, Li M, Ren K, Lou W (2011) Privacy-preserving multi-keyword ranked search over encrypted cloud data. In: Proceedings of the 31st annual IEEE international conference on, computer communications (INFOCOM'11), pp 829–837
9. Chaudhuri S (2012) What next?: A half-dozen data management research goals for big data and the cloud. In: Proceedings of the 31st symposium on principles of database systems (PODS'12), pp 1–4
10. Davidson SB, Khanna S, Milo T, Panigrahi D, Roy S (2011) Provenance views for module privacy. In: Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems (PODS'11), pp 175–186
11. Dean J, Ghemawat S (2010) MapReduce: a flexible data processing tool. Commun ACM 53(1):72–77
12. Doka K, Tsoumakos d, Koziris N (2011) KANIS: preserving k-anonymity over distributed data. In the 5th international workshop on personalized access, profile management, and context awareness in databases (PersDB'11), Seattle, WA, USA
13. Dwork C (2006) Differential privacy. In: Proceedings of the 33rd international colloquium on automata, languages and programming (ICALP'06), pp 1–12
14. Ekanayake J, Li H, Zhang B, Gunarathne T, Bae S-H, Qiu J, Fox G (2010) Twister: a runtime for iterative MapReduce. In: Proceedings of the 19th ACM international symposium on high performance, distributed computing (HDPC'10), pp 810–818
15. Fung BCM, Wang K, Chen R, Yu PS (2010) Privacy-preserving data publishing: a survey of recent developments. ACM Comput Surv 42(4):1–53
16. Fung BCM, Wang K, Yu PS (2007) Anonymizing classification data for privacy preservation. IEEE Trans Knowl Data Eng 19(5):711–725

17. Gentry C (2009) Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on theory of, computing (STOC'09), pp 169–178
18. Hadoop (2013) http://hadoop.apache.org. Accessed on 10 Mar 2013
19. Hu H, Xu J, Ren C, Choi B (2011) Processing private queries over untrusted data cloud through privacy homomorphism. In: Proceedings of the IEEE 27th international conference on data, engineering (ICDE'11), pp 601–612
20. Ko SY, Jeon K, Morales R (2011) The hybrex model for confidentiality and privacy in cloud computing. In: Proceedings of the 3rd USENIX conference on hot topics in cloud computing (HotCloud'11), article 8
21. KVM (2013) http://www.linux-kvm.org/page/Main_Page. Accessed on 10 Mar 2013
22. LeFevre K, DeWitt DJ, Ramakrishnan R (2005) Incognito: efficient full-domain k-anonymity. In: Proceedings of 2005 ACM SIGMOD international conference on management of data (SIGMOD '05), pp 49–60
23. LeFevre K, DeWitt DJ, Ramakrishnan R (2006) Mondrian multidimensional k-anonymity. In: Proceedings of 22nd international conference on data engineering (ICDE '06), article 25
24. Li M, Yu S, Cao N, Lou W (2011) Authorized private keyword search over encrypted data in cloud computing. In: Proceedings of the 31st international conference on distributed, computing systems (ICDCS'11), pp 383–392
25. Li N, Li T, Venkatasubramanian S (2010) Closeness: a new privacy measure for data publishing. IEEE Trans Knowl Data Eng 22(7):943–956
26. Machanavajjhala A, Kifer D, Gehrke D, Venkitasubramaniam M (2007) L-diversity: privacy beyond k-anonymity. ACM Trans Knowl Discov Data 1(1): article 3
27. Mell P, Grance T (2009) The NIST definition of cloud computing (Version 15). National Institute of Standards and Technology, Information Technology Laboratory, U.S.
28. MicroSoft (2013) Windows Azure. http://www.windowsazure.com/en-us/. Accessed on 10 Mar 2013
29. Microsoft HealthVault (2013) http://www.microsoft.com/health/ww/products/Pages/healthvault.aspx. Accessed on 10 Mar 2013
30. Mohammed N, Fung B, Hung PCK, Lee CK (2010) Centralized and distributed anonymization for high-dimensional healthcare data. ACM Trans Knowl Discov Data 4(4): article 18
31. Muniswamy-Reddy K-K, Macko P, Seltzer M (2010) Provenance for the cloud. In: Proceedings of the 8th USENIX conference on file and storage technologies (FAST'10), pp 197–210
32. Neuman BC, Ts'o T (1994) Kerberos: an authentication service for computer networks. IEEE Commun Mag 32(9):33–38
33. OpenStack (2013) http://openstack.org/. Accessed on 10 Mar 2013
34. Pei J, Xu J, Wang Z, Wang W, Wang K (2007) Maintaining k-anonymity against incremental updates. In: Proceedings of the 19th international conference on scientific and statistical database management (SSBDM '07), pp article 5
35. Puttaswamy KPN, Kruegel C, Zhao BY (2011) Silverline: toward data confidentiality in storage-intensive cloud applications. In: Proceedings of the 2nd ACM symposium on cloud computing (SoCC'11), article 10
36. Roy I, Setty STV, Kilzer A, Shmatikov V, Witchel E (2010) Airavat: security and privacy for MapReduce. In: Proceedings of 7th USENIX conference on networked systems design and implementation (NSDI'10), pp 297–312
37. Shvachko K, Hairong K, Radia S, Chansler R (2010) The Hadoop distributed file system. In: Proceedings of 2010 IEEE 26th Symposium on mass storage systems and technologies (MSST'10), pp 1–10
38. Sweeney L (2002) K-anonymity: a model for protecting privacy. Int J Uncertainty Fuzziness Knowl Based Syst 10(5):557–570
39. UCI machine learning repository. ftp://ftp.ics.uci.edu/pub/machine-learning-databases/. Accessed on 10 Mar 2013
40. Wei W, Juan D, Ting Y, Xiaohui G (2009) SecureMR: a service integrity assurance framework for MapReduce. In: Proceedings of annual computer security applications conference (ACSAC '09), pp 73–82

41. Xiao X, Tao Y (2006) Anatomy: simple and effective privacy preservation. In; Proceedings of 32nd international conference on very large data bases (VLDB'06), pp 139–150
42. Xiao Z, Xiao Y (2011) Accountable MapReduce in cloud computing. In; Proceedings of the 2011 IEEE conference on computer communications workshops (INFOCOM WKSHPS), pp 1082–1087
43. Xu J, Wang W, Pei J, Wang X, Shi B, Fu AWC (2006) Utility-based anonymization using local recoding. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data (KDD'06), pp 785–790
44. Yuan D, Yang Y, Liu X, Chen J (2010) A cost-effective strategy for intermediate data storage in scientific cloud corkflow systems. In: Proceedings of the 2010 IEEE international symposium on parallel and distributed processing (IPDPS'10), pp 1–12
45. Yuan D, Yang Y, Liu X, Chen J (2011) On-demand minimum cost benchmarking for intermediate dataset storage in scientific cloud workflow systems. J Parallel Distrib Comput 71(2):316–332
46. Zhang K, Zhou X, Chen Y, Wang X, Ruan Y (2011) Sedic: privacy-aware data intensive computing on hybrid clouds. In: Proceedings of 18th ACM conference on computer and communications, security (CCS'11), pp 515–526
47. Zhang X, Liu C, Nepal S, Pandey S, Chen J (2012) A privacy leakage upper-bound constraint based approach for cost-effective privacy preserving of intermediate datasets in cloud. IEEE Trans Parallel Distrib Syst 24(4): 1192–1202
48. Zhang X, Liu C, Nepal S, Chen J (2013) An efficient quasi-identifier index based approach for privacy preservation over incremental data sets on cloud. J Comput Syst Sci 79(5):542–555

Securing Outsourced Databases in the Cloud

Dongxi Liu

1 Introduction

Cloud database services, such as Amazon Relational Database Service (RDS) and Microsoft SQL Azure, are attractive for companies to outsource their databases. In cloud database services, a shared platform (e.g., database server hardware and software) is provided to host multiple outsourced databases. By using cloud database services, a client can deploy databases quickly without making the large upfront investment on proprietary hardware and software. Hence, the cloud database services can help companies reduce the total cost of ownership on their database management. Moreover, due to the scalability and elasticity of cloud database services, an enterprise can dynamically increase or decrease the cloud resources allocated to its databases according to its business requirements.

For databases deployed into a cloud database service, the service providers have the privilege to access the databases, since the underlying hardware and software are under their physical control. Hence, the databases in the cloud might be accessed improperly by the service providers accidentally or intentionally. The potential of such improper accesses causes the concern of users about the privacy of their outsourced databases. On the other hand, the underlying software of cloud database services (e.g., hypervisors, operating systems and DBMSs) might be compromised by attackers. At this case, the privacy of the outsourced databases is also at risk of being breached. Though attractive, cloud database services may not be fully exploited if the problem of data privacy cannot be satisfyingly addressed [2].

To protect data in cloud databases, a straightforward approach is to encrypt data before they are stored. By this way, the service providers or attackers can access only meaningless ciphertexts. However, after encryption, the databases may not be efficiently queried. It is not acceptable to decrypt the entire databases before executing

D. Liu (✉)
CSIRO Computational Informatics, Marsfield, NSW 2122, Australia
e-mail: dongxi.liu@csiro.au

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_9,

a query. If a database is large, decrypting the entire database will be unacceptably slow. In addition, if the decryption is done in the cloud, the encrypted database again becomes insecure. Ideally, a query should be executed directly over the encrypted database, producing the encrypted query result, which can only be decrypted by users.

There have been some encryption schemes and systems proposed to facilitate the encryption of databases and their queries [3, 7, 9, 16]. In the following, we describe several requirements for database encryption and query. These requirements are identified according to the role of databases in information systems. In an information system, a database may run for a very long period of time. During this period, the number of records in the database and the stored values may change dramatically. For example, a table containing staff personal information may contain only a few of staff records initially and then thousands of records after several years. And moreover, the staff salaries may increase from a few hundred dollars per week to a few thousand dollars per week.

R1: The native operations in DBMSs, such as SUM and AVG, should be used to support the operations on encrypted data, instead of using user-defined functions, since user-defined functions may not be optimized well by the DBMSs.

R2: The relational data model should be taken to manage encrypted data, and thus the existing DBMSs can be applied without worrying about their physical implementations (e.g., column-oriented DBMSs or row-oriented DBMSs).

R3: The providers of cloud database services should not need encryption keys. Otherwise, the database privacy is not protected against untrusted cloud database services.

R4: The maximum sum of values in one table column should not be predetermined, since it is hard to determine for a long-standing database.

R5: The number and range of values should not be required, since they may increase dramatically when a database runs for a long period of time.

As to be discussed in the next section, the existing works do not satisfy all the requirements. After discussing the exiting works, we will present an approach that satisfies all the above requirements.

2 Related Works

In this section, we introduce the security mechanisms deployed in the cloud database services, and the schemes of querying encrypted databases.

2.1 Security Mechanisms in Cloud Database Services

All cloud database services provide mechanisms to address the security concern of service users. The basic security scheme in cloud database services is access control [4]. Since a cloud database service is shared by multiple users, users must correctly

authenticate themselves to the service to access their own databases. However, the access control mechanisms cannot completely remove the security concern since users have to trust the cloud service providers to correctly implement the access control mechanisms and not to access the databases improperly. In addition, each cloud database service may provide its specific security scheme, as discussed below.

In the Oracle Database Cloud [14], all data are stored by using Oracle's Transparent Data Encryption scheme, which encrypts data stored on disk and in backups. The encryption and decryption of data is performed in the Oracle Database Cloud and transparent to users. The benefit is that users do not have to do extra work to encrypt their databases. However, the encryption keys are managed in the Oracle Cloud, so users must trust the service provider not to decrypt their data improperly. In the system presented later, the keys are managed by users themselves, so the cloud service providers cannot decrypt their data at any time.

The Amazon Relational Database Service (Amazon RDS) provides security mechanisms at the network level [1]. A user can control network access to his database by configuring firewall settings. Moreover, Amazon RDS allows database server instances to run in Amazon Virtual Private Cloud (Amazon VPC), which helps the isolation of database server instances. The databases in Amazon VPC can be accessed by the existing IT infrastructure of an enterprise through encrypted IPsec link.

Similarly, the Microsoft SQL Azure service [13] also controls network access to databases by configuring the SQL Database firewall. The firewall can be configured at the server level or at the database level. The server level firewall controls machines which can build connections with the virtual database servers. The database level firewall controls accesses to certain database instances in the virtual database servers. The communication with the Microsoft SQL Azure service is encrypted with SSL.

2.2 Related Schemes of Encrypting Database

The CryptDB [16] is a system supporting SQL queries over encrypted databases. This system needs the extension of existing DBMSs to support homomorphic operations like SUM and AVG, because the exploited homomorphic encryption scheme [15] performs multiplication on ciphertexts to get the sum of corresponding plaintexts. The existing DBMSs cannot natively support multiplication of values in one table column.

In Ref. [7], a mechanism of supporting aggregate queries is proposed, which is designed only for column-based databases by encrypting multiple values in one table column into one ciphertext. Hence, the mechanism in Ref. [7] is not flexible for data insertion and deletion, since the data to be updated is always packed together with other data not to be updated.

In Ref. [6, 12], a homomorphic encryption scheme is proposed to be efficient and practical. But it needs users to determine the maximum sum of plaintexts, which should not be bigger than the modulus. Otherwise, the scheme is not homomorphic.

That is, if it is used to encrypt values in a table column, the maximum sum of such values must be predetermined and it cannot be bigger than the modulus.

An order preserving encryption scheme has been proposed in Ref. [3]. In this scheme, the *i*th value in the plaintext domain is mapped to the *i*th value in the ciphertext domain, such that the order between plaintexts is preserved between ciphertexts. The scheme [3] can only deal with plaintexts in a finite domain. That is, the number of values in the plaintext domain must be known before using the scheme. The cryptographic analysis of the order preserving encryption scheme is performed in Ref. [5].

The work [2] shows a way of building order preserving polynomials, which are based on the polynomials proposed by Shamir for secret sharing [17]. In this mechanism, the number and range of plaintexts are needed to determine the range of random coefficients in a polynomial. On the other hand, the evaluation results of order preserving polynomials may reveal the distribution of plaintexts, since similar plaintexts are transformed with similar polynomials.

In Ref. [9], an indexing mechanism for range queries is proposed. This mechanism is not strictly order preserving since two different values may be mapped into the same bucket, which is used when checking query conditions. The mechanism can lead to inaccuracy of query results and hence some post-processing is needed to remove unexpected query results.

3 An Architecture of Managing Encrypted Databases

As discussed before, it is desirable that when protecting databases with encryption, the existing DBMSs should be applied without change. For this purpose, we describe an architecture of managing encrypted databases, as shown Fig. 1. In this architecture, enterprise applications are supposed to be built over databases, which are outsourced to a public cloud. Since the public cloud cannot be trusted, the outsourced databases are encrypted for data privacy. Therefore, the database service providers can only access meaningless ciphertexts. Between the encrypted databases and applications is a query proxy, which mediates their communication.

When an application issues a database query, the proxy translates it into a new one that is to be executed over the encrypted database in the cloud. When the query results are returned from the cloud, the query proxy decrypts them and then forwards the decrypted results to the application. For the application, the query result is the same as it directly accesses an unencrypted database. Each query is translated independently by the proxy. Hence, an enterprise can deploy multiple query proxies to process queries in parallel, though there is only one query proxy depicted in Fig. 1.

The query proxy maintains some meta data to perform query translation. The meta data might include databases schemas, encryption keys, and other specific information needed by cryptographic schemes and system management. Encryption keys are surely needed for encrypting or decrypting data. Database schemas are needed when determining the attribute names that are not given explicitly in a query

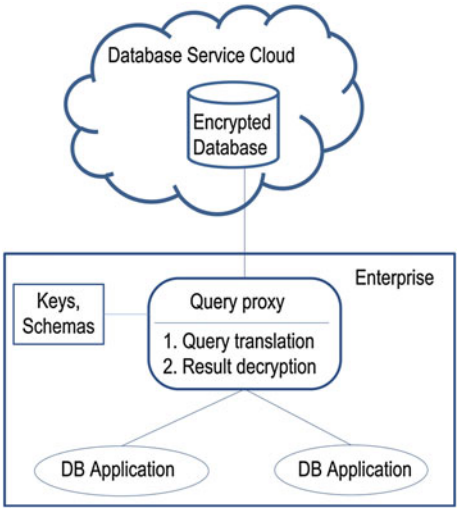


Fig. 1 Architecture of managing encrypted databases

statement, as in the query "*select * from a staff table*". The order-preserving indexing scheme to be introduced later needs the sensitivity of input values to be stored in the query proxy.

The architecture takes a threat model, in which the proxy is deployed into the administrative boundary of the enterprise. Hence, the untrusted database service providers cannot access keys and database schemas maintained in the query proxy. Moreover, the thread model does not allow untrusted service providers to perform plaintext-chosen attacks, since they do not control the query proxy. The prevention of plaintext-chosen attacks at the architecture level is required by order-preserving encryption or indexing schemes [3, 5, 11], since these schemes used for processing range queries leaks order information of plaintexts and hence are vulnerable to plaintext-chosen attacks.

4 Overview: An Approach with Good Usability

We will introduce an approach of managing encrypted databases. This approach comprises an order-preserving indexing scheme, a homomorphic encryption scheme, and how to apply them to encrypt databases and query encrypted databases. The schemes in this approach satisfy all five requirements discussed in the first section, so this approach has good usability in protecting databases in the cloud. In the following, we discuss several types of database queries. The order-preserving indexing scheme and the homomorphic encryption scheme are applied to deal with different query types.

A database query can be an equality query, a range query, an aggregate query or their combinations. An equality query uses the equality comparison as the filtering predicate. For example, the query "*select staffs whose room number is 405*" needs to compare whether the room number of a staff is equal to 405. A range query filters records with inequality comparisons, such as the query "*select staffs who join the company from year 2000 to 2012*". An aggregate query generates a value from a set of data. For example, the query "*select salary average of all staffs*" is an aggregate query. These queries can be combined to perform complex query operations. For example, we can query "*select salary average of staffs* who *join the company from year 2000 to 2012*".

To support the above query types, the approach encrypts a value (e.g., a field in a record) with different cryptographic schemes. For supporting equality queries, a secure hash scheme (e.g., HMAC-SHA1) is used to encrypt the value, such that the ciphertexts of equal values are still equal. To deal with range queries and the aggregate queries using MIN and MAX, the order-preserving indexing scheme is applied. Since this scheme preserves the order of plaintexts, the comparison of two ciphertexts can determine the order of their corresponding plaintexts. For aggregate queries of using SUM and AVG operations, the homomorphic encryption scheme is used to encrypt the value. As a result, the sum or average of ciphertexts can be decrypted to obtain the sum or average of corresponding plaintext values. Note that if a column does not support range queries (e.g., a Boolean column), then the order-preserving indexes of values in this column do not need to be produced; if a column cannot be summed, instead of using the homomorphic encryption scheme, we can use AES to encrypt values in this column.

In addition, database schemas may also contain sensitive information. In the approach, the schemas are anonymised by hashing table names and attribute names when creating an encrypted database. Since a field in a record can be encrypted into multiple ciphertexts, the encrypted records have more fields than the corresponding plaintext records.

5 Order-Preserving Indexing

The order-preserving indexing scheme is used to answer range queries and the aggregate queries using Min and Max over encrypted data. The order-preserving indexing scheme described in this section is proposed in [11].

5.1 Overview

The order-preserving indexing scheme preserves the order between two plaintext values. Formally, the order-preserving index scheme is defined below.

Definition (Order-Preserving Indexing) Suppose k is a secret key, and v_1 and v_2 are two values. If $v_1 < v_2$, then OPI$(k, v_1) < OPI(k, v_2)$, where OPI(k, v_i) means the order-preserving index of value v_i under the secret key k.

That is, by comparing OPI(k, v_1) and OPI(k, v_2), we can know the order of v_1 and v_2. Thus, when order-preserving indexes are stored in encrypted databases, they can be compared by DBMSs when executing queries over encrypted data.

Unlike order-preserving encryption schemes [3, 5], our order-preserving indexing scheme is one-way. That is, from indexes, the original values cannot be recovered even if the key is known. As a result, the order-preserving indexing scheme is simpler to design than order-preserving encryption schemes, since the decryption operation over indexes does not need to be considered.

Our order-preserving indexing scheme is vulnerable to plaintext-chosen attacks, similar to order-preserving encryption schemes. Suppose an adversary can access an oracle to choose arbitrary plaintexts to index. Then, given an index i, the adversary can approximate its plaintext value by the following steps (i.e., binary search).

Step 1: choose an arbitrary value to index
Step 2: compare the index with i
Step 2a: if greater, then choose a smaller value to index
Step 2b: otherwise, choose a bigger value to index;
Step 3: repeat Step 2 until the index is closest to i

To prevent plaintext-chosen attacks, as discussed before, the system architecture is designed to deploy the query proxy in a trusted domain, where the query proxy processes queries issued by database applications.

5.2 An Order-Preserving Indexing Scheme

Before introducing the order-preserving scheme [11], we first define the sensitivity of plaintext values, which indicates the smallest difference between two plaintext values.

Definition (Sensitivity of Plaintexts) Let V be the set of all plaintext values. The sensitivity of V is the minimum element in the set $\{|v_1 - v_2| | v_1 \in V, v_2 \in V \text{ and } v_1 \neq v_2\}$.

By its definition, the sensitivity is always bigger than 0. Though in the above definition, all plaintext values are needed to define their sensitivity. Actually, the sensitivity can be determined by data types or application requirements. For example, if a field contains integers, then the sensitivity is 1; if a field contains even numbers, then the sensitivity can be 2; for a field containing salaries of the form $d_1 d_2 d_3 \cdot d_4 d_5$, where d_i is a digit, the sensitivity can be 0.01.

The order-preserving scheme described in [11] is build over the expression $a^* f(x)^* x + b + noise$, where a and b are real numbers, f is a function that needs

to be instantiated, and *noise* is a random number. For a value v, its index is then computed by $a^* f(v)^* v + b + noise$. Since *noise* is randomly sampled, the indexing scheme is probabilistic. That is, indexing one value twice may generate two different indexes.

To keep the order-preserving property, the following requirements should be satisfied by parameters in the indexing expression $a^* f(x)^* x + b + noise$.

- $a > 0$;
- *noise* is sampled from the range $[0, a^* f(v + sens)^*(v + sens) - a^* f(v)^*(v))$, where $sens$ is the sensitivity of plaintext values;
- $f(x) > 0$ for $x \neq 0$;
- $f(x_1) \geq f(x_2)$ for $x_1 > x_2 \geq 0$ or $x_1 < x_2 \leq 0$.

Note that there is no requirement to plaintext values (i.e., their number, their range and their distribution). The notation $\text{nindex}^{sens}_{[a,b,f]}(v)$ is used to represent the index of value v, where a, b, f and $sens$ are regarded as secret keys of the indexing scheme. The following theorem ensures the order-preserving property of the indexing scheme.

Theorem (Order-Preserving Property) Given the sensitivity *sens* of input values V, for all $v_1 \in V$ and $v_2 \in V$, $if\, v_1 > v_2$, then $\text{nindex}^{sens}_{[a,b,f]}(v_1) > \text{nindex}^{sens}_{[a,b,f]}(v_2)$.

To use the indexing expression $\text{nindex}^{sens}_{[a,b,f]}$, we need to specify the instances of f, which must satisfy the parameter requirements to f. The following are several instances defined and analyzed in [11].

- $f(x) = |x|$;
- $f(x) = x^2$;
- $f(x) = \log_c(d + e^*|x|)$, where $c > 1, d > 1$ and $e > 0$.
- $f(x) = c^* \lfloor |x|/\pi \rfloor + d^* cos(|x| \,\%\, \pi + \pi) + e, where\, d > 0, c \geq 2^* d, e \geq d$, and $\lfloor \_ \rfloor$ and % are the floor and modulo operators, respectively.

These instances of functions $f(x)$ can be composed. For example, by composing the third and fourth ones, we can get $f(x) = \log_c(d + e^* | g^* \lfloor |x|/\pi \rfloor + h^* cos(|x| \% \pi + \pi) + i|)$, where $c > 1, d > 1, e > 0, h > 0, g \geq 2^* h, i \geq h$. Moreover, the composite $f(x)$ still satisfies the parameter requirements of the indexing scheme.

An example of order-preserving indexing is shown in Fig. 2. In the example, the input values are integers from -10 to 10 with the sensitivity 1 and the indexing expression is $16^* \log_7(10 + 18^* |x|)^* \text{x} + 317 + noise$. We can check that the order between input values is preserved among indexes.

The indexing scheme can be applied to index numeric values directly. To index strings, we need to convert strings into the numeric values. A simple idea is that a character in the string is converted into its ASCII encoding. For example, "BC" is converted to 0x4243. However, this simple idea may not work since strings are usually compared in the lexical order. For example, the string "BC" is greater than "ABC". If "BC" is converted to 0x4243 and "ABC" is converted to 0x414243, then 0x4243 is less than 0x414243, which is not correct. To index strings, our indexing scheme needs to know the maximum length of strings that will be compared. If the maximum length of input strings is l and a string has the length n, then $(l - n)$ bytes

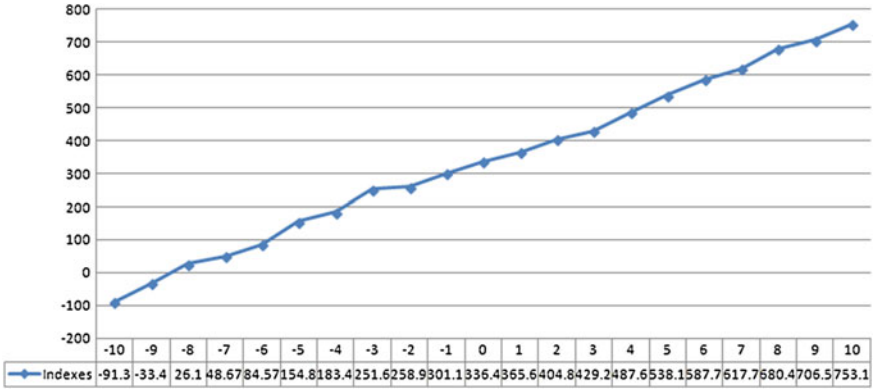


Fig. 2 An Example of order-preserving indexes

of zeros will be padded to the end of the converted integer. For example, if $l = 3$, then "BC" is converted to 0x424300. In addition, the sensitivity of input strings is 1 when they are converted into integers.

5.3 Programmability

The order-preserving indexing scheme allows indexing expressions to be programmed into more complex indexing expressions. In [11], a formal syntax of programming indexing expressions is given. Here, we give intuitive explanation on three programming forms: summation, sequential composition and conditional composition.

- The summation of indexing expressions is represented as $\text{nindex}^{sens}_{[a,b,f]}(v) + \text{nindex}^{sens}_{[a',b',f']}(v)$, which produces the index of v as the sum of $\text{nindex}^{sens}_{[a,b,f]}(v)$ and $\text{nindex}^{sens}_{[a',b',f']}(v)$.
- The sequential composition of indexing expressions is represented as $\text{nindex}^{sens'}_{[a',b',f']}(v); \text{nindex}^{sens}_{[a,b,f]}(v)$, meaning that v is first indexed by $\text{nindex}^{sens}_{[a,b,f]}$, producing an intermediate index, which is then indexed by $\text{nindex}^{sens'}_{[a',b',f']}$, where $sens'$ is the sensitivity of intermediate indexes.
- The conditional indexing expression can be composed in two ways:
  - if v > c then $\text{nindex}^{sens}_{[a,b,f]}(v)$ else $\text{nindex}^{sens}_{[a',b',f']}(v)$, where $\text{nindex}^{sens}_{[a,b,f]}(c) > \text{nindex}^{sens}_{[a',b',f']}(c)$;
  - if v < c then $\text{nindex}^{s}_{[a,b,f]}(v)$ else $\text{nindex}^{sens}_{[a',b',f']}(v)$, where $\text{nindex}^{sens}_{[a,b,f]}(c) < \text{nindex}^{sens}_{[a',b',f']}(c)$.

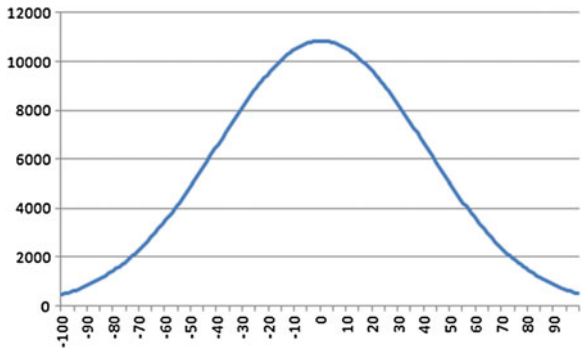


Fig. 3 Plaintexts in Gaussian distribution

The first way means that if v is greater than a constant c, then its index is generated by using the expression $\text{nindex}^{sens}_{[a,b,f]}(v)$, and otherwise, generated by using the expression $\text{nindex}^{sens}_{[a',b',f']}(v)$. The condition $\text{nindex}^{sens}_{[a,b,f]}(c) > \text{nindex}^{sens}_{[a',b',f']}(c)$ ensures that $\text{nindex}^{sens}_{[a,b,f]}(v_1)$ always generates indexes greater than $\text{nindex}^{sens}_{[a',b',f']}(v_2)$ when $v_1 > v_2$, so that the composite indexing expression still satisfies the order-preserving property. Similarly, the second way means that $\text{nindex}^{sens}_{[a,b,f]}(v)$ is used to index v if it is less than c; otherwise, $\text{nindex}^{sens}_{[a',b',f']}(v)$ is used.

Note that these three forms can be mixed in a composite indexing expression. For example, the true branch of a condition indexing expression can be a summation expression, while the false branch can be a sequential expression.

The composite indexing expression contains more secret parameters than its components. Hence, the programmability of indexing expression increases the robustness of the indexing scheme since the forms of indexing expressions are no longer fixed and include more secrets. On the other hand, the programmability of the indexing scheme gives users the capability to unlink the distributions of plaintext values and indexes by indexing plaintexts in different ranges with different expressions. As discussed in [3], it is not secure if the distribution of plaintexts is revealed by the distribution of ciphertexts. In the following, an example borrowed from [11] is used to illustrate how programmability is used to hide the distribution of plaintext values.

Suppose the plaintext values is selected from the range $[-100, 100]$ and their sensitivity is 1. An input value may have 10,000 duplicates. Figure 3 shows the input values in the Gaussian distribution.

Then, the following indexing program is applied to index the plaintext values. By using the conditional composition, the plaintext values are divided into 9 ranges, and processed with different expressions. Figure 4 shows the distribution of the indexes, which is different from the Gaussian distribution of plaintext values.

```
if x > 70 then
   7*(log7(621+12*|x|))*x+1*(*(log12(2030+3*|x|))*x+15683
else if x > 40 then
   17*(log7(1265+8*|x|))*x+7*(log12(621+12*|x|))*x+11706
else if x > 20 then
   25*(log7(6812+78*|x|))*x+17*(log12(1265+8*|x|))*x+8324
else if x => 0 then
   30*(log7(9168+38*|x|))*x+25*(log12(6812+78*|x|))*x+6983
else if x > −20 then
   25*(log7(7523+73*|x|))*x+30*(log12(9168+38*|x|))*x+6983
else if x > −40 then
   20*(log7(8211+31*|x|))*x+25*(log12(7523+73*|x|))*x−6121
else if x > −60 then
   12*(log7(4366+13*|x|))*x+20*(log12(8211+31*|x|))*x−3676
else if x > −80 then
   5*(log7(6723+7*|x|))*x+12*(log12(4366+13*|x|))*x−93
else
   1*(log7(2030+3*|x|))*x+5*(log12(6723+7*|x|))*x−3492
```

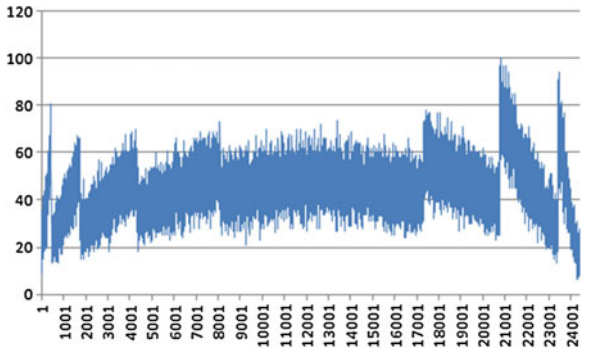


Fig. 4 Distribution of indexes

6 Homomorphic Encryption

Homomorphic encryption allows operations on plaintext values to be performed through operations on ciphertexts. Thus, if table columns in a database are encrypted homomorphically, then the aggregate queries of using SUM and AVG can be directly performed over encrypted table columns by the existing DBMSs.

6.1 Homomorphism

A homomorphic encryption scheme can be fully homomorphic or partially homomorphic. In a fully homomorphic encryption scheme, both additions and multiplications

can be performed over ciphertexts, while in a partially homomorphic encryption scheme, only additions or multiplications can be performed over ciphertexts.

Fully homomorphic encryption is a dream of cryptographic research. Though some fully homomorphic encryption schemes have been proposed [8], there is still no work on the practical applications of these schemes. This is partially caused by the gap between the cryptographic primitives and practical applications. For example, in [18], the plaintext values can only be a bit 0 or 1, while the data in practical applications usually consists of a sequence of bits (e.g., an integer of 32 bits). It is not efficient to encrypt each bit separately. Consequently, the current fully homomorphic encryption schemes are not practical enough to be applied to encrypt databases.

Partially homomorphic encryption can be practical. For example, the widely used RSA is a multiplicatively homomorphic encryption scheme. That is, suppose v_1' and v_2' is the RSA encryption of two plaintexts v_1 and v_2 with the same public key. Then, the decryption of $v_1' * v_2'$ with the corresponding private key is the result of $v_1 * v_2$. However, multiplicatively homomorphic encryption is not useful for performing aggregate queries of SUM and AVG, since these queries need the summation of plaintext values, not their multiplication. Instead, an additively homomorphic encryption scheme is more useful for these aggregate queries.

For an additively homomorphic encryption scheme, it is desirable for queries that the sum of plaintext values can be decrypted from the sum of corresponding ciphertexts. Thus, in order to get the sum of one table column, the values in the encrypted column can be added by the existing DBMSs. Some additively homomorphic encryption schemes do not satisfy this requirement. For example, in the homomorphic encryption scheme [15], the sum of plaintext values is obtained through the multiplication of ciphertexts. This encryption scheme is used by the database encryption systems [7, 16], where the existing DBMSs have to be extended to deal with aggregate queries involving SUM.

The modern encryption algorithms are usually built over finite algebraic structures enforced by using the modulo operation. However, this finiteness requirement is harmful for homomorphic encryption over databases. For example, an additively homomorphic encryption scheme is proposed in [12]; in this scheme, if the sum of plaintexts is greater than the modulus, then scheme is no longer homomorphic. Consequently, it is hard to use this scheme in database encryption, since it is hard to determine the maximum sum of values in a table column for a long-standing database.

6.2 A Homomorphic Encryption Scheme Without Modulus

We have proposed a generic scheme of defining homomorphic encryption without using modulo operations [10]. The scheme supports both additive and multiplicative homomorphism and allows values from an infinite algebraic structure to be encrypted. Here, we introduce one instance of this scheme.

Let *Enc* be the encrypting operation, *Dec* the decrypting operation and $K(n)$ the key. Then, given a value v, the encryption $Enc(K(n), v)$ will generate a ciphertext $(c_1, \ldots, c_n)$, which consists of n subciphertexts $c_1, \ldots,$ and c_n. The parameter n in a key indicates the number of subciphertexts to be generated. In decryption, the operation $Dec(K(n), (c_1, \ldots, c_n))$ will return v. Given another value v', let $Enc(K(n), v') = (c_1', \ldots, c_n')$. Then, the scheme ensures $Dec(K(n), (c_1 + c_1', \ldots, c_n + c_n')) = v + v'$ for additive homomorphism.

The multiplication of v and v' can be decrypted in two steps from the outer product of $(c_1, \ldots, c_n)$ and $(c_1', \ldots, c_n')$, which is represented as $(c_1 * c_1', \ldots, c_n * c_1', \ldots, c_1 * c_n', \ldots, c_n * c_n')$. At the first step, we perform the decryption $Dec(K(n), (c_1 * c_i', \ldots, c_n * c_i'))$ for $i \leq 1 \leq n$ to produce the intermediate ciphertext $(v * c_1', \ldots, v * c_n')$. At the second step, we get $v * v'$ from $Dec(K(n), (v * c_1', \ldots, v * c_n'))$. Specially, given a real number h, we have $Dec(K(n), (h * c_1, \ldots, h * c_n)) = h * v$.

In this instance, the key $K(n)$ is a list of n tuples of real numbers, $[(k_1, s_1, t_1), \ldots, (k_n, s_n, t_n)]$, where $n \geq 3$, $t_i \neq 0 (1 \leq i \leq n-1)$, $\sum_{i=1}^{n-2} k_i \neq 0$, and $k_n + s_n + t_n \neq 0$. The operation *Enc* encrypts v into $(c_1, \ldots, c_n)$ by the following steps.

- Let $r_1, \ldots, r_{n-1}$ be $n-1$ random numbers;
- $c_i = t_i * k_i * v + s_i * r_{n-1} + t_i * r_i$ for $1 \leq i \leq n-2$;
- $c_{n-1} = k_{n-1} * t_{n-1} * \sum_{i=1}^{n-2} r_i + s_{n-1} * r_{n-1}$;
- $c_n = (k_n + t_n + s_n) * r_{n-1}$.

The operation *Dec* decrypts the ciphertext $(c_1, \ldots, c_n)$ into v by the steps below. If the keys are correct, we can see the random noises in each subciphertexts are counteracted, and hence the correct value v is returned. Unlike the methods in [6, 12], we do not use the modulo operation to remove noises, so the presented encryption scheme can be applied to infinite data ranges.

- $L = \sum_{i=1}^{n-2} k_i$;
- $S = c_n / (k_n + t_n + s_n)$;
- $I = c_{n-1} - S * s_{n-1}$;
- $v = \sum_{i=1}^{n-2} (c_i - S * s_i) / (L * t_i) - I / (L * k_{n-1}^* * t_{n-1})$.

The last step of decryption divides different $c_i - S * s_i$ $(1 \leq i \leq n-2)$ with different secret values $L * t_i$. Thus, if an adversary wants to recover v from ciphertexts in brute-force, then he needs to guess t_i $(1 \leq i \leq n-2)$ for each subciphertexts, in addition to guessing other secrets $s_i (1 \leq i \leq n-1)$, L, $k_{n-1} * t_{n-1}$, and $k_n + t_n + s_n$.

6.3 Composition of Homomorphic Encryption

The generic scheme in [10] allows multiple instances to be defined. These instances can be composed into new instances, which are still homomorphic. Briefly, the composition can be achieved by encrypting each subcipertext from one instance again by using the same or another instance.

Suppose there are two homomorphic encryption instances. The first one has the key $K_1(n)$, the encryption operation Enc_1, and the decryption operation Dec_1, and the second one has the key $K_2(m)$, the encryption operation Enc_2, and the decryption operation Dec_2. For the composition of the first and second instances, a value v will be encrypted into $m*n$ subciphertexts, as shown below.

- $Enc_1(K_1(n), v) = (c_1, \ldots, c_n)$;
- $Enc_2(K_2(m), c_i) = (c_{i1}, \ldots, c_{im})$ for $1 \leq i \leq n$;
- The final ciphertext is $(c_{11}, \ldots, c_{1m,\ldots,}c_{n1}, \ldots, c_{nm})$.

To decrypt the ciphertext $(c_{11}, \ldots, c_{1m,\ldots,}c_{n1}, \ldots, c_{nm})$, the following steps are taken.

- $Dec_2(K_2(m), (c_{i1}, \ldots, c_{im})) = c_i$ for $1 \leq i \leq n$;
- $Dec_1(K_1(n), (c_1, \ldots, c_n)) = v$.

A composed homomorphic encryption scheme is more robust than its component instances. Suppose an adversary wants to recover a plaintext from a ciphertext generated by a composed scheme. Then, in addition to breaking each component instance, he needs to guess how to split subciphertexts, so that each subgroup of subciphertexts can be correctly decrypted by using Dec_2 into correct intermediate subciphertexts, which are then decrypted by Dec_1.

In the architecture of managing encrypted databases, the database service providers cannot know whether a ciphertext is generated by a composed homomorphic encryption scheme, since they cannot access the query proxy. This increases the difficulty for them to perform brute force attacks on stored ciphertexts.

6.4 Examples of Homomorphic Encryption

We use examples to illustrate the homomorphic encryption instance. The following key is supposed to be used.

$$[(6.03, 74.99, 94.17), (-56.60, 13.07, 32.45),$$
$$(76.11, 71.69, 34.48), (29.87, 32.70, 92.80)]$$

This key consists of four tuples, meaning that a value will be encrypted into a ciphertext that has four subciphertexts. A key component can be either a positive real or a negative real. The plaintext values in this example are five reals: 1,384.4, 1,384.4, 345.3, 9,233.9 and 563.21. Using the thomomorphic encryption instance, we get five ciphertexts listed in Table 1, with each having four subciphertexts.

Note that the first two values are encrypted into different ciphertexts, though they are the same. The noises used in the encryption are listed in Table 2, with each row containing the noises for encrypting the corresponding value. They can be used to verify the correctness of the encryption operation.

Table 1 Example of ciphertexts

(858839.59014,	−2536268.23010,	2119402.922028,	14817.6369)
(848724.37914,	−2511656.80840,	3298624.388448,	41558.3676)
(309553.73793,	−600157.07450,	4945540.441476,	49011.4665)
(5331366.24729,	−16941668.30340,	3738747.931116,	8301.4191)
(407243.945071,	−1005986.464300,	2392536.630136,	118441.6584)

Table 2 Example of noises

702.25	102.76	95.37
457.78	791.88	267.48
953.82	922.10	315.45
891.31	531.91	53.43
321.35	569.52	762.32

Applying the decryption operation to the ciphertexts, we can get the correct plaintexts. Moreover, the sum of plaintext values 12,909.21 can be obtained by decrypting the following sum of ciphertexts, which is obtained by adding the corresponding subciphertexts in each ciphertext.

$$(7755727.899571, -23595736.880700, 16494852.313204, 232130.5485)$$

The average of plaintexts 2,581.84 can also be correctly decrypted from the following average of ciphertexts, which is obtained by averaging the corresponding subciphertexts in each ciphertext.

$$(1551145.579914, -4719147.376140, 3298970.4626408, 46426.1097)$$

7 Translation of SQL Queries

A database schema designed by application developers are created differently in an encrypted database. We first describe the table structures in an encrypted database and then introduce how to translate a query from an application into a query that can be executed over the encrypted database.

7.1 Table Structures

A table designed by application developers may include multiple columns. In the presented approach, each column is processed independently. Hence, we take a table

that contains only one column as the example to explain the change of table structures in the encrypted database. Suppose a table *Staff* has been designed for an application with one column *Salary*. When creating such a table in a database, the query proxy hashes the table name, such that the table name is meaningless to the untrusted database service providers.

For the column *Salary*, the proxy actually creates multiple columns in the encrypted table. The number of columns depends on the number of subciphertexts generated by the homomorphic encryption scheme. Assume the homomorphic encryption scheme is configured in the query proxy to generate n subciphertexts for the *Salary* column. Then, there will be $n + 2$ corresponding columns created for the *Salary* column. The names of these $n + 2$ columns are obtained by hashing names *SalaryEqIdx*, *SalaryRngIdx*, $SalaryEnc_1$, …., and $SalaryEnc_n$. In these names, *EqIdx*, *RngIdx* and Enc_i are postfixes appended by the query proxy. Figure 5 shows the *Staff* table structure designed by application developers and the table structure managed by the cloud database service, where the notation $Staff'$ represents the hash of the name *Staff*, and similarly for other hashed names $SalaryEqIdx'$, $SalaryRngIdx'$, $SalaryEnc'_1, \ldots$, and $SalaryEnc'_n$.

Note that the n subciphertexts $SalaryEnc'_1$, …, and $SalaryEnc'_n$ can be stored not necessarily in the order of subciphertexts generated from encryption. For example, we can store the subcipertexts in the order $SalaryEnc'_2$, …, $SalaryEnc'_n$, and finally $SalaryEnc_1\prime$. Moreover, the subcipertexts of one value can be mixed with the subcipertexts of another value in the same record. Thus, the adversary is hard to know whether two subcipehrtexts come from the encryption of one value.

When a salary from the database application is being put into the encrypted table, the proxy produces $n + 2$ values for the corresponding columns $SalaryEqIdx'$, $SalaryRngIdx'$, $SalaryEnc'_1$,…, and $SalaryEnc_n\prime$, by using the hash algorithm like HMAC-SHA1, the order-preserving indexing scheme and the homomorphic encryption scheme. The columns $SalaryEqIdx'$ and $SalaryRngIdx'$ are used to process query conditions involving equality and range comparisons, and when the query conditions are satisfied the values in the n columns $SalaryEnc1'$, …, and $SalaryEncn'$ will be returned to decrypt. Note that if values in a column cannot be added or averaged, we also can use other encryption schemes like AES to encrypt this column.

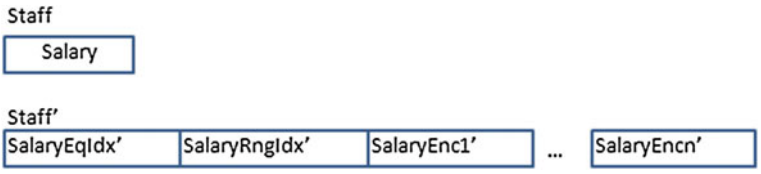


Fig. 5 Change of table structures

7.2 Query Translation

The query translation relies on some meta data. Assume the proxy has the key $K(n)$ for homomorphic encryption, a key k for secure hashing and indexing. An indexing expression is denoted by *Index*(v, s), meaning that v is indexed by *Index* with the sensitivity s. In the following, we assume the sensitivity is *sens*. Note that different columns may use different keys and indexing expressions. The numerical and string data types are represented by Num and String, respectively.

7.2.1 Creation of Databases and Tables

To create a database and a table, the database application can issue the following two statements.

create database *dbname*
create table *tblname* (*colnm Type*,...)

After receiving the above statements, the query proxy translates them into the following ones, which will be executed by the cloud database service. The original schema is recorded by the query proxy in its meta data, where *Hash* represents a secure hash algorithm like HMACSHA1.

create database Hash(*k*,*dbname*)
create table Hash(*k*,*tblname*) (Hash(k, *colnm*+"EqIdx") String, Hash(*k*, *colnm*+"RngIdx") Num,
Hash(k,*colnm*+"Enc1") Num,..., Hash(k,*colnm*+"Enc*n*") Num,…)

The new columns have different data types. The column *colnm*+"EqIdx" have the type String, since its values are always hexadecimal strings generated by the secure hash function. The values of column *colnm*+"RngIdx" are generated by the indexing mechanism and have the numerical type. The columns *colnm*+"Enc1",…, and *colnm*+"Enc*n*" for subciphertexts have the type Num, so that they can be summed or averaged by the DBMSs. Strings can be converted into numeric values before using the homomorphic encryption scheme, or they can be encrypted with other encryption schemes like AES, since it is not meaningful to perform addition operations over strings.

7.2.2 Data Insertion

After a table is created, the database application can put a new record into the table by using the following statement.

insert into *tblname* (*colnm*,...) values (*v*,...)

For this statement, the query proxy translates it into the following one. In the new statement, the value v is hashed, indexed and encrypted before being stored into the encrypted table. The encryption of v using the homomorphic encryption scheme with the key $K(n)$ produces n subciphertexts c_i $(1 \leq i \leq n)$.

insert into Hash(k, tblname)(Hash(k, colnm + "EqIdx"), Hash(k, colnm + "RngIdx"),
Hash(k, colnm + "Enc1"), ..., Hash(k, colnm + "Enc$n$"), ...)
values (Hash(k, v), Index(v, sens), c_1, ..., c_n, ...)

7.2.3 Queries of Data Selection

The data selection queries select one or more columns from a table. The following two forms of query statements can be used to select the column *colnm* or all columns (indicated by *) from the table *tblname* under the condition *cond*.

select *colnm*,... from *tblname* where *cond*
select * from *tblname* where *cond*

The second form can be changed into the first one by replacing * with all column names according to the table schema maintained by the query proxy. For the first form, the query proxy translates it into the following one, where the translation of *cond* into *cond*′ is discussed below.

select Hash(k, *colnm* + "Enc1″), ..., Hash(k, *colnm* + "Enc$n$″), ...
from Hash(k, *tblname*) where *cond*′

In the new query, all subciphertexts must be selected, so that the query proxy can perform the decryption. For the condition *cond*, it is defined over the primitive logical forms *colnm* < *c*, *colnm* = *c*, *colnm* > *c*, where *c* is a constant from the domain of the *colnm* column, by using the logical connectives. When translating the condition *cond*, we replace each primitive logical form with a translated one, as defined below.

The condition *colnm* < *c* is translated into *Hash*(*k*,*colnm*+"RngIdx") < *Index*(*c*,0). Note that *Index*(*c*, 0) is the minimum index of *c*, since no noise is added. The condition *colnm*=*c* is simply translated into *Hash*(*k*,*colnm*+"EqIdx") = *Hash*(*k*,*c*). Assume the sensitivity of values in the *colnm* column is *sens*. Then, *c*+*sens* is the next value of *c*, and *colnm* > *c* is equivalent to the new condition *colnm* ≥ *c* + *sens*, which is translated into *Hash*(*k*,*colnm*+"RngIdx") ≥ *Index*(c+sens,0). Again, *Index*(*c*+*sens*,0) is the minimum index of *c*+*sens*.

In addition, the keywords *order by colnm* and *group by colnm* might be used in queries. They are translated into *order by Hash*(*k*,*colnm*+"*RngIdx*") and *group by Hash*(*k*,*colnm*+"EqIdx"), respectively. That is the ordering comparisons are performed over the columns produced with the order-preserving indexing scheme, and the grouping operation replies on the columns that support equality comparison.

7.2.4 Aggregate Queries: SUM and AVG

The values in a table column can be calculated for their sum and average. The following query statement can be used for this purpose.

select SUM(*colnm*),... from *tblname* where *cond*

select AVG(*colnm*),… from *tblname* where *cond*

In the homomorphic encryption scheme, the sum or average of ciphertexts are performed on each subcipertexts. Hence, the above statements are translated into the following ones.

select SUM(Hash(*k*,*colnm*+"Enc1")),..., SUM(Hash(*k*,*colnm*+"Enc*n*")), …

from Hash(*k*,*tblname*) where *cond′*

select AVG(Hash(*k*,*colnm*+"Enc1")),..., AVG(Hash(*k*,*colnm*+"Enc*n*")), …

from Hash(*k*,*tblname*) where *cond′*

After receiving the sum or average of subciphertexts, the query proxy can decrypt them into the expected sum or average of values in the *colnm* column. The translation of *cond* is the same as that in the data selection queries.

7.2.5 Aggregate Queries: MAX and MIN

The maximum or minimum value in a column might be queried by using the following queries.

select MAX(*colnm*) from *tblname* where *cond*

select MIN(*colnm*) from *tblname* where *cond*

Each of the above queries is translated into two queries.

select MAX(Hash(*k*,*colnm*+"*RngIdx*")) from Hash(*k*,*tblname*) where *cond′*

select Hash(*k*,*colnm*+"Enc1"),..., Hash(*k*,*colnm*+"Enc*n*"), …

from Hash(*k*,*tblname*) where *cond′* and Hash(*k*,*colnm*+"*RngIdx*") = *max*

select MIN (Hash(*k*,*colnm*+"*RngIdx*")) from Hash(*k*,*tblname*) where *cond′*

select Hash(*k*,*colnm*+"Enc1"),..., Hash(*k*,*colnm*+"Enc*n*")), …

from Hash(*k*,*tblname*) where *cond′* and Hash(*k*,*colnm*+"*RngIdx*") = *min*

The first queries determine the maximum index or the minimum index. After getting the maximum index (*max*) or the minimum index (*min*), the query proxy then constructs the second queries to get back the subciphertexts corresponding to *max* or *min*. From the subciphertexts, the maximum or minimum values can be decrypted. Note that we cannot get the maximum or minimum values from the maximum or minimum indexes, since the order-preserving indexing scheme is one-way.

8 Implementation and Evaluation

We implemented a prototype of querying encrypted databases with the SQL Server 2008 as the underlying DBMS. Since the query proxy communicates with the DBMS in standard SQL queries, we can change to a cloud database service just by changing the communication channel (e.g., the IP address of the cloud database science, the port of TCP or UDP, and the secrets for being authenticated).

In this prototype, the database application stores person information in an encrypted database. The person table designed by the application developers has the following schema.

person(id int, name varchar(64), gender varchar(8), birthdate bigint, income numeric(10,2))

A fragment of the encrypted person table is shown in Figure 6 by using the Microsoft SQL Server Management Studio. There are six columns in Figure 6, which are generated from the processing of person incomes, corresponding to the hashes of incomes, their order order-preserving indexes and four subcipehrtexts of encrypting each income. The attribute names are hashed, as shown at the first row in Figure 6. Thus, from this encrypted table, the untrusted database service provider cannot get any meaningful information. For other attributes of the original person table (i.e., id, name, gender and birthdate), they are encrypted with AES, since they cannot be meaningfully added or averaged with the SUM or AVG operations in a query. In addition, the gender attribute can only be "Male" or "Female", and there are no meaningful range queries for this attribute. Hence, the order-preserving indexes are not generated for this attribute.

The order-preserving indexing in the encrypted person table is performed with the following expression. This expression is kept secret in the query proxy. Due to the programmability of the order-preserving indexing scheme, the form of the following indexing expression is not known. It brings difficulty for an adversary to effectively guess the indexing expression even if the adversary happens to know some pairs of plaintexts and indexes.

$$3754.3^{*}\log_{120.2}(513.8 + 77543.32^{*}|(3187.2^{*}\lfloor|x|/\pi\rfloor + 196.2^{*}\cos(|x|\%\pi + \pi) + 26867.3)|)^{*}x + 84648.87)$$

As described above, an income value is encrypted into 4 subciphertexts, so the key should have the form $[(k_1, s_1, t_1), (k_2, s_2, t_2), (k_3, s_3, t_3), (k_4, s_4, t_4)]$. In this evaluation, each k_i or t_i is allowed to have 4 digits, and each s_i to have 8 digits. Thus, the 4 subciphertexts of homomorphic encryption leads to a key space of size 10^{52} (i.e., 10^{52}=10^4*10^8*(10^8*10^4*10^8*10^4*10^8*10^4*10^4), which is the product of the space sizes of L, S, s_1, t_1, s_2, t_2, s_3, t_3, k_3). Other attributes other than income are encrypted with AES 128, which has key space of size 2^{128}.

The performance of querying encrypted databases is tested with respect to data insertion and query on a Dell Latitude E4310 laptop. To test the performance of data insertion, we generate 10,000 person records and insert them into a plain database (PlainDB), where data is not encrypted, and an encrypted database (EncDB), respectively. Figure 7 shows the used time (milliseconds) after inserting every 2,000 records. Compared with the insertion to PlainDB, the insertion of 10,000 records to EndDB takes about more 22.9 % time.

The insertion to EncDB involves four different cryptographic schemes: the hash algorithm (HMACSHA1), the order-preserving indexing scheme, the AES encryption and the homomorphic encryption. Figure 8 shows the time taken by each of these schemes. From this figure, we can see the HMACSHA1 algorithm takes more time than the other three schemes in total. Actually, if we change the order-preserving indexing scheme into a deterministic one by avoiding noises in indexes, then the equality check can be carried out over the indexes, too. At this case, the values in encrypted databases do not need to be hashed, and hence the performance of insertion will be increased.

The query performance is tested on two types of queries. The first type is to select records satisfying some conditions, while the second is an aggregate query using the SUM operation. We use the query below to select records from the encrypted database.

$$\text{select} * \text{from person where income} > \text{min and income} < \text{max}$$

By changing *min* and *max*, we can get five different results, including correspondingly 2,000, 4,000, 6,000, 8,000 and 10,000 records. The time spent on querying PlainDB and EncDB is shown in Fig. 9, from which we can see the performance overhead is linearly increased with the increase of the number of records in the query result. This increase of performance overhead is reasonable, since more records in the encrypted query result need more time to decrypt.

The aggregate query is performed by the following statement, which sums the income of persons satisfying the query conditions.

$$\text{select} * \text{SUM(income) from person where income} > \text{min and income} < \text{max}$$

Cc825d5bbb9aa59f4116303b1cbb4a4fd1c42...	C0ff0c7ac86...	Cc7ec26f81c94e47...	Cd625506ed757c43a...	C2092f10c44be9b4e...	C77df75d77d0f6dbf187
0xB26188E597BA04E117DB4287BF881265...	19772809.11	13049408114.23000	162111443534.48000	79202642383.96000	109932430413.04000
0xB9CE25F77AF8E77EF85E6E338CF4792B...	19781138.13	42946303689.76500	533779391613.01400	259356433621.51000	361969020131.28000
0xEC57E4156A2459DA3C530C1124B95B2E...	19787185.05	29666687091.13000	368732737115.10800	179428528457.21000	250045883035.36000
0xA8DEBC64F8DF4B76CEB747C23B2B51D...	19794019.65	2840338453.90500	35252591912.08200	17422421669.59000	23906376102.24000
0x746FF4D07895818DAFD0F31A00B257D1...	19798704.95	6652840774.45000	82623442536.41600	40665364177.93000	56029104294.80000
0x5A5D689702DF447CF0329628C4EB2C56...	19810175.00	29132989883.55500	362061314136.07000	176131800381.27000	245522905415.76000
0xC9E56B09922F2004B79A71C1F5F89FE45...	19814487.46	37566051837.71000	466873579866.96400	227267403350.93000	316597781046.00000
0x5D66C25692CC79AD0D833FE2265EE705...	19823026.95	21112541829.03500	262402816182.17800	127373261211.88000	177942844899.60000
0xCC78DAF3A2A87A68536C232616098922...	19827396.27	18847491820.73000	234191785910.27200	114158141190.95000	158811693496.64000
0xCC219DC5150E4BBBE2AA1ADA1FF5965...	19839157.75	35856416800.84500	445634418153.96600	216568728060.45000	302196262387.04000

Fig. 6 A fragment of encrypted person table

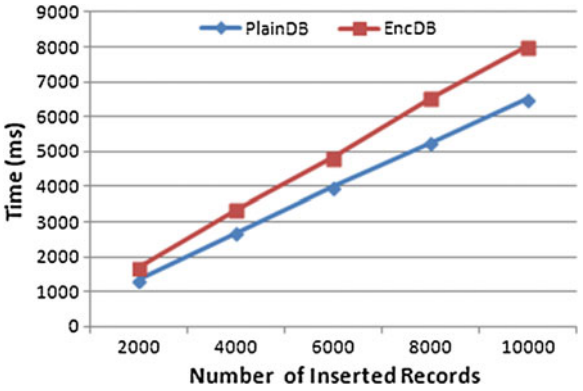
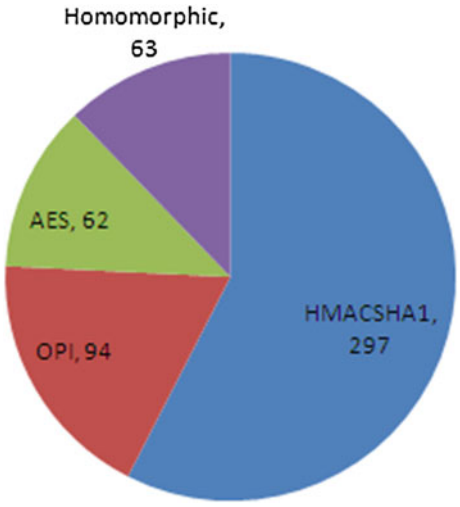


Fig. 7 Performance of insertion

Fig. 8 Performance of different cryptographic schemes



We still let the query return five different results, corresponding to the income sums of 2,000, 4,000, 6,000, 8,000 and 10,000 records. Figure 10 shows the performance result. This result shows that the query time does not increase quickly as that in the selection queries with the increase of records included in the query results. This is because the result of the above aggregate query has only one value (the sum of encrypted income in each person record satisfying the condition) to decrypt, regardless of the number of aggregated records.

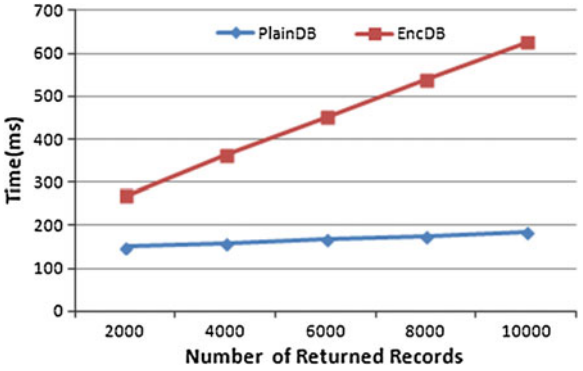


Fig. 9 Query performance for different records number

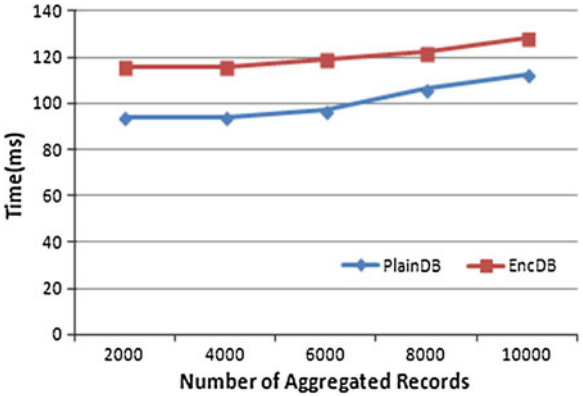


Fig. 10 Performance of aggregate queries

9 Summary

In this chapter, we introduce an approach for database encryption and query. In particular, we introduce an order-preserving indexing scheme and a homomorphic encryption scheme. Compared with the existing order-preserving and homomorphic encryption schemes, the presented schemes are more suitable for long-standing database, since they do not need users to predetermine the number and range of data stored in databases and their maximum sums. We implement a prototype that uses the exiting DBMS (i.e., Microsoft SQL Server 2008) and evaluate its performance. The evaluation shows that the approach incurs acceptable performance overhead.
The approach cannot deal with all SQL queries. For example, it cannot support queries that use conditions involving operations of several columns (e.g., $number*rate>10$). It is an interesting problem of improving the system to make it support more types of SQL queries in future.

References

1. Amazon Relational Database Service. http://aws.amazon.com/rds/
2. Agrawal D, Abbadi A, Emekçi F, Metwally D (2009) Database management as a service: challenges and opportunities. In: Proceedings of the 25th international conference on data, engineering, pp 1709–1716
3. Agrawal R, Kiernan J, Srikant R, Xu Y (2004) Order preserving encryption for numeric data. In: Proceedings of the 2004 ACM SIGMOD international conference on management of data, SIGMOD'04, pp 563–574
4. Bertino E, Sandhu RS (2005) Database security-concepts, approaches, and challenges. IEEE Trans Dependable Secure Comput 2(1):2–19
5. Boldyreva A, Chenette N, Lee Y, O'Neill A (2009) Order-preserving symmetric encryption. EUROCRYPT, pp 224–241
6. Brakerski Z, Vaikuntanathan V (2011) Fully homomorphic encryption from ring-LWE and security for Key dependent messages. CRYPTO 2011, pp 505–524
7. Ge T, Zdonik S (2007) Answering aggregation queries in a secure system model. In the 33rd international conference on very large data bases, pp 519–530
8. Gentry C (2009) Fully homomorphic encryption using ideal lattices. STOC 2009, pp 169–178
9. Hore B, Mehrotra S, Tsudik G (2004) A privacy-preserving index for range queries. VLDB 2004, pp 720–731
10. Liu D (2012) Homomorphic encryption for database querying. Australian Provisional Patent 2012902653 (filed by CSIRO)
11. Liu D, Wang S (2013) Nonlinear order preserving index for ncrypted database query in service cloud environments. Concurrency Comput: Pract Experience (in press)
12. Naehrig M, Lauter K, Vaikuntanathan V (2011) Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on cloud computing security workshop, CCSW'11, pp 113–124
13. Microsoft SQL Azure Database. http://www.windowsazure.com/en-us/home/features/data-management/
14. Oracle Database Cloud Service. https://cloud.oracle.com
15. Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. EUROCRYPT 1999, pp 223–238
16. Popa RA, Redfield CMS, Zeldovich N, Balakrishnan H (2011) CryptDB: protecting confidentiality with encrypted query processing. In: Proceedings of the 23rd ACM symposium on operating systems principles, SOSP'11, pp 85–100
17. Shamir A (1979) How to share a secret. Commun ACM 22(11):612–613
18. van Dijk M, Gentry C, Halevi S, Vaikuntanathan V (2010) Fully homomorphic encryption over the integers. EUROCRYPT 2010, pp 24–43

Trust Model for Cloud Systems with Self Variance Evaluation

Xiaofeng Wang, Jinshu Su, Xiaofeng Hu, Chunqing Wu and Huan Zhou

1 Introduction

The open, distributed, and dynamic nature of the cloud poses great challenges for trust establishment between entities. In these applications, the service consumer usually knows little about the trust or the reliability of the service provider. To mitigate the risks of the consumers, reputation-based trust systems [7, 10] are deployed as a popular approach to predict how much the service provider can be trusted, and they play a pivotal role in aggregating, filtering, and ordering information for consumers. As an important security enhancement technology [4, 8], many reputation (social trust) models have been proposed for different applications such as: decentralized overlay networks and applications [1, 16, 26], multi-agent systems [24, 25, 28], social web services [6, 9, 12] and recommender systems [3, 21].

Reputation is a statistical value about the trust probability derived from the behaviour history. Usually, the reputation is based on the interactions directly with the evaluator (personal experience) or as recommended by others (feedback) [10]. From the personal experience's perspective, most existing work used the simple average [17], the Bayesian [25, 29] or the belief models [24, 28] to quantify the trust as some statistical values. However, they ignore another important attribute of the predicted statistical value, namely the prediction variance (or prediction accuracy), which depicts how much the trust prediction may deviate from the real one. For example, a service provider has a service success probability of 0.9. But due to the incomplete personal experience, a customer quantifies the provider's trust as 0.7. By using existing trust models, the customer cannot assess the accuracy of the reputation prediction made by itself. Hence, he cannot decide how much to rely on the prediction to make a decision. Moreover, when the customer recommends this trust prediction to others

X. Wang (✉) · J. Su · X. Hu · C. Wu · H. Zhou
School of Computer, National University of Defense Technology,
Hunan 410073, Changsha, China
e-mail: xf_wang@nudt.edu.cn

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_10,

as a feedback, he cannot give any suggestion about how to aggregate the feedback so that others can minimize the deviation of their trust evaluation.

To aggregate feedbacks recommended by others, the summation method is widely applied in reputation systems such as eBay [17] and Eigentrust [11]. However, it is easy to be manipulated by malicious nodes for their personal profits [20]. A malicious node can falsely improve its own reputation or degrade the reputations of others. Therefore, malicious participants can obtain unwarranted service or honest participants can be prevented from obtaining service. As a measure to defend malicious feedbacks for the summation method, most existing work weighted the feedbacks by considering their credibility, such as the trust value based credibility used in Eigentrust [11] and the personalized similarity based credibility used in PeerTrust [26]. However, these credibility techniques usually need wide trust knowledge of the system [11, 20, 26] or manually tuned intuitive parameters [25, 28], which are sometimes unrealistic assumptions in a real world application. We believe that the difficulty of applying these summation-based credibility techniques is due to their foundation's lack of robustness. In other words, the intuitive summation method lacks the support for robustness to resist malicious feedbacks.

In this chapter, we present a general trust model based on linear Markov prediction to get a more comprehensive and robust reputation evaluation. The main contributions of our work are as follows:

1. In contrast to existing feedback based reputation trust models, our RLM model represents the reputation trust by two attributes: reputation value and reputation prediction variance. The model is tracked by a linear hidden Markov process, so that a more comprehensive and accurate reputation can be evaluated. The assessment of the reputation prediction variance can help to achieve a better local decision making as well as a more intelligent third-party reputation aggregation.
2. We propose the Kalman aggregation method for feedback aggregation instead of using the intuitive summation method. Our Kalman aggregation method can adjust the influence of a malicious feedback by the parameter of estimated feedback variance, which is used to support our robust trust evaluation techniques.
3. To defend against malicious feedback attacks for RLM model, we design and demonstrate a robust model parameter calibration method. We introduce the Expectation Maximization (EM) algorithm to autonomously calibrate the model parameters, and then propose the hypothesis test method to estimate the feedback deviation, which can filter out malicious feedbacks half automatically with a statistical significance level.
4. To defend against malicious feedbacks automatically without any prior experience, we further propose a self-adaptive WRLM trust model, which extends the original RLM model with the weighted prediction variance. The WRLM can self-adaptively and automatically calibrate the weight factor, so that malicious feedbacks will have high estimated prediction variance, making it have less or even no influence in the final reputation evaluation.

With the Kalman feedback aggregation, our RLM and WRLM trust models enable an evaluator to assess the accuracy of a reputation prediction made by itself, which

can inherently support further robust reputation evaluation methods. For the standard RLM model, the proposed model calibration method can filter out malicious feedbacks precisely. Moreover, our weighted RLM trust model can self-adaptively and automatically mitigate the influence of malicious feedbacks without any prior experience. In the chapter, we give both theoretical proof and experiments to demonstrate the validation, accuracy and robustness of the RLM and WRLM models. With a firm basis in the statistics inference theory, our RLM and WRLM trust models supply a new way to construct a robust reputation system for open service-oriented environments.

The remainder of this chapter is organized as follows. Section 2 introduces background. Section 3 describes the standard RLM trust model and Kalman feedback aggregation. Section 4 introduces the malicious feedback filter for RLM model using EM and hypothesis test methods. Section 5 extends the RLM model to be a robust and self-adaptive WRLM trust model. Experimental results are presented in Sect. 6, followed by the discussions and conclusion in Sect. 7.

2 Background

In open service-oriented environments, reputation based trust systems can determine how much an unknown service provider can be trusted in future interactions. As shown in Table 1, we summarize the characteristics of some existing trust models according to their function. Usually, the reputation/trust value can be modeled by two parts: the direct trust value from the evaluator and the feedbacks from others [10]. To measure the direct trust, Song et al. [19] used the fuzzy logic to compute the reputation score, which is the trust index's numerical value derived from some rules. The Bayesian reputation [29] computes the trust value according to the beta probability density functions (PDF). The posteriori reputation value is decided by $\alpha+1/\alpha+\beta+2$, where α and β are two parameters denoting the number of positive and negative results. Wang and Singh [24] modelled the reputation as a three dimension belief (b, d, u), representing the probabilities of positive, negative and uncertain outcomes. Huang et al. [9] proposed a method to aggregate heterogeneous social networks and used the enhanced topology of trust graph to predict the reputation. All these models quantify the trust as some predicted probability values. However, they ignore the prediction variance, which is one of the two attributes of a statistical prediction (i.e. [14]). Hence, these trust models cannot assess the accuracy of a reputation prediction made by itself. In contrast, the reputation prediction variance is considered in our RLM model to give a more comprehensive and accurate reputation evaluation, and both the reputation value and its prediction variance are tracked by our reputation filter.

To aggregate reputation feedbacks, the summation method [11, 26, 30] is widely used. The simplest summation method is to sum the number of positive ratings and negative ratings separately like eBay [17]. Combined with different system architectures, the summation method can have different forms. For example, in P2P systems,

Table 1 Related work

Trust model	Direct trust evaluation	Feedback aggregation	Malicious feedback defense	Self-variance assessment	Mal-feedback detection	
					System wide knowledge	Manually tuned parameter
Song [19]	Fuzzy logic	Rule based aggregation	No	X	X	X
Zhang [29]	Bayesian theory	Beta aggregation	Euclidean distance	No	Need	No
Wang [24]	Belief model	Statistical theory	X	No	X	X
Huang [9]	X	Social networks	X	No	X	X
Eigen trust [11]	Normalized satisfaction	Summation	Trust value based credibility	No	Need	No
Power trust [30]	Normalized satisfaction	Summation	Trust value based credibility	No	Need	No
Xiong [26]	Normalized satisfaction	Summation	Personalized similarity	No	Need	No
Yu [28]	Dempster Shafer	Trust net	Weighted majority	No	No	Need
Whitby [25]	Bayesian theorey	Beta aggregation	Quantile detection	No	No	Need
Sharma [18]	Heuristic based rating	Summation	Heuristic based weightage	No	No	Need

the Eigentrust used the trust value to weight a peer's feedback, and then they got the global reputation summation in a matrix notation. In the Bayesian reputation system, a feedback comprises the number of positive outcomes r and the number of negative outcomes s. The feedback is aggregated by adding r and s to the totalized positive and negative outcomes α and β respectively. Hence, we can say that the essence of beta aggregation is also a summation method. Although the summation method is easy to aggregate feedbacks, it lacks the support for robustness to resist malicious feedbacks. However, our proposed Kalman feedback aggregation method can adjust the influence of a malicious feedback through the parameter of estimated feedback variance, which supplies a support to resist malicious feedbacks.

In the aggregation of feedbacks, one fundamental problem is how to cope with the shilling attack [20] where malicious nodes submit dishonest feedback to boost their own ratings or bad-mouth legal nodes. Attackers in a reputation system can either work alone or launch attacks by colluding with one another. A collusive attack can be implemented by disparate attackers or a single attacker acquiring multiple identities through a Sybil attack [27]. Typically, the effect of a single attacker is relatively small, but collusive attackers usually have much more severe influence on the reputation system.

Most existing work considered the credibility of a feedback to detect malicious feedbacks, and they are compared in the literature [13]. A simple solution for measuring the credibility of a node's feedback is to use the node's reputation value, which is used in EigenTrust [11] and PowerTrust [30]. However it is possible that a node may maintain a good reputation by providing high quality services, but send malicious feedbacks to its competitors. The credibility can also be measured by using personalized similarity (PSM) [20, 26], where peer w uses a personalized similarity between itself and another peer v to weight the feedbacks from peer v. The disadvantage of PSM is that the peer w needs to have the wide trust knowledge about peer v's rating on some special peers, which is sometimes an unrealistic precondition. For other credibility methods, Yu and Singh [28] proposed the Weighted Majority Algorithm (WMA) and Whitby et al. [25] used the quantile detection method to filter out unfair ratings. Both these two methods need manually tuned intuitive parameters without guarantee of any quantitative confidence. Sharma et al. [18] also proposed to filter out malicious feedbacks, but it needs a feedback set which can not catch the real time reputation changes.In contrast, based on the EM algorithm, our hypothesis test method can filter out a malicious feedback precisely with a specific confidence level. In addition, our WRLM trust model can self-adaptively calibrate the variance weight factor, so that it can mitigate the influence of malicious feedbacks automatically without any prior experience.

3 Comprehensive RLM Trust Model

3.1 RLM Trust Formulation

Since the reputation value is essentially a statistical value derived from the observation samples (reputation feedbacks), we model the reputation in a statistical form. Assuming that the real reputation of a node is R, in our RLM model, this reputation evaluation is denoted as a two dimension tuple $rep = \{\langle R\rangle,\ P\}$, where $\langle R\rangle$ is the predicted reputation value, and P is the reputation prediction variance, which is the square error between the predicted reputation value $\langle R\rangle$ and the real reputation R. The evaluated tuple *rep* can also be sent to other nodes as a reputation feedback. Hence, a feedback also has two attributes: feedback reputation value and feedback variance.

To maintain the reputation for a node, we assume that the evaluator can receive feedbacks about the node continually through the feedback sessions. The feedback received at session k is denoted as $f_k = \{z_k,\ c_k\}$, z_k and c_k represent the feedback reputation value and the feedback variance respectively. After each reception of a feedback f_k, the evaluator tries to predict the node's real time reputation R_k, and evaluate the prediction variance P_k. Ideally, the reputation feedback value should equal to the real reputation. But due to the incomplete knowledge of the recommender and transient fluctuations of the service quality, the feedback reputation value may have a deviation from the real reputation. Because many independent sources contribute to this deviation, it is reasonable to model the deviation as a zero mean Gaussian noise, so we can model the relation between the feedback reputation value and the real reputation value as:

$$z_k = R_k + q_k \text{ and } q_k \sim \text{Normal}(0,\ Q_k) \tag{1}$$

where Q_k is the feedback noise variance. For a normal node, we assume that its reputation follows a stochastic process. In the statistical inference theory, the trust prediction problem belongs to the infinite impulse response filter problem, which is to predict an output of a system based on the inputs and previous outputs. For the infinite impulse response filter, linear autoregressive (AR) model is widely used, which is reported to have a good prediction performance [14]. Hence, we also use the linear autoregressive model to define the reputation space evolution, and the nonlinear evolution can be treated with locally weighted methods in a similar fashion [2]. As the first approximation, the reputation R_k can be modeled as a first order linear AR model:

$$R_k = A_k R_{k-1} + w_k \text{ and } w_k \sim Normal(0,\ W_k) \tag{2}$$

where A_k is the reputation state transfer factor, and W_k is the variance for the state transfer noise. Equations (1) and (2) define a linear space model for the reputation. This linear model forms a hidden Markov problem as illustrated in Fig. 1 (a Markov process with unknown state parameter R_k). The square nodes are targeted attributes

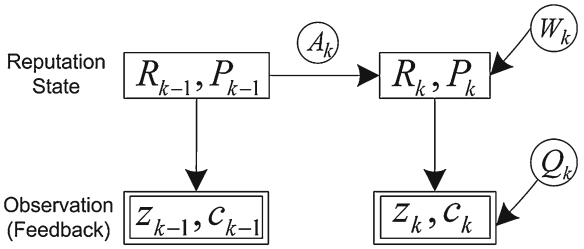


Fig. 1 Graphical RLM model

of the reputation evaluation, double squares are observed reputation feedbacks, and circular nodes are dynamic parameters to be tuned.

Our goal is to obtain the reputation value R_k and the prediction variance P_k from this model, which will be introduced in the next subsection by using our Kalman feedback aggregation method. All the dynamic parameters in the model such as A_k, Q_k and W_k, will be tuned to cope with malicious feedback, which will be introduced in next section.

3.2 Kalman Feedback Aggregation

In RLM model, the reputation state evolution can be tracked in the aggregation of the feedbacks. The Kalman Filter (KF) is an optimal linear estimator for linear Gaussian systems, and it can give the least mean squared prediction of the system state [15]. Because of the linear properties of our RLM trust model, we change the typical Kalman filter to aggregate RLM reputation feedback. Our Kalman feedback aggregation can simultaneously track the evolution of the reputation value and its prediction variance. Moreover, it can adjust the influence of a feedback by the feedback noise variance, which can support further robustness techniques to counter the malicious feedback.

To run the Kalman feedback aggregation, all the dynamic parameters (A_k, Q_k and W_k) in the model are assumed to be known. They will be tuned by our robust model calibration method in the next section. A typical Kalman Filter comprises two steps: the propagation step and the update step. In RLM model, let R'_k denote the posteriori prediction of R_k, P'_k the posteriori estimation of P_k, and the symbol $\langle\rangle$ denote the prediction operator. Then, the corresponding equations for our Kalman feedback aggregation can be defined in Eqs. (3–7), for $k = 1, \cdots N$.

Propagation Step

$$R'_k = A_k \langle R_{k-1} \rangle \tag{3}$$

$$P'_k = A_k^2 P_{k-1} + W_k \tag{4}$$

In the propagation step, the posteriori prediction of R_k and P_k are computed according to the RLM model. To run the Kalman feedback aggregation, we initialize the reputation value $\langle R_0 \rangle$ as 0.5, meaning we know nothing about the initial trust, and the prediction variance $P_0 = 0.01$, meaning that we are not quite sure about the initial reputation prediction [14].

Update Step

$$S_k = P'_k + Q_k \tag{5}$$

$$\langle R_k \rangle = R'_k + \frac{P'_k}{S_k}(z_k - R'_k) \tag{6}$$

$$P_k = \frac{Q_k}{S_k} P'_k \tag{7}$$

In the update step, the feedbacks are aggregated to minimize the mean squared error of the reputation evaluation. Equation (5) computes the variance S_k of the residual prediction error. The final prediction of the reputation value R_k is updated by considering the deviation $(z_k - R'_k)$ and the ratio P'_k/S_k in Eq. (6). From Eq. (5), we can find that if a feedback has a very big feedback noise variance Q_k, then the ratio P'_k/S_k will be very small, leading to a slight update of the predicted reputation value $\langle R_k \rangle$ in Eq. (6). In short, we can get the following conclusion: if a reputation feedback has a bigger feedback noise variance than another feedback, it will have a smaller influence to the reputation value evaluation $\langle R \rangle$ in RLM trust model. This supplies a support to defend malicious feedbacks.

In addition, assuming that there are two feedbacks f_1 and f_2 under a certain reputation state, they have the same reputation transfer parameters: the reputation state transfer factor A and transfer noise variance W, feedback f_1 has a bigger feedback noise variance Q_1 than f_2, thus $Q_1 > Q_2$. From Eq. (5), we can find that $Q_1/S_1 > Q_2/S_2$, and this will lead to a bigger reputation prediction variance P_1 for feedback f_1 in Eq. (7). That is to say for a certain reputation state, after respectively aggregating two reputation feedbacks, which have the same reputation transfer parameters in RLM model, the feedback with a bigger feedback noise variance will result in a bigger reputation prediction variance P in the new reputation evaluation. This important characteristics can support further robust trust evaluation introduced later.

4 Robust RLM Model Calibration

Before running the Kalman feedback aggregation, the parameters A_k, Q_k and W_k in RLM model need to be computed. More importantly, the RLM model needs to be robust to the malicious feedback. In this section, we first introduce the Expectation

Maximization (EM) algorithm to give an autonomous parameter calibration for the model. The EM algorithm can mitigate the influence of a malicious feedback that has incorrect feedback reputation value. Then, we future enhance the model with the hypothesis test method to resist the malicious feedbacks that have both incorrect feedback reputation value and incorrect feedback variance.

4.1 Parameter Calibration

To give a autonomous calibration for our RLM model, we will introduce the expectation maximization (EM) algorithm, which can give a maximum likelihood parameter estimation [5]. Moreover, our EM calibration algorithm can play as a preliminary measure to mitigate the influence of a malicious feedback.

For the parameters in RLM model, our goal is to choose values such that the likelihood of the estimated reputation $\log p\,(R_{1:N})$ is maximized. But due to the analytical issues, we can only have access to a lower bound of the measure [22], which can be formulated as:

$$\log p\,(R_{1:N},\ \mathrm{z}_{1:\mathrm{N}}) = \sum\nolimits_{\mathrm{i}=1}^{\mathrm{N}} \log p\,(\mathrm{z}_{\mathrm{i}}\,|R_{\mathrm{i}}) + \sum\nolimits_{\mathrm{i}=1}^{\mathrm{N}} \log p\,(R_{\mathrm{i}}\,|R_{\mathrm{i}-1}) + \log p\,(R_0) \tag{8}$$

We need to find the parameters that will maximize the above log-likelihood. However, as the sequence of reputation state R_k has not been observed, this maximization is not tractable directly, so we have to apply the EM algorithm. The EM algorithm transforms the maximization of the above likelihood function to iterations of successive two steps (expectation and maximization), where the reputation state sequence is assumed to be known. In the expectation step, EM computes an expectation of the log likelihood with respect to the current estimate of the reputation value. In the maximization step, EM computes the parameters which can maximize the expected log likelihood.

In our RLM model, one important characteristic is that the reputation feedback contains the attribute: feedback variance, which implies how to aggregate the feedback so that a more accurate reputation prediction can be derived. To takes into account the feedback variance c_k, we extend the typical EM algorithm with an initialization step. Thus, after each new feedback $f_k = \{z_k, c_k\}$ becomes available, the EM algorithm will run an iteration that consists of three steps. The final EM equations are:

Initialization Step

$$Q_k = c_k,\ A_k = 1,\ W_k = \beta$$

Expectation Step

$$\sum\nolimits_k = {W_k}^{-1} + {Q_k}^{-1} \tag{9}$$

$$\langle R_k \rangle = (W_k{}^{-1} A_k \langle R_{k-1} \rangle + Q_k{}^{-1} z_k) \Big/ \sum_k \tag{10}$$

Maximization Step

$$A_k = \left(\sum_{i=1}^{k} \langle R_i \rangle \langle R_{i-1} \rangle \right) / \left(\sum_{i=1}^{k} \langle R_{i-1} \rangle^2 \right) \tag{11}$$

$$Q_k = \frac{1}{k} \sum_{i=1}^{k} (z_i - \langle R_i \rangle)^2 \tag{12}$$

$$W_k = \frac{1}{k} \sum_{i=1}^{k} (\langle R_i \rangle - A_i \langle R_{i-1} \rangle)^2 \tag{13}$$

In the initialization step, the variance of the feedback noise is set to be c_k, and the reputation state transfer factor is assumed to be 1, meaning that the reputation state does not change. For the variance W_k of the reputation transfer noise, its initial value β should be set according to the user's estimation of how noisy the system is (e.g., 0.01 for noisy environment and 10^{-4} for less noisy environment [14]). In the expectation step, to compute an expectation of the log likelihood, EM computes the expected reputation value $\langle R_k \rangle$ with respect to its conditional distribution. In the maximization step, the dynamic parameters are chosen so as to maximize the likelihood expectation.

In EM algorithm, if a malicious feedback f_k only changes its feedback reputation value z_k, then the feedback reputation value z_k will have a bigger deviation from the expected reputation value $\langle R_k \rangle$ than a normal feedback. The bigger deviation $(z_k - \langle R_k \rangle)$ of a malicious feedback leads to a bigger feedback noise variance Q_k estimated in Eq. (12). Theorem 1 shows that if a feedback has a bigger feedback noise variance, it will have a smaller influence to the reputation evaluation. Thus, a malicious feedback which only changes its feedback reputation value usually has a smaller influence to the reputation value evaluation than a normal feedback.

Although the EM algorithm can resist part of the malicious feedbacks by creating biger feedback noise variance, a malicious node can still manipulate the model. Assuming that there is a malicious feedback f_k, it has an extremely low feedback variance c_k approaching 0. In the initialization step of EM algorithm, the feedback noise variance Q_k of the RLM model is initialized with c_k. This extremely low c_k makes the feedback reputation value z_k account for a large portion of the expected reputation value $\langle R_k \rangle$ in Eq. (10). Because of the high dependency between z_k and $\langle R_k \rangle$, no matter how much does the feedback reputation value z_k deviate from the real reputation value R_k, the deviation $(z_k - \langle R_k \rangle)$ will be very small, leading to a small final estimation of the feedback noise variance Q_k in Eq. (12). Thus, we can find that in RLM trust model, if a malicious feedback sets its feedback variance to be an extremely low value, then no matter how much does its feedback reputation value deviate from the real reputation value, it can still have a high influence to the new reputation evaluation through the EM method.

4.2 Malicious Feedback Detection

In last subsection, we introduced the EM algorithm to give an robust and autonomous parameter calibration. Although the EM algorithm can resist part of the malicious feedbacks by considering their feedback noise variance, a malicious node can still manipulate the model by setting the feedback variance to be an extremely low value. In this case, the malicious feedback will have a big influence and cause great performance decline to our reputation evaluation.

To make our RLM model robust under such attack, we further introduce the hypothesis test technology to detect the malicious feedbacks. Let H_0 be the hypothesis that the reputation feedback is honest. Recall from Sect. 4 that the Kalman aggregation provides the predicted reputation value $\langle R_k \rangle$ after receiving a feedback $f_k = \{z_k,\ c_k\}$. In a system without malicious feedbacks, the deviation between $\langle R_k \rangle$ and z_k should follows a zero-mean normal distribution with variance $P_k + Q_k$, where P_k is yielded by the Kalman aggregation and Q_k is yielded by our EM algorithm.

To detect the malicious feedback, the hypothesis testing simply evaluate whether the deviation between the feedback reputation value and the predicted reputation is normal enough. Given a significance level α, which determines the confidence level of the test, the problem is to find the threshold value t_k such that:

$$P\left(|z_k - \langle R_k \rangle| \geq t_k \,|H_0\right) = \alpha \tag{14}$$

Under the hypothesis H_0, $(z_k - \langle R_k \rangle)$ follows a zero-mean normal distribution with variance $P_k + Q_k$, so we can also have that:

$$P\left(|z_k - \langle R_k \rangle| \geq t_k \,|H_0\right) = 2 \times \theta\left(t_k \Big/ \sqrt{P_k + Q_k}\right) \tag{15}$$

where $\theta(x) = 1 - \Phi(x)$, with $\Phi(x)$ being the cumulative distribution function (CDF) of a zero-mean unit variance normal distribution. Solving Eqs. (14 and 15), we can get:

$$t_k = \sqrt{P_k + Q_k}\theta^{-1}\left(\alpha/2\right) \tag{16}$$

If the deviation between the feedback reputation value and the predicted reputation value exceeds the threshold t_k, then the hypothesis is rejected. Therefore, the feedback is flagged as malicious, and the update of the reputation and the prediction variance is aborted.

In EM calibration algorithm, a malicious feedback can attack the RLM model by setting its feedback variance to be an extremely low value. However, our hypothesis test technology can enhance the model to resist such attacks. If a malicious feedback sets its feedback variance with a lower value than what it should be, it will be more difficult for the malicious feedback to pass the hypothesis feedback test. Assuming that there is a malicious feedback f_k, it gives a lower feedback variance c_k than the original one. From the proof of Theorem 4, we can find that the lower c_k will result in a smaller feedback noise variance Q_k in Eq. (12), which will further lead to a smaller

reputation prediction variance P_k evaluated in Eq. (7) based on Theorem 2. In brief, the lower feedback variance c_k will create a smaller $P_k + Q_k$, leading to a smaller test threshold value t_k in Eq. (16). The smaller test threshold value will make it more difficult for the malicious feedback to pass the test.

Algorithm 1 Reputation Evaluation Algorithm for RLM

1: INPUTS : $f_k = \{z_k, c_k\}$, $\langle R_{k-1} \rangle$, P_{k-1}, W_{k-1}
2: OUTPUTS: $\langle R_k \rangle$, P_k, W_k, $isMalicious$
3: accept the suggested feedback variance as the local estimated feedback variance $Q_k = c_k$
4: assume the reputation state does not change $A_k = 1$
5: set the state transfer variance according to the experience $W_k = W_{k-1}$
6: run an EM algorithm iteration to estimate Q_k, A_k, W_k using Eqs. (9–13)
7: use the Kalman aggregation to compute $\langle R_k \rangle$, P_k using Eqs. (3–7)
8: compute the malicious feedback threshold t_k using Eq. (16)
9: **if** $z_k - \langle R_k \rangle > t_k$ **then**
10: $isMalicious = true$
11: $\langle R_k \rangle = \langle R_{k-1} \rangle$, $P_k = P_{k-1}$, $W_k = W_{k-1}$
12: **else**
13: $isMalicious = false$
14: run another EM iteration to update Q_k, A_k, W_k using Eqs. (9–13)
15: use the Kalman Aggregation to get the final prediction $\langle R_k \rangle$, P_k
16: **end if**
17: return $\langle R_k \rangle$, P_k, W_k, $isMalicious$

Finally as shown in Algorithm 1, every node in a network needs to run the reputation evaluation locally upon receiving an indirect feedback in the RLM model. After receiving a feedback, the algorithm outputs the result for the reputation evaluation and malicious feedback detection. Firstly, it initializes the dynamic parameters in lines 3–5, and uses the EM algorithm to get a preliminary parameter estimation in line 6. To detect malicious feedbacks, the algorithm uses the estimated parameters to evaluate the new reputation value and its prediction variance (line 7), and then calculates the malicious feedback threshold according to the hypothesis test (line 8). If the deviation between z_k and $\langle R_k \rangle$ is beyond the threshold (line 9), the feedback is labeled as malicious(line 10), and the update caused by the feedback is abandoned (line 11). Otherwise, the algorithm runs another EM iteration to get a more accurate parameter estimation, and uses the Kalman aggregation method to give the final reputation evaluation $\langle R_k \rangle$ and P_k.

5 Robust Weighted RLM Model

We have proposed the standard RLM trust model with malicious feedback filtering in the last two sections. The RLM model with the hypothesis test method can filter out malicious feedbacks half automatically with a statistical significance level.To

defend against malicious feedbacks automatically without any prior experience, in this section, we further propose a self-adaptive WRLM trust model, which extends the original RLM model with the weighted prediction variance.

5.1 WRLM Trust Model

The basic supposition of the model is the same as RLM. The only one difference is that we introduce the weight factor in the z_k calculation. We make the variance of z_k is weighted with ω_k, then ω_k varies with the malicious feedbacks. So we treat the weight ω_k probabilistically as the Gamma prior distribution. The distribution also ensures the weight remain positive. Finally, we can model the relationship between the feedback reputation value and the real reputation value as:

$$\begin{aligned} z_k = R_k + q_k \text{ and } q_k &\sim \text{Normal}(0,\ Q_k/\omega_k) \\ \omega_k &\sim \text{Gamma}(a_{\omega_k},\ b_{\omega_k}) \end{aligned} \tag{17}$$

The calculation of R_k is the same as the Eq. (2). This is also a hidden Markov problem, so it is almost the same as Fig. 1. There are several different parameters. It is shown in Fig. 2.

The method we adopt above is the standard Kalman Filter. The disadvantage of that is all feedbacks' value should be right value. So we need extra hypothesis test to distinguish whether the feedback is malicious. In WRLM model, we have considered this factor when modeling for this problem. The weight ω_k is introduced to balance the malicious feedback's effect. This makes the model a little different

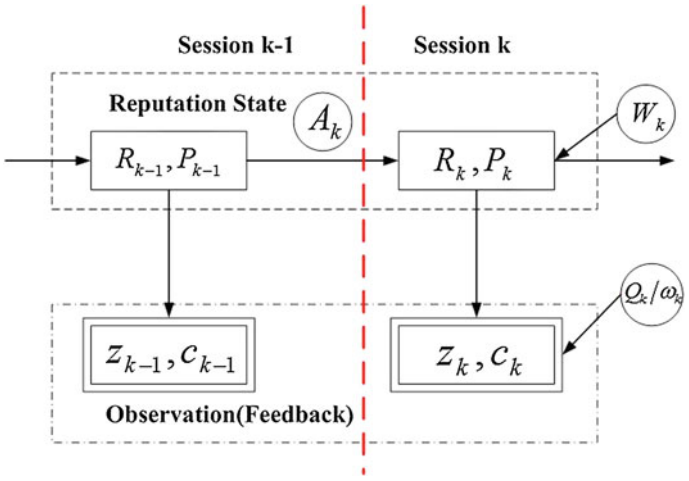


Fig. 2 Graphical WRLM model

from the standard Kalman Filter model. But this method still comprises two steps: the propagation and the update step. The calculating steps are below.

Propagation Step

$$R'_k = A_k \langle R_{k-1} \rangle \tag{18}$$

$$P'_k = W_k \tag{19}$$

Update Step

$$S'_k = \left(P'_k + \frac{Q_k}{\omega_k} \right)^{-1} \tag{20}$$

$$K'_k = P'_k S'_k \tag{21}$$

$$\langle R_k \rangle = R'_k + K'_k (z_k - R'_k) \tag{22}$$

$$P_k = (1 - K'_k) P'_k \tag{23}$$

The observable difference from the standard Kalman Filter algorithm above is that P'_k is not related to P_{k-1} in the propagation step. The original equation should be $P'_k = A_k^2 P_{k-1} + W_k$. In next section, we will illustrate the validity with the EM algorithm. In the equations, $\langle R_{k-1} \rangle$ is the predicted reputation value at session $k-1$. The R'_k and the P'_k are intermediate values which will be used to further calculation in the update step. They are the estimated value according to the Eqs. (18, 19) of the proposed model. The z_k brought by the feedback f_k is the observed reputation value of other node. K'_k, called the optimal Kalman gain, is just to measure the real reputation value how closer to the estimated value R'_k or the feedback value z_k. So the Eq. (23) is used to predict the real reputation value $\langle R_k \rangle$. P_k is the updated covariance of the prediction above. So this is a dynamic and real-time process.

5.2 Calculating with the Feedback

It's also a problem how to estimate the parameters. As mentioned above, we adopt EM algorithm. But because of the introduced weight factor ω_k, it is a little different from the equations above. The maximum likelihood function is $\log p\,(R_{1:N}, z_{1:N}, \omega_{1:N})$ instead of $\log p\,(R_{1:N})$. So the maximum log-likelihood function can be formulated as:

$$\begin{aligned} \log p\,(R_{1:N}, z_{1:N}, \omega_{1:N}) = & \sum_{i=1}^{N} \log p\,(z_i\,|R_i,\,\omega_i\,) + \sum_{i=1}^{N} \log p\,(R_i|R_{i-1}) \\ & + \log p\,(R_0) + \sum_{i=1}^{N} \log p\,(\omega_i) \end{aligned} \tag{24}$$

We also apply the EM algorithm to calculate the parameters making the function maximum. The detail equations are below.

Expectation step

$$\langle \omega_k \rangle = \frac{a_{\omega_k,0} + 0.5}{b_{\omega_k,0} + (z_k - R_k)^2 / Q_k} \tag{25}$$

$$\langle R_k \rangle = P_k \left(\frac{A_k \langle R_{k-1} \rangle}{W_k} + \frac{z_k \langle \omega_k \rangle}{Q_k} \right) \tag{26}$$

$$P_k = \left(\frac{\langle \omega_k \rangle}{R_k} + \frac{1}{W_k} \right)^{-1} \tag{27}$$

Maximization step

$$A_k = \frac{\sum_{i=1}^{k} \langle R_i \rangle \langle R_{i-1} \rangle}{\sum_{i=1}^{k} \langle R_{i-1} \rangle^2} \tag{28}$$

$$Q_k = \frac{1}{k} \sum_{i=1}^{k} \langle \omega_i \rangle (z_i - \langle R_i \rangle)^2 \tag{29}$$

$$W_k = \frac{1}{k} \sum_{i=1}^{k} (\langle R_i \rangle - A_k \langle R_{i-1} \rangle)^2 \tag{30}$$

If we substitute the propagation Eqs. (18) and (19), into the update equations, we reach for recursive expressions for $\langle R_k \rangle$ and P_k. Then we will find the expressions are the same with the Eqs. (26) and (27). So it prove the validity of the WRLM model, on the other side.

In WRLM trust model, when the aggregator receives a malicious feedback with higher or lower reputation feedback value, the weight factor can automatically mitigates the influence of malicious feedbacks. Suppose the value z_k brought by a malicious feedback is too high for some reason. Eq. (25) reveals that if the z_k is so large that it dominates the denominator, then the weight $\langle \omega_k \rangle$ of that feedback will be very small. As the denominator goes to ∞, $\langle \omega_k \rangle$ approaches 0. If $\langle \omega_k \rangle$ is very small, then S'_k, the posterior covariance of the residual prediction error, will be very small, leading to a very small Kalman gain K'_k according to Eqs. (20) and (21). In short, the influence of the data sample z_k will be down-weighted when predicting R_k, the hidden real reputation value at time session k. So the malicious feedbacks don't make the prediction far from the real reputation value.

Of course, we should pay more attention on the model's initialization. The initial value not only makes the iteration convergent, but also should have practical sig-

nificance. If we define the maximum reputation value is 1, the initial value of $\langle R_0 \rangle$ should be 0.5, meaning we know nothing about the initial trust. We are not quite sure about the initial reputation prediction, so prediction variance P_0 can be initialized as 0.01 (a big variance value) [14]. At the beginning of the model, it is assumed that most feedbacks are normal, so the weight factor is not needed to balance the influence. That is to say, the $\langle \omega_k \rangle$ should be 1. $\langle \omega_k \rangle$ has a prior mean of $a_{\omega_k,0}/b_{\omega_k,0} = 1$ and a variance of $a_{\omega_k,0}/b^2_{\omega_k,0} = 1$, so we set $a_{\omega_k,0} = 1$ and $b_{\omega_k,0} = 1$ for instance to make the initialized value as 1. A_k is the coefficient of the estimated reputation changing. Because the changing is gradient, we initialize A_k as 1. Finally, the initial values of Q and W should be set based on the users initial estimation of how many malicious ones of the feedbacks (e.g. $Q = W = 0.01$ for many malicious feedbacks, $Q = W = 10^{-4}$ for less malicious feedbacks [14]).

6 Experiments and Results

In this section, we evaluate our RLM trust model in a simulated reputation environment. We do three sets of experiments to assess the validation, accuracy and robustness of our RLM trust model respectively. In our simulation, the reputation about a node is conducted over N =1,000 feedback sessions, which constitute a feedback dataset. Over the feedback sessions, the real reputation value R_i of a node changes randomly with a factor f (next reputation value / current reputation value). We assume a wide range [0.6, 1.4] for the factor f so that the RLM model can be tested in a difficult situation. Moreover, the minimum and maximum values of a node's real reputations are set to be 0.1 and 1 respectively.

At each feedback session, as the node's real reputation R_i changes, a new reputation feedback f_i is created. There are two kinds of reputation feedbacks: normal feedback and malicious feedback. Normal reputation feedbacks are created to reflect the opinion of a normal recommender. In real scenarios, because of the incomplete local knowledge, a recommender usually cannot give an exactly accurate feedback. As illustrated in Sect. 3, we simulate the normal feedback reputation value z_i as the real reputation value R_i added by a deviation that follows a zero-mean Gaussian distribution. The variance of the distribution is set to be $k\sigma$, where k is a scaling factor (e.g., $k = 1, 2, 3$), and σ is the deviation unit. Since the feedback is a subjective inaccurate rating, we set $\sigma = 0.01$, which means a relatively big deviation noise [14]. Hence, when $k = 1$ (resp. $k = 3$), each feedback reputation value will have a different deviation that follows a zero mean normal distribution with variance 0.01 (resp. 0.03).

From all the created normal feedbacks, some are selected to be simulated as malicious feedbacks. In the simulation, the malicious feedback probability p_m is a variable (e.g., 10, 20, and 30 %), so that we can evaluate its influence on the trust prediction. In one feedback dataset, all the malicious feedbacks are assumed to be collusive, which means that they are of the same kind.

In our RLM reputation model, besides the feedback reputation value, a feedback also comprises the suggested feedback variance c_i. For an honest recommender, c_i should equal to its estimated prediction variance P_i for the feedback reputation value. An attacker can set c_i to be a value bigger or smaller than P_i. If an attacker sets c_i to be a bigger value (intuitively the attacker suggests that he has less confidence in the feedback, and the feedback reputation value may have a bigger deviation), then the aggregator will assign a bigger estimated feedback variance for the feedback (demonstrated in Theorem 4). Therefore, the feedback will have a smaller influence on the reputation aggregation based on Theorem 1. Meanwhile, it will result in a bigger estimated prediction variance, meaning that the aggregator will be less confident about the reputation aggregation. This is contrary to the intent of a malicious attacker. Hence we assume that an attacker always tries to set the suggested feedback variance as lower as possible than P_i. In this scenario, the malicious feedback can have a bigger influence on the reputation aggregation, and mislead the aggregator to believe in the aggregation with more confidence. Hence in the experiment, the suggested feedback variance of a malicious feedback is set to be a low value 10^{-4}.

6.1 Performance Metrics

To evaluate the accuracy of reputation predictions, we calculate the prediction variance and normalized mean squared error (NMSE) of the predictions given by different trust models. Given N trust predictions, the prediction variance is the their mean square error, which can be defined as:

$$\text{PredictionVariance} = \sum_{i=1}^{N} (\langle R_i \rangle - R_i)^2 / N \tag{31}$$

The NMSE is the mean square error of all the reputation predictions normalized by the variance of the real reputation. It can be calculate as $(\sum_{i=1}^{N} (\langle R_i \rangle - R_i)^2 / N) / (\sum_{i=1}^{N} (R_i - (R_i))^2 / N)$, hence we can get:

$$\text{NMSE} = \left(\sum_{i=1}^{N} (\langle R_i \rangle - R_i)^2 \right) / \left(\sum_{i=1}^{N} (R_i - (R_i))^2 \right) \tag{32}$$

For the comparison of robustness, we use the classical false/true positives/negatives indicators. Specifically, a positive is a malicious reputation feedback which should be rejected by the trust model, and a negative is a normal reputation feedback which should be accepted. The number of positives (resp. negatives) in all the feedbacks is n_p(resp. n_n). A false positive is a normal feedback that has been wrongly labeled as malicious, and a true positive is a malicious feedback that has been correctly detected. The number of false positives (resp. true positives) reported by the trust model is n_{fp}(resp. n_{tp}). The false positive rate (FPR) is the proportion of all the normal feedbacks that have been wrongly detected, thus $FPR = n_{fp} / n_n$.

Similarly, the true positive rate (TPR) is the proportion of malicious feedbacks that have been correctly detected, which is $TPR = n_{tp}/n_p$. To detect the malicious feedback, RLM model use the significance level α to decide the confidence (strictness) of the detection. Normally, a higher significance level will increase both the true and false positive rates. According to many experiments in other testing [14, 15], a significance level of 5 % offers a good compromise between the true and false positive rates. Hence, we also set α as 5 % in our experiments.

6.2 Validation of RLM Model

To validate the RLM trust model, we run the model in a clean trust environment with no malicious feedbacks. The RLM model predicts the reputation value of a node, and evaluates the variance of the reputation prediction after each session. Hence, we need to evaluate the fitness of RLM model to represent the reputation value and the reputation prediction variance. First, we set the variance of the feedback deviation to be 1σ, and the malicious feedback probability $p_m = 0$. Figure 3 shows a typical result given by RLM trust model over sessions. The red line denotes the real reputation value of a node at each session, the stars represent the noised reputation feedbacks, and the blue line denotes the reputation value predicted by RLM model. To have a full test about the model performance, the real reputation value evolves randomly with a big change factor over the sessions. A smooth reputation change will be much easier for the trust models, hence, it is not tested in our experiment. We can find that although the feedbacks are not exactly accurate, the RLM model can still give a good reputation prediction, which is so close to the real reputation that their two curves are indistinguishable at most of the sessions.

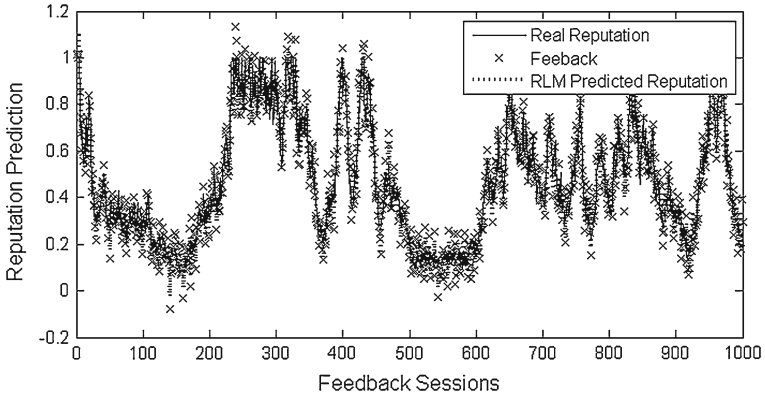


Fig. 3 Sketch map for the real reputation of a node, the reputation feedback and the reputation predicted by RLM model over sessions

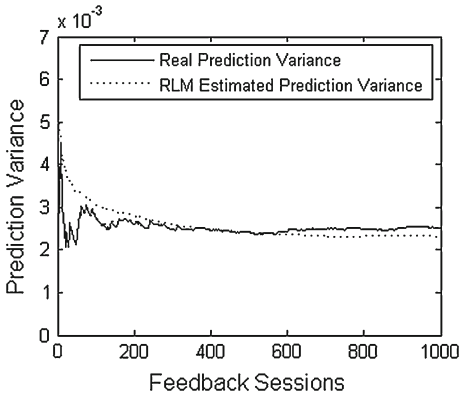


Fig. 4 Real and estimated prediction variance

The RLM model also gives an estimation of the reputation prediction variance P, which can be called the RLM estimated prediction variance. To test the fitness of the estimated prediction variance, we compute the real prediction variance between the predicted reputation value and the real reputation value. Figure 4 shows that the RLM trust model has a high efficiency to estimate the prediction variance. The curves of the RLM estimated prediction variance and the real prediction variance are close except at the initial 200 sessions. This is because the RLM model is initialized with some constant parameters, so it needs some time to stabilize.

For the accuracy test, we compare our RLM model with two other typical general trust models: summation model [10] and Bayesian model [29]. The summation model is widely used in commercial services like eBay, and it can be used in a specific environment like the Engentrust in P2P networks. Since our RLM trust model is a general model without considering the underlying architecture, we implement a pure summation model for comparison. Based on the Beta distribution, the Bayesian model computes the reputation by two parameters: α and β, indicating the number of positive and negative results.

We do two experiments for the accuracy test in a clean trust environment. In the first experiment, the variance of the feedback deviation is 1σ, and the three trust models (Summation, Bayesian and RLM) are tested with the same feedback input. Figure 5 plots the cumulative distribution function of the prediction errors given by these three models. We can see that the majority errors given by RLM model are less than 0.1, while the errors given by the summation and Bayesian models spread to 0.2. Hence, we can get the conclusion that the RLM model has the best prediction accuracy, and the Bayesian model is slightly better than the summation model.

In the second experiment, the variance of the feedback deviation is set to be 1σ, 2σ and 3σ respectively. We compute the prediction variance between the real reputation value and the reputation value predicted by each trust model. Since the Bayesian model is more accurate than the summation model, Fig. 6 only compares the result of Bayesian and RLM trust models. We can see that, under all the cases, the prediction variance given by RLM model is smaller than the Bayesian model. In

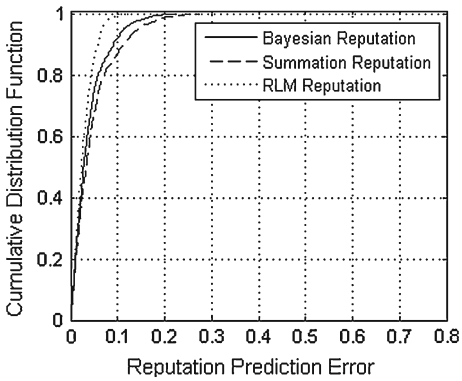


Fig. 5 CDF of reputation prediction errors

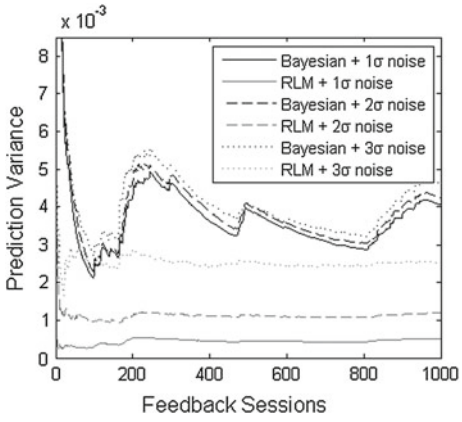


Fig. 6 NMSE of the trust models over sessions

particular, RLM model achieves a considerably higher improvement ratio (of about 50 %) for prediction accuracy when the variance of the feedback deviation is small (1σ), as compared to when the variance of the feedback deviation is big (3σ). This is because the RLM model calibrates the parameters with the maximum likelihood estimation, which is hugely influenced by the feedback deviation. Hence, as the feedback deviation increases, the accuracy benefits of RLM model will be reduced.

6.3 Robustness of RLM Model

In last two subsections, we examined the validation and accuracy of the trust model in a clean trust environment. Next, we evaluate the robustness of RLM trust model under the attack of malicious feedbacks. To resist the malicious feedback, Whitby et al. [25] introduced the quantile filtering method based on the Bayesian reputation system. They filtered out a feedback if it is outside the q quantile and $(1 - q)$ quantile of

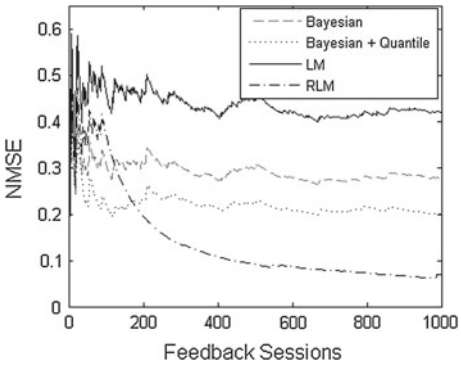


Fig. 7 Average prediction variance with different reputation feedback noises

the Beta distribution for the reputation. The quantile filtering is an intuitive solution without guarantee of any quantitative confidence about the filtering. In contrast, based on RLM model, our hypothesis test method can filter out a feedback with a specific confidence level α through the statistical theory. For the comparison, the Bayesian trust model with quantile malicious feedback filtering is called the Bayesian + Quantile model, and we set the q as 0.01 which is a good choice as reported in [25]. Beside the Bayesian + Quantile model, we also test the robustness of the RLM trust model without the hypothesis test technology, which is called LM trust model.

Firstly, the feedback dataset is created with random positive/negative feedbacks, and the malicious feedback probability p_m is set to 20 %. We run the pure Bayesian model, the Bayesian + Quantile model, the LM model and the RLM trust model on the same feedback dataset. For LM and RLM models, the suggested feedback variance of the malicious feedback is set to be a low value 10^{-4}, so that all malicious feedbacks can have a big threat to the models. Figure 7 plots the normalized mean squared errors given by the four models. Within the initial 100 feedback sessions, the performances of all the four models are not stable. Then, the RLM trust model gradually reaches the smallest NMSE, meaning that RLM model has the best prediction performance under the attack. Figure 7 also shows that the RLM model without the hypothesis test technology is highly vulnerable to the malicious feedback with low suggested feedback variance. Unsurprisingly, the Bayesian model with quantile filtering has a better performance than the pure Bayesian trust model.

Next, we set the malicious feedback probability p_m as 10, 20 and 30 % respectively. With each p_m value, we create five feedback datasets, so that we can get the representative average result for each case. Figure 8 plots the average prediction variance given by the four trust models. It confirms the result that the RLM model has the best prediction performance under the attack. Compared with Bayesian + Quantile model, the prediction variance given by RLM model is much smaller (26 % on average). In addition, we can observe that, when the probability p_m gets close to 30 %, all the four models have a huge performance decline.

Both the Bayesian + Quantile trust model and our RLM model try to detect malicious feedbacks. Therefor based on last experiment, we evaluate the detection effi-

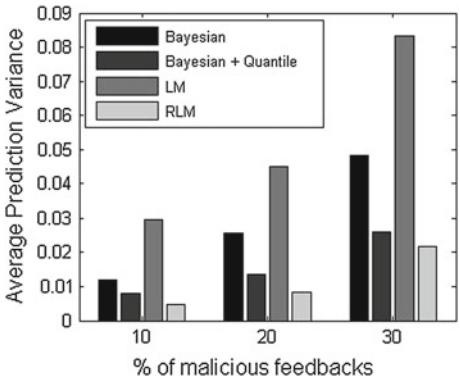


Fig. 8 NMSE of the trust models with malicious feedbacks

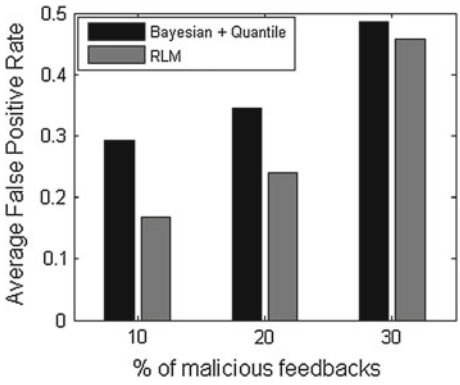


Fig. 9 Average prediction variance with malicious feedbacks

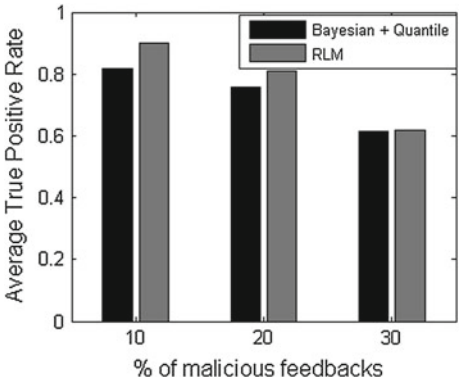


Fig. 10 Average FPR of the different detection methods

ciency of the different models by comparing their false/true positive rate (FPR/TPR). Figures 9 and 10 show that, with all the different malicious feedback probabilities, RLM model has better detection performance than the Bayesian + Quantile model. In particular, when the malicious feedback probability p_m is low (10 %), RLM model

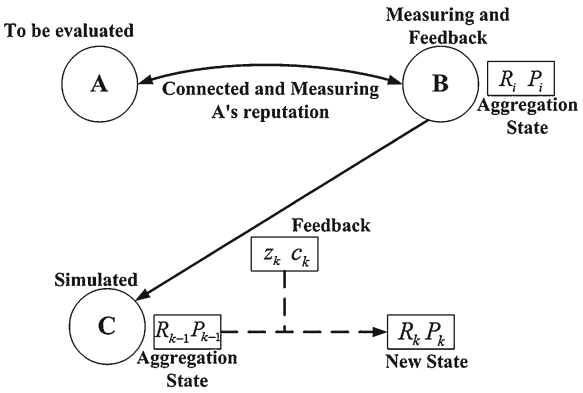


Fig. 11 Experiment principle

has a significantly lower false positive rate (0.12 on average) and a higher true positive rate (0.09 on average) than the Bayesian + Quantile model. When the probability p_m is high (30 %), the performance advantage of RLM model decreases, with a lower false positive rate (0.03 on average) and an almost same true positive rate. This demonstrates that RLM model has higher detection accuracy than Bayesian + Quantile model. However, as the malicious feedbacks probability increases, RLM's accuracy advantage will decrease.

6.4 Analysis of WRLM Model

It's obvious that the weight factor works when the feedback is too high or too low for the real reputation. But this is a just a academic analysis, we use matlab to stimulate and do experiment to prove it. Figure 11 shows the principle of the experiment, when focusing on one node.

In Fig. 11, node A (may be a service provider) is the one whose reputation needs to be evaluated. Node C is the one we simulate its behavior and node B is one of the nodes offering feedbacks. We suppose that the node A applies itself to promoting its quality of service and simply model its real reputation varying with time as $R = \log_{100}(t+40)$. We create a data set to simulate 5,000 feedbacks and inputs along with time. Some too high or too low data values are added to the data set to simulate the malicious feedbacks. The node is initialized as introduced above. Figure 12 shows the result of the experiment. It is mainly about a comparison between the standard Kalman Filter and the weighted Kalman Filter proposed by us to predict reputation. The main method in the RLM model is standard Kalman Filter.

It's obvious that the blue line which represents the standard KF mainly grows along with the curve. The red line which represents the weighted KF is almost the same with the blue line when there are no malicious feedbacks. But there exists a great

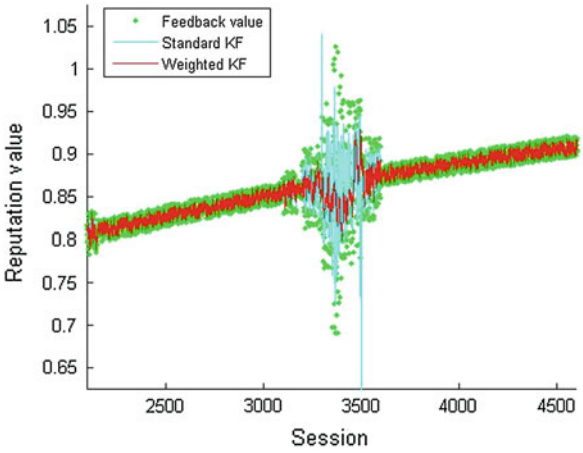


Fig. 12 Comparison between standard and weighted KF

deviation for the standard method when comes to the session with a lot of malicious feedbacks. So the WRLM proposed by us is more self-adaptive and automatic to predict the next reputation value.

7 Discussion and Conclusion

Reputation based trust systems can play a vital role in service selection and promoting service providers to improve their service quality in open cloud systems. A reputation-based trust system usually comprises two components: the underlying architecture, which concerns of how to distribute and collect the feedbacks, and the trust model, which describes the representation and aggregation of reputation-based trusts. This chapter focuses on the design of a comprehensive and robust general trust model. Most early research work about reputation does not consider the accuracy of a reputation evaluation or the credibility of a reputation feedback from others. This makes the reputation system extremely vulnerable to biased reputation calculation and malicious reputation feedback. Some recent work proposes various methods to assessment the reputation feedback credibility. However, they usually need wide trust knowledge of the system or manually tuned parameters. Hence, we need a efficient evaluation about the trust evaluation to get a robust reputation system.

The evaluation about local trust evaluation can be used to denote the reputation accuracy, while the evaluation of the reputation feedback is the feedback credibility. In this chapter, we introduced the Robust Linear Markov (RLM) model and its extension WRLM model for trust representation and aggregation. For a comprehensive reputation evaluation, RLM defined the reputation by two attributes: reputation value and reputation prediction variance. Based on the novel Kalman feedback aggregation,

the RLM model can enable an evaluator to assess the accuracy of a reputation prediction made by itself, which can can help to achieve a better local decision making and a more intelligent third-party reputation aggregation. To defend against malicious feedbacks for RLM model, we introduced the Expectation Maximization algorithm to autonomously tune the model parameters, and then proposed the hypothesis test method to resist malicious feedbacks half automatically with a statistical significance level. To defend against malicious feedbacks automatically without any prior experience, we further proposed the self-adaptive WRLM trust model by extending the original RLM model with the weighted prediction variance. WRLM can self-adaptively and automatically mitigate the influence of malicious feedbacks with different estimated prediction variance. We also demonstrated the robustness of our RLM and WRLM model through theoretical analysis. Simulation results show that the RLM model can efficiently capture the reputation value and its prediction variance. Under the attack of malicious feedbacks, the RLM model has higher malicious detection accuracy (lower false positive rate and higher true positive rate) than the Bayesian + Quantile method. In addition, the WRLM trust model can self-adaptively defend malicious feedbacks with more robustness than the standard RLM model.

The proposed RLM and WRLM models are general trust models about reputation without specification of the detailed application scenarios. Further work is needed to investigate how to apply the RLM and WRLM models in specific applications such as P2P environments, cloud service computing and social networks. In the P2P environments, all the nodes need to cooperate to have a consistent over all reputation evaluation about all the participant nodes. Like EigenTrust [11] and PowerTrust [30], the trust matrix can be used to represent the reputation from the system view. However, the trust matrix needs to be extended to include the reputation prediction variance, and the reputation aggregation mechanism needs to be adjusted to use the Kalman feedback aggregation. In the cloud service computing, the RLM and WRLM models need to consider how to capture the community between users. In the initialization step, feedbacks from different communities should be given different parameters, so that the feedbacks will have different influence to the trust evaluation according their communities. In the social network applications, trusts are usually evaluated through trust networks. We need to investigate how the reputation prediction variance propagates through the trust networks. In addition, the Kalman feedback aggregation methods needs to used in the concatenation and aggregation operators, which are the two popular operators used in the trust networks.

Acknowledgments The authors are partially supported by the NSFC No. 61103194, the Aid program for Science and Technology Innovative Research Team in Higher Educational Institutions of Hunan Province.

References

1. Andersen R, Borgs C, Chayes J, Feige U et al. (2008) Trust-based recommendation systems: an axiomatic approach. In: Proceedings of 17th international conference on world wide web (WWW)
2. Atkeson C, Moore A, Schaal S (1997) Locally weighted learning. AI Rev 11:11–73
3. Chen WY, Zhang D, Chang EY (2009) Combinational collaborative filtering for personalized community recommendation. In: Proceedings of ACM international conference on knowledge discovery and data mining
4. Conner W, Iyengar A, Mikalsen T, Rouvellou I, Nahrstedt K (2009) A trust management framework for service-oriented environments. In: Proceedings of 18th international conference on world wide web (WWW)
5. Dempster A, Laird N, Rubin D (1997) Maximum likelihood from incomplete data via the EM algorithm. J Roy Stat Soc: Ser B 39(1):1–38
6. Golbeck J, Hendler J (2006) Inferring trust relationships in web-based social networks. ACM Trans Internet Technol 6(4):497–529
7. Golbeck J (2008) Weaving a web of trust. Science 321(5896):1640–1641
8. Hoffman K, Zage D, Nita-Rotaru C (2009) A survey of attack and defense techniques for reputation systems. ACM Comput Surv 14(4):1
9. Huang J, Nie F, Huang H, Tu Y (2012) Trust prediction via aggregation heterogeneous social networks. In: Proceedings of the 21st ACM international conference on information and, knowledge management, pp 1774–1778
10. Jøsang A, Ismail R, Boyd C (2007) A survey of trust and reputation systems for online service provision. Decis Support Syst 43(2):618–644
11. Kamvar S, Schlosser M, Garcia-Molina H (2003) The Eigentrust algorithm for reputation management in P2P networks. In: Proceedings of 12th international conference on world wide web (WWW)
12. Kuter U, Golbeck J (2007) SUNNY: a new algorithm for trust inference in social networks using probabilistic confidence models. In: Proceedings of internation conference on artificial intelligence (AAAI)
13. Liang Z, Shi W (2008) Analysis of ratings on trust inference in open environments. Elsevier Perform Eval 65(2):99–128
14. Maybeck PS (1979) Stochastic models, estimation, and control. Mathematics in science and engineering, vol 141. Academic Press, New York
15. Morris JM (1976) The Kalman filter: a robust estimator for some classes of linear quadratic problems. IEEE Trans Inf Theor 22(5):526–534
16. Raya M, Papadimitratos P (2008) Gligor VD, Hubaux JP (2008) On dataCentric trust establishment in ephemeral ad hoc networks. In: Proceedings of INFOCOM
17. Resnick P, Zeckhauser R, Swanson J, Lockwood K (2006) The value of reputation on eBay: a controlled experiment. Exp Econ 9(2):79–101
18. Sharma NK, Gaur V, Muttoo SK (2012) A dynamic reputation system with built-in attack resilience to safeguard buyers in e-market. In: Proceedings of ACM SIGSOFT software engineering, notes, pp 1–19
19. Song S, Hwang K, Kwok YK (2006) Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling. IEEE Trans Comput 55(6):703–719
20. Srivatsa M, Xiong L, Liu L (2005) TrustGuard: countering vulnerabilities in reputation management for decentralized overlay networks. In: Proceedings of 14th international conference on world wide web (WWW)
21. Stern D, Herbrich R, Graepel T (2009) Matchbox: large scale online Bayesian recommendations. In: Proceedings of 18th international conference on world wide web (WWW)
22. Ting J, D'Souza A, Schaal S (2006) Bayesian regression with input noise for high dimensional data. In: Proceedings of ACM 23rd international conference on, Machine learning

23. Wang X, Ou W, Su J (2009) A reputation inference model based on linear hidden markov process. In: Proceedings of ISECS international colloquium on computing, communication, control, and management
24. Wang Y, Singh MP (2006) Trust representation and aggregation in distributed agent systems. In: Proceedings of international conference on artificial intelligence (AAAI), Boston
25. Whitby A, Jøsang A, Indulska J (2004) Filtering out unfair ratings in Bayesian reputation systems. In: Proceedings of international joint conference on autonomous agents and multiagent systems (AAMAS)
26. Xiong L, Liu L (2004) PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. IEEE Trans Knowl Data Eng 16(7):843–857
27. Yu H, Gibbons P, Kaminsky M, Xiao F (2008) A near-optimal social network defense against sybil attacks? In: Proceedings of the 2008 IEEE symposium on security and privacy
28. Yu B, Singh MP (2003) Detecting deception in reputation management. In: Proceedings of international joint conference on autonomous agents and multiagent systems (AAMAS)
29. Zhang Y, Fang Y (2007) A fine-grained reputation system for reliable service selection in peer-to-peer networks? IEEE Trans Parallel Distrib Syst 18(8):1134–1145
30. Zhou R, Hwang K (2006) PowerTrust: a robust and scalable reputation system for trusted peer-to-peer computing. IEEE Trans Parallel Distrib Syst 18(5):460–473

Part III
Case Studies: Cloud Security, Privacy and Trust

Cryptographic Role-Based Access Control for Secure Cloud Data Storage Systems

Lan Zhou, Vijay Varadharajan and Michael Hitchens

1 Introduction

With the rapid increase in the amount of digital information that needs to be stored, cloud storage has attracted much attention in recent times because of its ability to deliver resources for storage to users on demand in a cost effective manner. The cloud can provide a scalable high-performance storage architecture, and can help to significantly reduce the cost of maintenance of individual services. There are different types of infrastructures associated with a cloud [2]. A public cloud is a cloud which is made available to the general public, and resources are allocated in a pay-as-you-go manner. A private cloud is an internal cloud that is built and operated by a single organisation. Potentially there could be several benefits to storing data in the public cloud.[1] The storage capacity of a cloud is almost unlimited, and users only need to pay for the storage space for their actual needs. Outsourcing data to cloud can also help to save the costs and efforts in storage maintenance tasks, such as data backup and replication, disaster recovery, and hardware maintenance. Furthermore, cloud storage can provide a flexible and convenient way for users to access their data from anywhere on any device.

While the cloud storage has many benefits, it also brings important security issues. Since data in the cloud is stored in one or more data centres which are often distributed geographically in different locations, users do not know where their data is stored and there is a strong perception that users have lost control over their data after it is uploaded to the cloud. In order to allow users to control the access to their data stored in a public cloud, suitable access control policies and mechanisms are required. The access policies must restrict data access to only those intended by the data owners.

[1] In this chapter, when we use the word cloud, we are referring to a public cloud.

V. Varadharajan (✉) · L. Zhou · M. Hitchens
Information and Networked Systems Security Research, Department of Computing, Macquarie University, Sydney, Australia
e-mail: vijay@science.mq.edu.au

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_11,

These policies need to be enforced by the cloud. In many existing cloud storage systems, data owners have to assume that the cloud providers are trusted to prevent unauthorised users from accessing their data.

Access control has been widely used by data storage systems in the evaluation of whether a user has access to a particular resource in the system. In a storage system, the stored data needs to be protected from unauthorised access, and the system is expected to control the access to the data according to specific security context and policies that are defined for the storage system. In access control models, the entities that perform the access are referred as subjects, and the resources to be accessed are called objects. Before enforcing access control, in general it is necessary to determine the identity of the subject requesting access to an object. The process of authentication involves verification of the identity of a subject (e.g. a user) that it is as who it claims to be. Typical authentication methods include passwords, tokens such as smart cards and biometrics such as iris scans and fingerprints. Authorisation or access control[2] refers to a set of security policies which defines the users' permissions to access resources (objects) in the system. In this chapter, we assume that standard authentication mechanisms are available which can be performed by the system; we focus on the *authorisation* service in a cloud scenario.

Depending on the way that the security policies are specified, access control can be categorised into different models. We first describe several well known access control models.

Mandatory Access Control (MAC) Model The concept of mandatory access models have been developed and formalised in [5, 30]. In a system that uses mandatory access control (MAC) model, access policies are specified by a central security administrator in the system. There is no concept of individual ownership in the MAC model; all resources are controlled by the system and subject to the MAC policies, and the central administrator(s) decides who can access the resources in the system. Typically subjects (users) in the system are allocated security labels referred to as security clearances and objects in the system are allocated security labels referred to as security classifications. To access a resource in a MAC-based system, the subject must hold proper security clearance required for that resource with its security classification. The security policy defines rules as on the security labels, that is security clearances and classifications. If these rules are satisfied, then the access is allowed.

Discretionary Access Control Model Discretionary access control is a user-centric access control model. In contrast to MAC model, resources in the system governed by discretionary access control (DAC) model have owners. These owners have the control over the access permissions to the resources and can determine which users are allowed to access their resources. Since the access permission to a resource is solely specified by its owner, defining security access policies in a DAC model can be easy to implement and hence common in practice.

Attribute-based Access Control Model Since the late 1990s, attribute-based access control has emerged with the development of distributed systems. In attribute-

[2] We will use access control and authorisation interchangeably in this chapter and will not enter into a detailed discussion on the differences between these two terms.

based access control (ABAC), access permissions to resources are assigned to a set of attributes instead of individual users. Users who qualify the set of attributes can access the resource. In ABAC models, attributes are associated with characteristics of users. The attributes do not necessarily need to relate to each other, and the access policies are defined using a combination of attributes with certain logical relations. ABAC plays an important role in service-oriented architecture (SOA) and has been used as a standard in web service security specification such as extensible access control markup language (XACML) and security assertion markup language (SAML).

Role-based Access Control Model With role-based access control, access decisions are based on the roles that individual users have been assigned to. Users are granted membership to roles based on their competencies and responsibilities in the organisation. Access rights are grouped by role name, and the use of resources is restricted to individuals authorised to the associated role. The use of roles to control access can be an effective means for developing and enforcing enterprise-specific security policies, and for streamlining the security management process. The RBAC model was formally introduced in 1992 [20]. In this model, a role can inherit permissions from other roles. A user who has been granted membership to a role has access to permissions of this role as well as other roles that this role inherits permissions from. The RBAC model was extended and updated in 1996 [36], and the RBAC standard was proposed in 2000 [37].

2 Cryptographic Access Control Schemes

In traditional access control systems, enforcement of access policies is carried out by trusted parties which are usually the service providers. In a public cloud, as data can be stored in distributed data centres, there may not be a single central authority which controls all the data centres. Furthermore the administrators of the cloud provider themselves would be able to access the data if it is stored in plain format. To protect the privacy of the data, data owners employ cryptographic techniques to encrypt the data in such a way that only users who are allowed to access the data, as specified by the access policies, will be able to do so. We refer to this approach as a policy-based encrypted data access. The authorised users who satisfy the access policies will be able to decrypt the data using their private keys, and no one else will be able to reveal the data content. Therefore, the problem of managing access to data stored in the cloud is transformed into the problem of management of keys which in turn is determined by the access policies.

2.1 Broadcast Encryption

A trivial solution to protect the privacy of data stored in cloud is to use a cryptographic encryption scheme to encrypt the data before storing it in the cloud. This would allow

Table 1 Access control matrix

	f_1	f_2	f_3
u_1	1	1	1
u_2	0	1	0
u_3	1	1	0

only the users who have access to the key(s) to decrypt the data and to view the data in the plain form. The problem of achieving secure access to data stored in the cloud is transformed into the problem of access to keys. This approach is suitable for MAC and DAC models, whose access policies can be represented by an access control matrix (ACM).

Table 1 shows an example of such an access control model. Let us assume that the matrix is for a cloud storage system using a DAC model. The set $\{f_1, f_2, f_3\}$ represents all the objects in the model, that is, say files stored in the cloud. The set $\{u_1, u_2, u_3\}$ represents all the subjects, that is, the users who want to access these files stored in the cloud. Each file in the model has an owner, and the owners have the flexibility to control who can access the files. Each row in the matrix is a capability list (CL) of the subject, and the column corresponding to each object is called an access-control list (ACL) for that object. A snapshot of the access matrix represents a protection state where 1 means "has access" and 0 indicates "no access". It is clear that the owner of each file can simply employ a secret key encryption scheme to encrypt the data and distribute the secret key to the users with whom s/he wishes to share the data, and store his/her resource in the encrypted form to the cloud. For example, the owner of the file f_1 encrypts and uploads the file and gives the secret key to the users u_1, u_3. Then only u_1 and u_3 can decrypt the file f_1 because they possess the secret key corresponding to the encryption, and no one else can reveal the content of the file. We require a secure way of achieving key distribution to these selected users, who have the access to view the data according to the access control policies.

Since the secret key encryption requires different keys to encrypt different objects, the number of keys will become large when there are massive amount resources in the system. Therefore the owners may wish to use public key encryption techniques to protect the privacy of their files as they can simply use the public keys of users to encrypt data and do not need to transfer any key to users. However, if an owner uses a public/private key pair to encrypt/decrypt the file, s/he may need to encrypt the same file multiple times if s/he wants more than one users to access the file because the public keys are different for different users. This will make the approach impractical when there are a large number of users in the system. Fortunately, owners can use broadcast encryption schemes to encrypt files in this scenario.

The concept of Broadcast Encryption (BE) was introduced by Fiat and Naor in [21]. In BE schemes, a broadcaster encrypts messages and transmits them to a group of users who are listening in a broadcast channel. Then they use their private keys to decrypt the transmissions. While encrypting the messages, the broadcaster can choose the set of users that is allowed to decrypt the messages. Following this

original scheme, many other BE schemes have been proposed such as [8, 22, 25]. These schemes require public parameters for every user, and every time a user wants to join or leave the system, the public parameters need to be updated.

In the example shown in Table 1, the owner of f_1 can use a broadcast encryption scheme to encrypt the file to both u_1 and u_3 using their public keys and uploads the encrypted file to the cloud. Then u_1 and u_3 will be able to decrypt the file using their own private keys. This approach needs a public-key infrastructure (PKI) to manage the public keys of all the users in the system. This is needed for the owners to ensure that they are using the correct public keys to encrypt their files for the right users. However, the system can be made even simpler by using ID-based cryptographic techniques.

In 1984, Shamir [39] suggested the possibility of a public key encryption scheme in which the public key can be an arbitrary string. In 2001, Boneh and Franklin introduced an ID-based encryption (IBE) scheme, in which the sender can use the identity of the receiver as the public key to encrypt the messages. An ID-based broadcast encryption scheme (IBBE) is defined in a similar way. In an IBBE scheme, the system does not need to have any preset parameters for every user, and a broadcaster only needs to know the identity of the user if this user is allowed to decrypt the messages. In this case, one user joining or leaving the system will not affect any other user. Moreover, the users do not even need to have the decryption key at the time when the messages were encrypted. They can obtain their keys afterwards. Several IBBE schemes have been proposed subsequently in [9, 16, 27].

Generally, an IBBE scheme involves three different parties: a Private Key Generator (PKG), the users with unique identities, and the broadcasters who possess the messages. The PKG generates decryption keys for each authorised users based on his/her identities. A broadcaster can encrypt messages to a selected group of users and transmit the messages via a broadcast channel. The broadcaster uses only the public key and users' identities to encrypt the messages. More formally, an IBBE scheme is composed of four algorithms which are described as follows:

$IBBE.Setup(\lambda)$: takes as input the security parameter λ and outputs a master secret key mk and a group public key pk. mk is given to PKG, and pk is made public.

$IBBE.Extract(\mathsf{mk}, \mathsf{ID})$: an algorithm executed by the PKG, on input of a user identity ID and the master secret mk, returns the user private key $\mathsf{sk}_{\mathsf{ID}}$.

$IBBE.Encrypt(\mathsf{pk}, \mathcal{U}, M)$: an algorithm executed by the broadcaster, on input of the set $\mathcal{U}$ of identities of users to whom it wishes to encrypt the message and the public key pk, outputs a pair $(\mathsf{Hdr}_{\mathcal{U}}, K_{\mathcal{U}})$, where $\mathsf{Hdr}_{\mathcal{U}}$ is called the header and $K_{\mathcal{U}}$ is in the key space of a symmetric encryption scheme $\mathcal{E}_{sym}$.

Assume that a broadcaster wishes to encrypt a message M to a group $\mathcal{U}$ of users with identities $\{\mathsf{ID}_1, \cdots, \mathsf{ID}_n\}$. Let $\mathcal{E}$ be the IBBE scheme and $\mathcal{E}_{sym}^{key}(M)$ be the encryption of M using $\mathcal{E}_{sym}$ under the secret key key. The ciphertext C is denoted as:

$$C = \mathcal{E}_{IBBE}(M, \mathcal{U}) = \{\mathsf{Hdr}_{\mathcal{U}}, \mathcal{E}_{sym}^{K_{\mathcal{U}}}(M)\}$$

$IBBE.Decrypt(\mathsf{pk}, \mathsf{sk}, C)$: an algorithm executed by the user to decrypt the ciphertext on input of the user secret key sk and the public key pk.
This algorithm has two steps: the first step takes as input the user secret key sk and the header Hdr, and recovers the value $K_{\mathcal{U}}$, and then the second step uses the symmetric key $K_{\mathcal{U}}$ to decrypt M from $\mathcal{E}_{sym}^{K_{\mathcal{U}}}(M)$.

Using an IBBE scheme, the owners of the files can encrypt their files using the identities of the users with whom they wish to share the files and upload the encrypted files to cloud. The certificate authorities in the system generate the private keys for users in the system and distribute the private keys to users via a secure channel. The private keys for users can be issued after the files been uploaded to the cloud.

2.2 Attribute-Based Encryption

Integration of cryptographic techniques with ABAC model has led to a technique called attribute-based encryption (ABE). In an ABE system, the access permission to a resource is associated with a set of attributes and a security policy based on these attributes. Only the users who hold the keys for the attributes are able to decrypt the content. This feature allows ABE schemes to be used in protecting the privacy of resources in a cloud storage system which uses ABAC model to control access privileges.

The first ABE scheme was proposed in [24], in which ciphertexts are labeled with sets of attributes and private keys are associated with access structures that control which ciphertexts a user is able to decrypt. Hence this scheme is also referred to as key-policy ABE or KP-ABE. In KP-ABE schemes, the owner of the data does not have the control over who is allowed to access the data. The owner must trust the key-issuer who issues the appropriate keys to grant or deny access to the appropriate users. In [6], another form of ABE scheme was introduced. This scheme works in the reverse manner where the user keys are associated with sets of attributes and the ciphertexts are associated with the policies. Hence it is referred to as the ciphertext-policy ABE or CP-ABE scheme. Some variations of the CP-ABE schemes have been proposed such as the ones in [12, 19, 28] with features like chosen ciphertext attack (CCA) secure solution and constant size solution.

The access policies of ABE is specified in a tree structure. Each leaf node corresponds to an attribute defined in the system. Each non-leaf node represents a threshold gate which connects its children attributes or threshold gates. We denote n_x as the number of child nodes that a threshold gate x has and k_x as its threshold value where $0 < k_x \leq n_x$. It is clear that the threshold gate is an OR gate when $k_x = 1$ and it is an AND gate when $k_x = n_x$. Verifying whether a set of attributes S satisfies the access tree $\mathcal{T}$ is a recursive process. We denote $\mathcal{T}(x) = 1$ if and only if the node x is satisfied. When x is a leaf node, $\mathcal{T}(x)$ returns 1 if the attribute associated to the node is in the set S. When x is a non-leaf node, $\mathcal{T}(x)$ returns 1 if at least k_x child

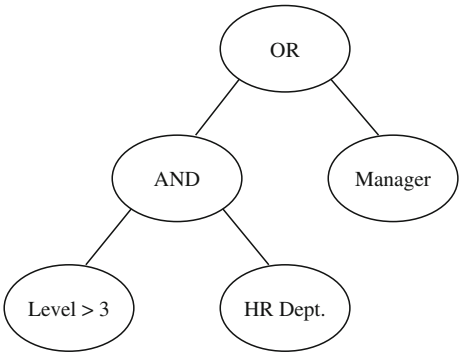


Fig. 1 ABE access structure example

nodes evaluated as 1. Assume the root node of the access structure is r. The access tree is satisfied if and only if $\mathcal{T}(r) = 1$.

Let us look at an example shown in Fig. 1. Assume that a recruitment agency wants to store the profiles of its job candidates in the cloud so the information can be easily shared with their client companies. Now the marketing department of a company is looking for a sales, and they want to access the profiles from the recruitment agency. Then the recruitment agency specifies the access policies for the company as that only the managers of the marketing department or the staffs from the HR department and whose level is greater than 3 can view the profiles. The recruitment agency can use an ABE scheme to encrypt the suitable profiles and stores them in the cloud. Three decryption keys will be generated in this example for the three attributes respectively, and they will be distributed to the employees who qualify the attributes. The managers of the marketing department will be given the key for the attribute "Manager", all the staffs in the HR department will be given the key for the attribute "HR Dept.", and only the staffs whose levels are greater than 3 will be given the key for the attribute "Level >3". Then a staff of the company will be able to decrypt and view the candidates' profiles with the given keys if and only if the attributes associated with his/her keys satisfy the access tree.

When using a KP-ABE scheme to encrypt resources, attributes are assigned to the ciphertext in the encryption, and the policies for the access structure are associated with the decryption keys when the keys are generated. It is the authority who generates the keys that decides the access policies. In the above example, KP-ABE is suitable in the scenario where the profiles of candidates will be shared with another client company which has a different organisation structure. Then ciphertext of the profiles are associated with the same set of attributes, but the decryption keys will be generated separately under the different access policies of the other company.

In a CP-ABE scheme, the access policies are associated with the ciphertext and are specified in the encryption. Keys can be generated prior to the data encryption, and remain unchanged when data is encrypted under different access policies. In the above example, CP-ABE is suitable in the case where the profiles need to controlled under different policies in the same client company. For example, the profiles of the

sales candidates are encrypted so that only the managers of the marketing department can view them, and the profiles of the technician candidates are allowed to be viewed only by the managers of the technical department. Then the employees of the client company only need to hold a single key for an attribute even it is used in several different access structures. The recruitment agency only needs to specify the different access policies while encrypting different profiles.

A typical ABE scheme consists of following four algorithms: *Setup*, *Encrypt*, *KeyGen*, *Decrypt*. However, the input parameters of the algorithms *Encrypt* and *KeyGen* are different in KP-ABE and CP-ABE because the access structures are associated with keys and ciphertexts respectively in these two types of ABE schemes. Let us now describe the algorithms of an ABE scheme as follows:

$ABE.Setup(\lambda)$: takes as input the security parameter λ and outputs a master secret key mk and a public key pk. mk is kept secretly, and pk is made public.

$KP-ABE.Encrypt(\mathsf{pk}, M, S)$: an algorithm on input of the system public key pk, a message M and a set of attributes S, outputs the ciphertext C. In this algorithm, only the attributes are used in computing the ciphertext, and the access structure of the attributes is not specified. Not all the attributes need to appear in the access structure, but S needs to be the super set of all the attributes which will be used in the access structure.

$KP-ABE.KeyGen(\mathsf{pk}, \mathsf{mk}, \mathcal{T})$: an algorithm on input of the system public key pk, master secret key mk and the access structure T, outputs a set of decryption keys dk.

$CP-ABE.Encrypt(\mathsf{pk}, M, \mathcal{T})$: an algorithm on input of the system public key pk, a message M and the access structure T, outputs the ciphertext C.

$CP-ABE.KeyGen(\mathsf{pk}, \mathsf{mk}, S)$: an algorithm on input of the system public key pk, a message M and a set of attributes S, outputs a set of decryption keys dk. Similar to that in $KP-ABE.Encrypt$, the attribute set S is a super set of the attributes in the access structure.

$ABE.Decrypt(\mathsf{pk}, \mathsf{dk}, C)$: an algorithm on input of the system public key pk, the decryption keys dk and the ciphertext C, outputs the message M in the plaintext form.

In an ABE system, a user needs to authenticate to the key generation authority to prove his/her identity to obtain the keys to decrypt the data. However, in some cases, the attributes may not be managed by a single authority. For example, an attribute for holding a certain certificate may need to be authorised by an external authority who can verify the validity of the certificate. Several multi-authority ABE schemes have been introduced in [10, 11, 29], which allows the decryption keys associated with the attributes to be issued by different authorities.

2.3 Role-Based Encryption

We call the cryptographic schemes which integrates encryption schemes with the RBAC model a cryptographic RBAC scheme which can enforce the RBAC policies in an untrusted environment. The cryptographic RBAC scheme allows data to be encrypted to a specific role in the system, and only users who are members of this role or members of roles that inherit from this role would be able to access the data by decrypting it. This approach allows data to be encrypted before storing it in an untrusted cloud environment and the stored ciphertext can only be decrypted by those who are allowed by the access policies.

A hierarchical cryptographic access control scheme [1] was proposed in 1983. Because of the similarity in structures between hierarchical access control and RBAC, a hierarchical cryptographic access control scheme can be easily transformed into a cryptographic RBAC scheme. The problem of access control for securely outsourcing data using cryptographic techniques was first considered in [31]. An improved scheme was proposed in [17] to address access policy updates. Several cryptographic access control approaches have been investigated in [3, 26] to address the problem of secure data access and cost effective key management in distributed environment. Subsequently, a two layer encryption model was proposed in [18] to prevent a service provider from accessing the content of data but the service provider is able to run queries or perform other operations on the data for users who can decrypt the data using their keys.

Hierarchical ID-based Encryption (HIBE) is an alternative approach for the management of keys. In the HIBE schemes of [7, 23], a user with an identity in the hierarchy tree can decrypt messages encrypted to its descendant identities, but cannot decrypt messages for others. HIBE schemes can be easily used to enforce RBAC by associating the leaf nodes in the hierarchy tree with users and non-leaf nodes with roles. However, there are several issues with the HIBE schemes. Firstly, in a HIBE scheme, the length of the identity becomes longer with the growth in the depth of hierarchy. Secondly, the identity of a node must be a subset of its ancestor node so that its ancestor node can derive this node's private key for decryption. Therefore, this node cannot be assigned as a descendant node of another node in the hierarchy tree unless the identity of the other role is also the super set of this node's identity.

Recently we have seen the development of schemes built directly on RBAC policies. A role-based encryption scheme (RBE) is introduced in [40]. This scheme has several superior characteristics such as constant size ciphertext and decryption key, efficient user revocation and user management. However, the user revocation in this scheme requires the update of all the role related parameters. Another scheme was proposed in [43]. In this scheme, the size of the ciphertext increases linearly with the number of all the ancestor roles. In addition, if a user belongs to different roles, multiple keys need to be possessed by this user. Moreover, the management of the user membership for each individual role requires the use of the system secret keys. An improved RBE scheme is introduced in [42] which has a more efficient

user revocation while retaining the superior characteristics from the original scheme [40].

We adopt the RBE definition given in [40], and describe the algorithms of an RBE scheme. An RBE scheme includes the following parties: a group administrator GA who has the authority to generate keys for the users and the roles and manage the role hierarchy, a set of owners who store their private data in the cloud, a set of users $\mathcal{U}$ with whom the owners may wish to share the private data, a set of roles that these users can have, the associated role managers RM[3] and the cloud provider.

$RBE.Setup(\lambda)$: takes as input the security parameter λ and outputs a master secret key mk and a system public key pk. mk is kept secret by the GA while pk is made public to all users of the system.

$RBE.CreateRole(\mathsf{mk}, \mathsf{ID}_R, \mathcal{P}_R)$: an algorithm executed by the GA to create a new role with identity ID_R. $\mathcal{P}_R$ is the set of roles which will be the ancestors of the new role, returns the role secret sk_R, a set of public parameters pub_R for the role and an empty user list $\mathcal{RUL}$ which will list all the users who are the members of that role.

$RBE.CreateUser(\mathsf{mk}, \mathsf{ID}_U)$: an algorithm executed by the GA to add a new user to the system. ID_U is the unique identity of the new user, returns the secret decryption key dk, which the user will use to decrypt all information retrieved from the system.

$RBE.AddUser(\mathsf{pk}, \mathsf{sk}_R, \mathcal{RUL}_R, \mathsf{ID}_U)$: an algorithm executed by the role manage RM of a role to add the user ID_U to the set of users who are members of that role, updates the role's public parameter pub_R and the user list $\mathcal{RUL}_R$ if the user qualifies the role.

$RBE.Encrypt(\mathsf{pk}, \mathsf{pub}_R, M)$: an algorithm executed by the owner to encrypt the private data M, outputs the ciphertext C.

$RBE.Decrypt(\mathsf{pk}, \mathsf{pub}_R, \mathsf{dk}, C)$: an algorithm executed by the user to decrypt the ciphertext C, outputs the plain text message M if the user has the permission to access the data, and fails otherwise.

$RBE.RevokeUser(\mathsf{pk}, \mathsf{sk}_R, \mathcal{RUL}_R, \mathsf{ID}_U)$: an algorithm executed by the role manager RM on input an identity ID_U of the user U, removes U from the $\mathcal{RUL}$ and updates the role's public parameters.

Now let us look at a RBAC example. Assume that Fig. 2 represents a hierarchical role structure of a department in a company. The department has two projects *PL1* and *PL2*, and each project has two sub-roles *PE* and *QE*. The role *PL* inherits permissions from both roles *PE* and *QE* within the project, and there is a role *DIR* that inherits from both role *PL1* and *PL2*. Now the department wants to store confidential documents in the cloud and uses an RBE scheme to encrypt the documents.

Initially, the administrator executes algorithms to set up the system parameters, and generates keys for existing users, that is, the staffs in the department. Assume

[3] In systems where there are small number of users, the GA can act as the role manager to manage the user membership of each role to make the systems compact. However, in large scale systems, it is infeasible for a single party to manage all the users, therefore separate role managers make the user management more flexible and efficient.

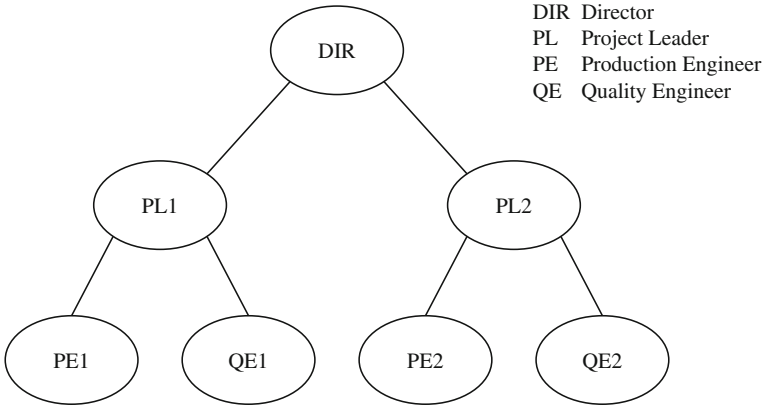


Fig. 2 Hierarchical RBAC

that one user in the role *PL1* has created a document for project 1 and wants to upload it to the cloud for other leaders of the project 1 to review. The user can simply encrypt the document to the role *PL1*, and upload it to the cloud. Other users who are in the role *PL1* can use their own decryption keys to decrypt the document and review. Since the role *DIR* inherits from the role *PL1*, the users in the role *DIR* who are the directors of the department can also decrypt the document using their own decryption keys.

Now let us assume that the company has newly assigned an additional director to the department. This new director will be given a decryption key corresponding to his identity, and his identity will be included in the updated role parameters for the role *DIR*. The new director can now use his own decryption key to view the documents for the department including the one that the leader of project 1 uploaded previously. Note that only the role *DIR* needs to update the role parameters, and other roles are not required to take any action.

Now let us assume that one of the quality engineers from project 2 has resigned from the company. The role parameters for *QE2* need to be updated to exclude the identity of this quality engineer. Even though this quality engineer still holds the previous decryption key which was able to decrypt the documents encrypted to the role *QE2*, s/he will not be able to use it to decrypt any future documents after the role parameters is updated. Similarly, only the role *QE2* needs to update its role parameters and none of the other roles in the department needs to make any change. In addition, other quality engineers of the same role do not need to update their keys, and hence are not affected by the leaving of this quality engineer.

Now consider the situation where a leader of project 1 has been assigned to lead project 2. Two actions need to be performed for this position change. The role *PL1* needs to revoke this user's role membership, and the role *PL2* will need to grant the role membership to this user. To account for this change, both the roles *PL1* and *PL2* need to update their role parameters, but the decryption key of this user remains

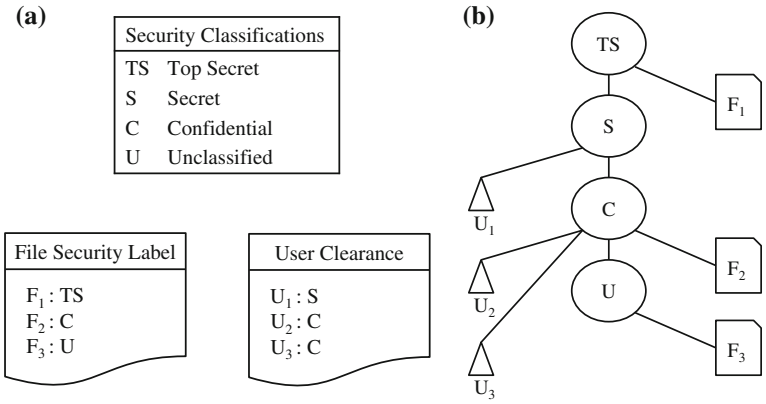


Fig. 3 Cryptographic MAC model. **a** MAC example. **b** Cryptographic MAC model

unchanged. This decryption key cannot be used to decrypt any future document for project 1, but can decrypt documents for project 2 from then on.

Mandatory Access Control (MAC) Model In a MAC model, each resource (object) is associated with a security clearance, and each user (subject) is assigned a clearance. The users are only allowed to access the resources whose security classification level is lower than or equal to their clearance level. We consider an example of a MAC model shown in Fig. 3a. The system security classifications are defined as a set {*Top Secret (TS), Secret (S), Confidential (C), Unclassified (U)*}. Three files (F_1, F_2, F_3) in the system are associated with the security classification (TS, C, U) respectively, and three users (U_1, U_2, U_3) of the system are assigned the (S, C, C). Because of the information flow control in MAC model, U_1 can access files with the security level (S, C, U), and the users U_2 and U_3 can only access files with the security level (C, U).

Now we transform the access policies into a hierarchical structure, and then use an RBE scheme to enforce the MAC policies. The hierarchy is shown in Fig. 3b, where we map the four security classifications to four different roles. We organise the roles following the direction of the information flow propagation, and let the role TS inherits from S, S inherits from C, and C inherits from the role U. Each file is then encrypted to the role that is associated to its security label using an RBE scheme, and users are granted the membership of the roles corresponding to their clearance levels. Then we can see that U_1 can decrypt the files encrypted to the role (S, C, U), and U_2 and U_3 can only decrypt the files encrypted to the role (C, U). Hence the access policy enforced by the RBE scheme is identical to the MAC policy specified in Fig. 3a. Therefore, one can see that we can use RBE schemes to enforce MAC policies. In general, the RBE schemes are capable to be used to enforce the policies of a more complex lattice-based MAC model in a similar way to that we have described above.

3 Role-Based Encryption for Secure Cloud Data Storage

In this section, we discuss the entities involved in a Role-Based Encryption (RBE) scheme for cloud data storage systems. Use of the RBE scheme ensures that only users with specific roles that are allowed by the data owners can decrypt the data, and anyone else, including the cloud providers themselves, will not be able to decrypt the data. Then we describe a specific RBE scheme introduced in [42]. This scheme can work in a hybrid cloud environment which allows an organisation to store data securely in a public cloud, while maintaining the sensitive information related to the organisation's structure, such as the information that defines the role hierarchy, in a private cloud.

3.1 A RBE Based Cloud Storage System

We first illustrates all the parties and their interactions in an RBE system in Fig. 4. The GA is a trusted party who creates the role and user parameters for the system. The GA creates the private keys of the roles and users, and has secure channels by which it can communicate with these entities and transfer their keys to them. The GA is only involved at system setup and to answer calls to $RBE.CreateRole$ and $RBE.CreateUser$. It takes no part in any other operations and so does not need to be online for continuous system operations.

The role manager for each role manages the set of users in a given role. Note that it does not manage the permissions of the roles, only the users. Any data owner is allowed to encrypt data using the role public parameters and thereby add a permission to the role. This activity allows significant flexibility in adding data to the cloud storage, and it could also be controlled by restricting access to the role public parameters, or at least the part required for encryption. Data owners would then make requests of the role manager for it to encrypt data to the role. The role manager, in its function of controlling the user to role mapping, decides which users are to be assigned to the role and which are to be excluded. The policies and mechanisms used to make this decision are not important to the current proposal and so are not further considered. It is assumed that there are some standard authentication mechanisms available that can be performed between the RM and a user.

Users are able to store and retrieve data from the cloud. They can decrypt ciphertexts retrieved from the cloud using the decryption key provided to them by the GA and the public information of a role of which

- they are members, and
- either the ciphertext was encrypted to that role, or that role is a ancestor of the role to which the ciphertext was encrypted.

Note that only users that are members of the role to which the data was encrypted, or are members of one of the ancestor roles of that role, can decrypt the ciphertext.

It is assumed that users keep secret the decryption key given to them by the GA. It is further assumed that users have some credentials that they can use to prove their identity to role managers. Role managers will use these credentials as the basis for deciding whether or not to assign users to the role that they manage.

A user is able to join a role after the owner has encrypted the data for that role, and the user will be able to access that data from then on, and the owner does not need to re-encrypt the data. A user can be revoked at any time (e.g. if the user is found to be malicious), in which case the revoked user will not have access to any data encrypted to this role.

So far only the read permission has been discussed; that is, a user who belongs to a role to which a message was encrypted can decrypt and read the message. Now let us consider the write permission. There can be several cases. The first case is that when only a generic write permission is required. In a RBE scheme, to encrypt a message, a user does not need any secret information. As mentioned previously, the input to the $RBE.Encrypt$ operation does not contain any secret value. Therefore, any user can encrypt a message. Consider another case where write permission needs to be enforced following the same policy as for a read permission, such as that a user wishes to modify an existing encrypted message. Public key encryption schemes used in practice are usually hybrid schemes where secure symmetric encryption schemes

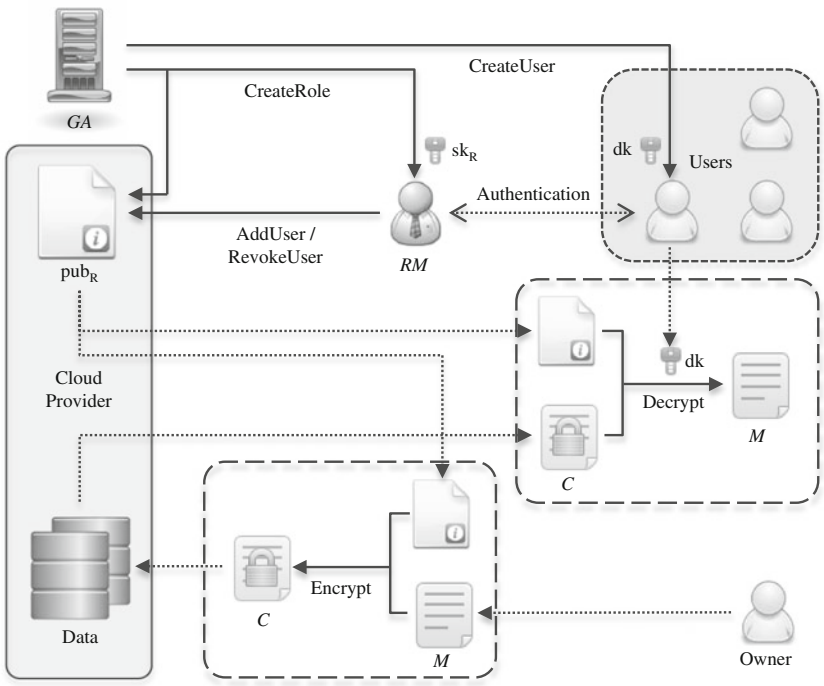


Fig. 4 Role based encryption system

are employed to encrypt messages, and the keys to decrypt the message are encrypted using the public key encryption. In a RBE scheme, what is encrypted to a role could be the symmetric key used to decrypt messages. The user can decrypt the symmetric key of the encrypted message, hence cannot only read the message but also modify the message.

Now let us extend this case to a more general scenario. Assume an owner wishes to assign multiple permissions, such as read, modify, delete and execute, to a role. The owner can create a key for each different permission, and encrypt these keys to this role. Since the user who has access to messages encrypted to this role can recover the keys for the permissions of that role, the user is able to obtain these permissions and gets the ability to carry them out.

3.2 RBE Construction

In this subsection, we describe a concrete RBE construction proposed in [42] to help readers better understand RBE schemes.

The Bilinear Pairings Let $\mathbb{G}_1$, $\mathbb{G}_2$ be two cyclic multiplicative groups of prime order p, and $\mathbb{G}_T$ be a cyclic multiplicative group of prime order p. Let g and h be two random generators where $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$. A bilinear pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ satisfies the following properties:

- *Bilinear:* for $a, b \in \mathbb{Z}_p^*$ we have $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$.
- *Non-degenerate:* $\hat{e}(g, h) \neq 1$ unless $g = 1$ or $h = 1$.
- *Computable:* the pairing $\hat{e}(g, h)$ is computable in polynomial time.

This scheme requires the asymmetric bilinear groups, where the bilinear map takes inputs from two distinct isomorphic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_1 \neq \mathbb{G}_2$. The reason for using the asymmetric bilinear groups is that the generator of one group is kept as part of the master secret key. The benefit of defining two distinct groups is that this scheme can make use of certain families of non-supersingular elliptic curves defined in [32, 4].

RBE Scheme Now we describe the specific RBE scheme as follows,

Setup(λ): Take as input the security parameter λ and generate three groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. Randomly choose two generators $g \in \mathbb{G}_1$ and $h \in \mathbb{G}_2$, two secret values $s, k \leftarrow \mathbb{Z}_p^*$ and two hash functions $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, $H' : \mathbb{G}_T \rightarrow \mathbb{G}_1$ and a secure symmetric encryption scheme Enc. The master secret key mk and system public key pk are defined as

$$\mathsf{mk} = (s, k, g), \quad \mathsf{pk} = (w, v, g^k, h, h^s, ..., h^{s^q}) \text{where } w = g^s, v = \hat{e}(g, h)$$

and q is the maximum number of users or ancestor roles that each role can have.

CreateRole$(\mathrm{mk}, \mathrm{ID}_R, \mathrm{P}_R)$: To create a role R with identity ID_R, we first assume that $\mathcal{P}_R$ is the set of roles $\{\mathsf{ID}_{R_1}, ., \mathsf{ID}_{R_m}\}$ which are the identities of all the ancestor roles of R. GA generates the role secret of R as

$$\mathsf{sk}_R = g^{\frac{1}{s+H(\mathsf{ID}_R)}}$$

and computes the role public parameters as

$$A = h^{(s+H(\mathsf{ID}_R))\prod_{i=1}^{m}(s+H(\mathsf{ID}_{R_i}))},\ B = h^{k(s+H(\mathsf{ID}_R))\prod_{i=1}^{m}(s+H(\mathsf{ID}_{R_i}))}$$

The list of the users who are members of the role is denoted as $\mathcal{RUL}_R$, which is initially set to be empty.

CreateUser(mk, ID_U): To create a user U with identity ID_U in the system, GA simply computes the user secret as

$$\mathsf{dk}_U = g^{\frac{1}{s+H(\mathsf{ID}_U)}}$$

and gives dk_U to the user U. This effectively adds the user to the system. The user will use dk_U as their decryption key in recovering information from the system.

AddUser(pk, sk_{R_i}, $\mathcal{RUL}_{R_i}$, ID_{U_k}): This operation is performed by the role manager RM when it wants to add a user U_k with identity ID_{U_k} to the role R_i. $\mathcal{RUL}_{R_i}$ is the set of n users that are members of the role R_i (possibly empty) and U_k is not in $\mathcal{RUL}_{R_i}$. The role manager RM chooses two random values $r_i, t_i \leftarrow \mathbb{Z}_p^*$ if $\mathcal{RUL}_{R_i}$ is empty, or uses the existing r_i, t_i otherwise. Then RM computes

$$K_i = v^{r_i},\ T_i = g^{-t_i},\ W_i = w^{-r_i},\ V_i = h^{r_i\cdot(s+H(\mathsf{ID}_{U_k}))\prod_{j=1}^{n}(s+H(\mathsf{ID}_{U_j}))}$$

and

$$S_i = H'(K_i)\cdot \mathsf{sk}_{R_i}\cdot g^{kt_i} = H'(v^{r_i})\cdot g^{\frac{1}{s+H(\mathsf{ID}_{R_i})}+kt_i}$$

and adds ID_{U_k} into $\mathcal{RUL}_{R_i}$. When RM finishes the computation, it sends T_i to the cloud via a secure channel, and outputs the role public information as

$$(\mathsf{ID}_{R_i}, A_i, B_i, W_i, V_i, S_i, \mathcal{RUL})$$

Encrypt(pk, pub_{R_x}, M): Assume that the owner of the message M wants to encrypt M for the role R_x. The owner randomly picks $z \leftarrow \mathbb{Z}_p^*$, and using $\mathsf{pk} = (w, v, g^k, h, h^s, \cdots, h^{s^m})$ and the public information of the role R_x to computes

$$C_1 = w^{-z},\ C_2 = A_x{}^z,\ C_3 = B_x{}^z,\ K = v^z$$

and the ciphertext is output as a tuple $(C_1, C_2, C_3, Enc_K(M))$.

Decrypt(pk, pub_{R_i}, dk_{U_k}, C): When a user U_k wants to decrypt the ciphertext C stored in the cloud, s/he first requests the ciphertext from the cloud, and the cloud returns the following tuple to U_k

$$\{C_1, C_2, \overline{C}_3, Enc_K(M), \text{where } \overline{C}_3 = \hat{e}(T_i, C_3)\}$$

Assume role R_x has a set $\mathcal{R}$ of ancestor roles, the set $\mathcal{M} = R_x \cup \mathcal{R}$ has m roles $\{R_1, ..., R_m\}$. There is a set $\mathcal{N}$ of n users $\{U_1, ..., U_n\}$ in $R_i \in \mathcal{M}$, and the user $U_k \in \mathcal{N}$ who is a member of the role R_i wants to decrypt the message $(Enc_K(M))$.

After receiving the ciphertext from the cloud, U_k computes

$$K' = (\hat{e}(C_1, h^{p_{i,\mathcal{M}}(s)}) \cdot \hat{e}(\frac{S_i}{H'(K_i)}, C_2) \cdot \overline{C}_3)^{\frac{1}{\prod_{j=1, j\neq i}^{m} H(\mathsf{ID}_{R_j})}} \tag{1}$$

where

$$K_i = (\hat{e}(\mathsf{dk}_{U_k}, V_i) \cdot \hat{e}(W_i, h^{p_{k,\mathcal{N}}(s)}))^{\frac{1}{\prod_{j=1, j\neq k}^{n} H(\mathsf{ID}_{U_j})}} \tag{2}$$

and

$$p_{i,\mathcal{M}}(s) = \frac{1}{s} \cdot \left(\prod_{j=1, j\neq i}^{m} (s + H(\mathsf{ID}_{R_j})) - \prod_{j=1, j\neq i}^{m} (H(\mathsf{ID}_{R_j})) \right).$$

$$p_{k,\mathcal{N}}(s) = \frac{1}{s} \cdot \left(\prod_{j=1, j\neq k}^{n} (s + H(\mathsf{ID}_{U_j})) - \prod_{j=1, j\neq k}^{n} (H(\mathsf{ID}_{U_j})) \right).$$

Using the key K', U_k can decrypt the $Enc_K(M)$, therefore recover the message.

RevokeUser(pk, sk_R, $\mathcal{N}$, ID_U): To revoke a user U_k from a role R_i which has a set $\mathcal{N}$ of n users, and $U_k \in \mathcal{N}$, the role manager RM first removes ID_{U_k} from role user list $\mathcal{RUL}$. Then RM chooses two random values $r_i', t_i' \leftarrow \mathbb{Z}_p^*$ and computes

$$K_i' = v^{r_i'},\ T_i' = g^{-t_i'},\ W_i' = w^{-r_i'},\ V_i' = h^{r_i' \cdot \prod_{j=1, j\neq k}^{n} (s + H(\mathsf{ID}_{U_j}))}$$

and

$$S_i' = H'(K_i') \cdot \mathsf{sk}_{R_i} \cdot g^{kt_i'} = H'(v^{r_i'}) \cdot g^{\frac{1}{s+H(\mathsf{ID}_{R_i})} + kt_i'}$$

The cloud needs to update T_i to T_i', and the role public information now changes to

$$(\mathsf{ID}_{R_i}, A_i, B_i, W_i', V_i', S_i', \mathcal{RUL})$$

Note that the public value S_i of the role R_i has been updated to S_i'. Therefore the new K_i' will be needed in computing K' in Eq. 1. Since the identity of the revoked user U_k is not included in computing the new value V_i', U_k will not be able to compute the new K_i' using Eq. 2. Hence U_k cannot decrypt messages using K'.

3.3 Remarks

In the above scheme, the owner of the message is able to specify a role R and encrypt the message M to this role. Only users who are members of this role and/or its ancestor roles can decrypt the message. For an individual role, the randomized role secret can only be recovered by the users in that role, and therefore these users are able to recover the message M. Clearly a user who cannot decrypt the randomized role secret cannot learn anything about the content of the message.

The use of the randomized role secret efficiently solves the user revocation here. In the schemes that use fixed secret values to decrypt messages, user revocation is impossible unless new secret values are generated to replace the old secret values known to the revoked users. In this scheme, it is assumed that the cloud does not collude with revoked users. It is also possible to remove this assumption by using a hybrid cloud structure which we will not discuss in this chapter.

Overall, this scheme has all the features that we discussed in Sect. 2.3, and hence can be used to enforce both RBAC and MAC policies in cloud storage systems as described in Sect. 2.3.

4 Administration in Role-Based Access Control

RBAC has been widely used for security administration in distributed systems since being first formalised in the 1990's. However, the administration of RBAC systems themselves has been less widely studied. In small RBAC systems, a central authority is usually sufficient to manage all the users and permissions. However large-scale RBAC systems may have hundreds or even thousands of roles and hundreds of thousands of users and permissions. In such cases, it becomes impractical to centralise the task of managing these users and permissions, and their relationships with the roles with a small team of security administrators. Therefore, decentralising the administration tasks of RBAC systems is an important issue when developing such large-scale role-based systems.

Several administrative RBAC (ARBAC) models have been developed to provide solutions to decentralise the administration privileges. The administrative model for RBAC was first considered in [35], and an comprehensive model was proposed in this work, called ARBAC97. It was later extended and improved in [13–15, 33, 34, 38]. A common feature of these works is managing a RBAC system using RBAC itself. The administration privileges are decentralised to a set of administrative roles in these models, and administrative policies are specified to limit the privileges of administrative roles. Each administrative role is assigned an administration domain, and the role is allowed to perform administration task only on the roles that are covered by the administration domain. Next we review two existing administrative models for RBAC systems.

4.1 ARBAC97 Model

The challenge of administering RBAC was first considered in [35] where a comprehensive administrative model, ARBAC97, was introduced. ARBAC97 is based on the RBAC model defined in [36]. In ARBAC97, a RBAC system is administered by an administrative RBAC, which contains a set of administrative roles AR that are separate from the roles R in the normal RBAC system.

The ARBAC97 model defines three components: URA97 for user-role assignment, PRA97 for permission-role assignment, and RRA97 for role-role assignment. In each component, the ability to perform the assignment is associated with administrative roles. In other words, the administrative roles have control over the administrative operations in the normal RBAC. For each administrative role in these components, the authority range of the role is specified by a concept of *role range*, which is defined as a set of roles in the normal RBAC, and is denoted by following notations

$$
\begin{aligned}
[x, y] &= \{r \in R \mid x \leq r \wedge r \leq y\} \\
(x, y] &= \{r \in R \mid x < r \wedge r \leq y\} \\
[x, y) &= \{r \in R \mid x \leq r \wedge r < y\} \\
(x, y) &= \{r \in R \mid x < r \wedge r < y\}
\end{aligned}
$$

where $x < y$ means that role y inherits the permissions from role x.

The URA97 component defined the following two relations: *can-assign:* $(a, \mathcal{C}, \mathcal{R})$ and *can-revoke:* $(a, \mathcal{R})$, where a is an administrative role, and $\mathcal{R}$ is either a role range or a role set that has been explicitly specified. $\mathcal{C}$ denotes a prerequisite condition which is a boolean expression using the operators $\neg$, $\vee$, $\wedge$ on regular roles of the system. Sandh et al. [35] gives the example role hierarchies as shown in Fig. 5 to illustrate the model. In this example, *can-assign*$(DSO, ED \wedge \neg PL1, (ENG2, PL2])$ means that the members of the administrative role DSO can assign a user, who is currently a member of the role ED and not a member of the role $PL1$, to be a member of regular roles of the set $\{PL2, PE2, QE2\}$. *can-revoke*$(PSO2, [ENG2, PL2])$ means that the members of the administrative role $PSO2$ can revoke membership of a user from any regular role in the set $\{PL2, PE2, QE2, ENG2\}$. PRA97 is defined as the dual of the URA97 model. Thus it has two relations *can-assignp:* $(a, \mathcal{C}, \mathcal{R})$ and *can-revokep:* $(a, \mathcal{R})$ which are similar to the *can-assign* and *can-revoke* in the URA97.

The RRA97 component defined five relations: *can-assigna*, *can-revokea*, *can-assigng*, *can-revokeg*, *can-modify*. The relation *can-modify* specifies the authorization of operations role creation, deletion, and modification, and is defined as $(a, \mathcal{R})$ where a is an administrative role, and $\mathcal{R}$ is a role range. The role range in RRA97 is a special case where no end points is included, and any role range referenced in the *can-modify* relation is called an *authority range*. For example, *can-modify:* $(PSO1, (ENG1, PL1))$ means that the members of the administrative role $PSO1$ can create, delete, and modify roles in the range $(ENG1, PL1)$. In order to

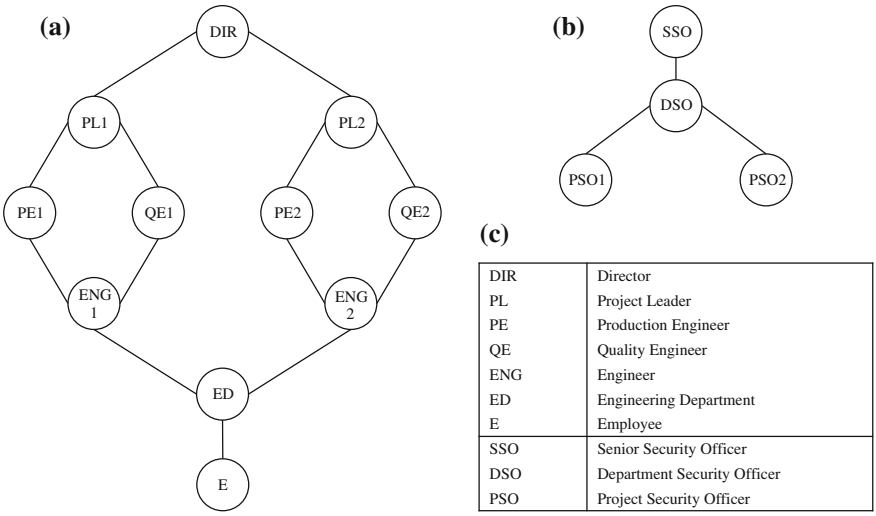


Fig. 5 ARBAC97 example role hierarchies. **a** Roles. **b** Administrative roles. **c** Legend

maintain global consistency when modifying roles, several restrictions have been placed for defining the *authority range* in RRA97. First, authority ranges do not overlap partially. In the example, as the role hierarchies, $(E2, DIR)$ and $(ED, PL1)$ are partially overlapping, they are not allowed to be defined at the same time. Secondly, authority ranges must be encapsulated. ARBAC97 defines the encapsulated range as follows.

Definition 1 *A range* (x, y) *is said to be encapsulated if for all roles* $r_i \in (x, y)$ *and all roles* $r_e \notin (x, y)$,

$$r_e > r_i \Leftrightarrow r_e > y, \text{ and } r_e < r_i \Leftrightarrow r_e < x$$

The ARABC97 model further defined the restrictions on individual operations in the relation *can-modify*, including role creation, role deletion, edge insertion, and edge deletion. ARBAC97 has been extended in ARBAC99 [38] and ARBAC02 [34] where there have been changes to the URA and PRA relations.

4.2 SARBAC Model

SARBAC is another administrative model for RBAC. It was first proposed in [14, 15], and then improved in [13] where a role-based administration template (RBAT) model was proposed. The intention of SARBAC is to improve the RRA model in ARBAC97. RRA97 was based on encapsulated ranges, which are relatively complicated to

administer and require a significant effort in deciding an administrative policy. In order to simplify the RRA model, SARBAC introduced the concept of an administrative scope, which is defined based on the role hierarchies. Similar to the authority range in the ARBAC97, an administrative scope is used to specify a set of roles that can be modified by an administrative role. In this chapter, we follow the description of SARBAC given in [13].

First, let us look at the notations given in [13]. Let $s \in R$; define $\uparrow s = \{r \in R : r \geq s\}$ and $\downarrow s = \{r \in R : r \leq s\}$. The expression $\uparrow s \cup \downarrow s$ is denoted as $\updownarrow s$. The administrative scope is defined as follows.

Definition 2 *The administrative scope of a role r, denoted by $\sigma(r)$, is defined to be*

$$\sigma(r) = \{s \in \downarrow r : \uparrow s \subseteq \updownarrow r\}$$

Based on this definition, a role-based administration template (RBAT) was introduced. RBAT defines a single relation *can-administer:* (a, r), which means that an administrative role a can control the set of roles in $\sigma(r)$. For example, *can-administer*$(PSO1, PL1)$ means that the administrative role $PSO1$ can create, delete, and modify the roles in $\sigma(PL1)$, which specifies the set of roles $\{PL1, PE1, QE1, ENG1\}$.

One feature of SARBAC is that when some RRA operations affect the authority ranges of the administrative roles, no new relation policy needs to be specified; the administrative scope changes following the updating of the role hierarchy, while in ARBAC97, these operations are not allowed. RBAT formalised the relations between the RRA operations and the administrative scopes by defining the scope preserving hierarchy operations and the preserving conditions of the scopes.

5 Administrative Model for Role Based Encryption Schemes

When using the above ARBAC models to manage the RBAC systems which are using cryptographic RBAC schemes, the issue of secure enforcement of the administrative polices of the administrative models becomes significant. In RBAC models, the privileges of each administrator are restricted by the systems themselves, so that the administrators cannot change the roles that they do not have permissions to change. However the existing administrative RBAC models cannot work with cryptographic RBAC in a distributed environment, as the administrative policies cannot be enforced and there is no authority that can restrict the privileges of the administrative roles.

In this section, we describe a trusted administrative model AdC-RBAC described in [41] that can manage and enforce role-based access policies for RBE schemes in large-scale cloud systems. The AdC-RBAC model uses cryptographic techniques to ensure that the administrative tasks such as user, permission and role management are performed only by authorised administrative roles. Any other party, including the cloud providers themselves, cannot change RBAC systems and policies. This model

uses role-based encryption techniques to ensure that only administrators who have the permissions to manage a role can add/revoke users to/from the role and owners can verify that a role is created by qualified administrators before giving out their data. We also show how this model can be used in an untrusted cloud while guaranteeing its security using cryptographic and trusted access control enforcement techniques. This model consists of three components: UAM for user membership management, PAM for permission management, and RAM for role management.

5.1 User Administration Model

In RBE schemes, user membership administration has already been decentralised. The management of user membership for each role is controlled by a role secret key sk_R which is used as an input parameter to the *AddUser* and *RevokeUser* operations. Only the parties who hold this key can add/revoke users to/from this role. Therefore, the following relation is defined for UAM.

Definition 3 *User management policies in AdC-RBAC are specified by the following relation*

$$\text{can} - \text{manage} \subseteq AR \times \mathcal{R}$$

where AR is the set of administrative roles and $\mathcal{R}$ is either a role range or a role set that has been explicitly specified.

The example shown in Table 2a means that the members of the administrative role $PSO1$ can manage the user membership of the roles specified by the range $[ENG1, PL1]$. To enforce this relation in UAM, the secret keys sk_R of the roles that are within the range are encrypted to the role $PSO1$. Note that the problem of specifying the role range has been transformed into the problem of encrypting a set of role secret keys. The concepts of partial overlap and incomparability have been introduced in [35] to define restrictions for the ARBAC97 model. The definitions of these terms used in the context of AdC-RBAC model are as follows.

Definition 4 *Assume that there exist two key sets K_1 and K_2 which correspond to two role ranges $\mathcal{R}_1$ and $\mathcal{R}_2$. Two ranges are said to be overlap partially if $K_1 \cap K_2 \neq \emptyset$*

Table 2 Example of can-manage

Administrative role	Role range
(a) *Single relation*	
PSO1	[ENG1, PL1]
(b) *Multiple relations*	
DSO	[DIR, DIR]
PSO1	[ENG1, PL1]
PSO2	[ENG2, PL2]

and $K_1 \nsubseteq K_2$ *and* $K_2 \nsubseteq K_1$. *Ranges* $\mathcal{R}_1$ *and* $\mathcal{R}_2$ *are said to be incomparable if* $K_1 \cap K_2 = \emptyset$.

In UAM, to avoid any potential conflict that may be caused by managing the user membership of a role by multiple administrative roles, the following restriction is introduced.

Definition 5 *In UAM, role ranges are incomparable.*

In some cases, the system may want an administrative role to have full control over another administrative role; that is, one role range is allowed to be the superset of another role range. In UAM, this can be achieved by the inheritance in the encryption. Table 2b shows a more complex scenario which contains a list of relations.

First, assume that the administrative roles are organised in a hierarchy as shown in Fig. 5b, and that the role DSO inherits permissions from the role $PSO1$ and $PSO2$. Corresponding set of role secret keys to each administrative role can be encrypted using a RBE scheme. Assume that the secret keys for the roles are encrypted to the administrative roles following the relations specified in Table 2b. Since the role DSO can decrypt the data encrypted to the role $PSO1$ and $PSO2$, it can recover the secret keys for roles in the set $[DIR, DIR] \cap [ENG1, PL1] \cap [ENG2, PL2]$; hence it can manage the user membership of these roles. Therefore the effective range that the role DSO can manage is $(ED, DIR]$.

5.2 Permission Administration Model

In a RBE system, all the owners who can access the system are able to encrypt data to the roles; hence in the general case, anyone is allowed to encrypt data to the roles in a RBAC system. Since RBE schemes do not have specific requirements on permissions-role assignment, there is no restriction on the permission assignment in RBE schemes. Therefore, the permission administration model does not have to be defined in the AdC-RBAC model.

However, in some cases, the RBAC system may want to allow only certain specific administrative roles to encrypt data to roles. In RBE schemes, role public parameters pub_R are required for encrypting a data to a role. These role parameters are defined as public, so that the data can be encrypted to the roles by any parties using these parameters as part of the encryption key. In order to restrict permission assignment in PAM, these role parameters are encrypted to the administrative roles who are permitted to assign permissions to these roles, so that only the authorised administrative roles can encrypt data to these roles. The following relations are defined in PAM.

Definition 6 *Permission assignment policies in AdC-RBAC are specified by the following relation*

$$\text{can} - \text{assign} \subseteq AR \times \mathcal{R}$$

Table 3 Example of can-assign

Administrative role	Role range
(a) *Single relation*	
PSO1	[E, PL1]
(b) *Multiple relations*	
DSO	[DIR, DIR]
PSO1	[E, PL1]
PSO2	[ED, PL2]

where AR is the set of administrative roles and $\mathcal{R}$ is either a role range or a role set that has been explicitly specified.

The example shown in Table 3a means that the members of the administrative role $PSO1$ can assign permissions to the roles specified by the range $[E, PL1]$. To enforce this relation in PAM, the parameters pub_R of the roles within the range are encrypted to the role $PSO1$.

Table 3b shows an example with multiple relations. Different from UAM, one administrative role assigning permissions to a normal role does not affect permissions assignment of other administrative roles. Therefore there is no restrictions in PAM, and the specified ranges are allowed to overlap with each other. Using this example, two different modes for PAM are given as *flat mode* and *hierarchy mode*.

Flat mode In a *flat mode*, the administrative roles are organised in a flat manner; that is, these roles do not inherit permissions from each other. Each administrative role is only allowed to assign permissions to the roles that are specified in the role range in the relation. In the example, DSO can only assign permissions to the role DIR, and the role $PSO1$, $PSO2$ can only assign permissions to the roles in the ranges $[ENG1, PL1]$, $[ENG2, PL2]$ respectively. To encrypt the roles' parameters (which are used to encrypt data) to the administrative roles, an identity-based encryption (IBE) scheme is sufficient. Administrative roles can use their private keys to decrypt the parameters pub_R of the roles for which they have the authority to assign permissions; hence they can encrypt the data using the parameters of the corresponding role.

Hierarchy mode In a *hierarchy mode*, the administrative roles are organised in a hierarchical manner; that is, these roles can inherit permissions from other roles. As shown in Fig. 5b, the role DSO inherits permissions from the role $PSO1$ and $PSO2$. Hence the difference from the *flat mode* is that in the hierarchy mode, the role DSO can assign permissions not only to the role DIR, but also to all the roles to which the role $PSO1$ and $PSO2$ can assign permissions. The IBE scheme cannot be used to encrypt the role parameters in this scenario as it cannot reflect the relationship among these administrative roles. Thus we need to use the RBE scheme for the encryption in this mode. Assume that the parameters for the regular roles are encrypted to the administrative roles following the relations specified in Table 3b. Since the role DSO can decrypt the data encrypted to the role $PSO1$ and $PSO2$, it can recover the parameters of all the roles in the range $[E, DIR]$; hence it can assign permissions to all these roles.

5.3 Role Administration Model

In RBE schemes, only the single administrator GA can create roles or modify roles. In order to decentralise these operations, the single administrator needs to delegate the privileges to create and modify roles to multiple administrative roles. However, the role hierarchy is constructed based on the system master secret, and the master secret cannot be given to the administrative roles. In RAM, any administrative role can administer the regular roles, and any party is able to verify whether or not a role in the RBAC system is created/modified by an authorised administrative role. The relation is defined as follows.

Definition 7 *Role administration policies in AdC-RBAC are specified by the following relation*

$$\text{can-administer} \subseteq AR \times \mathcal{R}$$

where AR is the set of administrative roles and $\mathcal{R}$ is either a role range or a role set that has been explicitly specified.

The same restriction on authority ranges in this relation is required as in the case of ARBAC97.

Definition 8 *In RAM, authority ranges of the administrative roles do not overlap partially and must be encapsulated, and the edge roles of the authority ranges cannot be modified.*

In RAM, identity-based signature schemes is used to certify the authority of the administrative roles. Let $\mathcal{D}_{AR}(M)$ denote data M's signature which is signed to the identity of the administrative role AR. In order to facilitate the verification of the authority, a set of administrative parameters AP is defined for each regular role. Now consider the relation *can-administer:* $(a, (x, y))$ which means that the members of an administrative role a can administer the roles in the range (x, y). In this relation, the following parameters are associated with the edge roles of the range,

$$\langle \mathsf{ID}_r, \mathcal{P}_r, \mathcal{S}_r, AR_r, \tau, \mathcal{D}_s(P_r) \rangle$$

where P_r denotes $\mathsf{ID}_r||\mathcal{P}_r||\mathcal{S}_r||AR_r||\tau$, ID_r is the identity of the edge role of the range, $\mathcal{P}_r$ is the set of the identities of r's immediate senior roles, $\mathcal{S}_r$ is the set of the identities of r's immediate junior roles, AR_r is the identity of the administrative role who can administer the range, τ denotes the type of the edge: upper bound or lower bound, and $\mathcal{D}_s(P_r)$ is the signature on the parameters, which is issued by the most senior administrative role who defines the relations in RAM. These parameters are computed and attached to the role when this role is specified as the edge of the authority range by the most senior role s. For other regular roles within the range, the parameters associated with them are as follows,

$$\langle \mathsf{ID}_r, \mathcal{P}_r, \mathcal{S}_r, AR_r, \mathcal{D}_a(P_r) \rangle$$

where P_r denotes $\mathsf{ID}_r||\mathcal{P}_r||\mathcal{S}_r||AR_r$, ID_r is the identity of the regular role within the range, $\mathcal{P}_r$ is the set of the identities of r's immediate senior roles, $\mathcal{S}_r$ is the set of the identities of r's immediate junior roles, AR_r is the identity of the administrative role who can administer the range, and $\mathcal{D}_a(P_r)$ is the the parameters' signature which is issued by the administrative role AR_r. These parameters are computed and attached to the role when this role is created or modified by the administrative role.

Assume that a user of the system wants to encrypt some data to a role, or decrypt some data that are encrypted to a role. Then the user will need to verify the parameters AP of the role to check whether this role is certified by the authorised administrative roles of the system. Now we describe an algorithm, Algorithm 1, for this verification process. In Algorithm 1, a function $verify$ is defined to denote the verification of an ID-based signature. This function $verify(\mathcal{D}_z, \mathsf{ID}_z)$ verifies a signature $\mathcal{D}_z$ against the identity of the role z; it returns $true$ if the verification succeeds, and otherwise returns $false$. When running this algorithm to verify a role, a user gets a $true$ if the last change of the role was made by an authorised administrative role; otherwise user gets a $false$. In addition, creating a role out of the specified role range by an administrative role will cause failures in the verification of all the roles which have inheritance relationships with this role.

In RAM, the parameters AP associated with each role specifies only one administrative role. In the example shown in Table 4, the authority range of the role DSO is the superset of the role $PSO1$. To simplify the key management, an hierarchical ID-based signature (HIBS) scheme can be used by administrative roles to sign the role associated parameters. When DSO modifies the roles in $(ENG1, PL1)$, it signs the parameters AP associated with the roles using its own private key, and HIBS schemes allows the signature to be verified against the identity of the role $PSO1$. Similarly, the signature generated by DSO can also be verified against the identity of the role $PSO2$.

5.4 Administration Example with RBE Schemes

The above described AdC-RBAC model provides a framework for the administration of RBAC systems. It is designed to work with RBE schemes described in Sect. 2.3. To integrate this administrative model with a RBE scheme, the RBE scheme needs to be modified to meet the requirements of the AdC-RBAC model. In this subsection, we show how to integrate the AdC-RBAC model with the RBE scheme described in

Table 4 Example of can-administer

Administrative role	Role range
DSO	(E, DIR)
PSO1	(ENG1, PL1)
PSO2	(ENG2, PL2)

Algorithm 1 Algorithm to Verify the Authorisation

```
function IsEdge(r, a, t)
    if role r has parameter τ and τ = t and AR_r = ID_a then
        return true
    end if
    return false
end function

function VerifyRole(r, a, t)
    if role r has parameter τ then
        return verify(D_s(P_r), ID_s)
    else
        return verify(D_a(P_r), ID_a)
    end if
end function

function VerifyAuth(r, a, t)
    State_r ← false
    if IsEdge(r, a, t) then
        State_r ← verify(D_s(P_r), ID_s)
    else if VerifyRole(r, a, t) then
        if t = upper-bound then
            R ← P_r
        else if t = lower-bound then
            R ← S_r
        end if
        for each x ∈ R do
            State_r ← State_x
            if State_r = false then          ▷ Only verify the role that has not been verified
                if a ≥ AR_x then             ▷ Check if a inherits the permissions from AR_x
                    State_r ← VerifyAuth(x, a, t)
                end if
                if State_r = false then
                    return State_r
                end if
            end if
        end for
    end if
    return State_r
end function

Require: Initiate a State_r for each role r ∈ R to false
  Input w          ▷ Checking if the role w is administered by the authorised administrator.
  b ← false
  if VerifyAuth(w, AR_w, upper-bound) then          ▷ Verify the senior roles of the role w.
      b ← VerifyAuth(w, AR_w, lower-bound)          ▷ Verify the junior roles of the role w.
  end if
  Output b     ▷ true if w is administered by the authorised administrator, or false otherwise.
```

Sect. 3. We only describe the changes to the individual operations for each algorithm of the RBE scheme.

Since this RBE scheme allows user membership of each role to be managed by the role manager of each individual role, the UAM model can be used directly to administer the user membership in a RBE scheme. An additional step needs to be taken when a user joins the system. The system needs to execute *CreateUser* to issue a secret key for the new user. This step ensures that the users to be managed in the system are authentic users.

In this RBE scheme, the owner who is able to encrypt the data to a role can be anyone who can access the system. The owner does not have to be a user in the RBAC system. This approach provides considerable flexibility for permission assignment; hence the PAM component can be used in the RBE scheme directly without any prerequisite conditions.

The RAM component requires that all the administrative roles have privileges to perform the role operations such as role creation and modification instead of a single authority. However, in this RBE scheme, these operations are executed by a single group administrator GA. We modify two operations of this RBE scheme to decentralise these privileges.

Setup(λ): GA executes the same algorithm as in the original RBE scheme to setup the system. Then additional parameters are computed as role public keys $\mathsf{rk} = (h^k, \cdots, h^{k^m})$. These parameters are public only to the administrative roles.

CreateRole(mk, ID_R): To create a role with identity ID_R, the GA only generates the role secret key sk_R and sends it the the manager of the role via a secure channel.

In the modified scheme, the GA still needs to execute *CreateRole* to include the new role into the system. This step ensures that the roles which are to be administered in the system are authentic. Note that the modified operation *CreateRole* does not specify any role hierarchy information, as the role parameters A, B which indicate the role hierarchy information are not computed by GA any more. Since we let GA generate additional role public keys rk in *Setup*, all the administrative roles are able to compute the parameter B. When an administrative role wants to add or modify a role in the RBAC system, the parameter B can be re-generated for the role based on the role hierarchy. We let the owner in the system compute the parameter A to avoid the risks in the scenario where malicious administrative roles compute the incorrect parameters B.

By adopting the above mentioned modifications in the RBE scheme described in Sect. 3, the privileges to administer the roles have been decentralised; hence the RAM component can be used to administer the RBE scheme.

5.5 Remarks

Administration of large-scale RBAC systems is a challenging problem. Many administrative models for RBAC systems have been proposed to decentralize the administration tasks associated with roles. Cryptographic RBAC schemes can protect data stored in cloud from unauthorised access by enforcing RBAC access policies using

cryptographic techniques. Similarly, the AdC-RBAC model integrates cryptographic techniques with ARBAC models to enforce administrative RBAC policies. This model guarantees that RBAC model integrated with cryptographic techniques does not loss flexibilities in administration while being used in cloud data storage systems.

6 Conclusion

In this chapter, we first discussed the security issues in cloud storage, and the importance of using access control in a cloud storage system. Then we described several well-known access control models which can be used to protect the privacy of data stored in a cloud storage system. These access control models are summarised in Table 5. Due to the distributed nature of the cloud system, one approach to enforce access policies in these access control models involves integrating cryptographic techniques with these models. Therefore, we have discussed the cryptographic encryption schemes that can be used along with these access control models to ensure the enforcement of the access policies.

We then described a specific role-based encryption (RBE) scheme which can be used to protect data privacy in a cloud storage system. In order to manage the large-scale RBAC system, the administration tasks need to decentralised. We have reviewed the existing administrative RBAC models which were developed to provide solutions to decentralise the administration privileges. Since the administrative RBAC policies in these ARBAC models also need to be enforced in a cloud environment, we then described a trusted administrative model AdC-RBAC that can manage and enforce role-based access policies for cryptographic RBAC schemes in large-scale cloud systems.

Table 5 Access control model summary

Access control model	Features	Disadvantages
Mandatory access control	– Security label based –Simple rules –Centralised security policies	–No user based –Lack of flexibility –Difficult in implementation
Discretionary access control	–Identity based –User oriented –Decentralised security policies	–Complex in management –Owner dependence –Difficult in auditing
Attribute-based access control	–Attribute-privilege based –Flexible in management	–Policy management could be complex when attribute amount is large
Role-based access control	–Role-privilege based –Permission inheritance –Separation of duty	–Role management could be complex

Using cryptographic RBAC schemes ensures that only users with specific roles that are allowed by the data owners can decrypt the data, and anyone else, including the cloud providers themselves, will not be able to decrypt the data. It is worth noting that the security of a RBAC system using one of these schemes is under the assumption that the authorised users and roles behave in a trusted manner so they do not breach the RBAC policies. However, in a cloud storage system that uses RBAC to control the access to the data, an authorised user of the system may leak the data in the cloud to unauthorised users; or an authorised user may be excluded from accessing the permissions of the role that have been legitimately assigned to the user by a malicious administrator of the system. Such issues rely on trust aspects in these systems. Therefore, trust models for RBAC systems is desired to work with cryptographic RBAC schemes together to enhance the system security.

References

1. Akl Selim G, Taylor Peter D (1983) Cryptographic solution to a problem of access control in a hierarchy. ACM Trans. Comput. Syst. 1(3):239–248
2. Armbrust Michael, Fox Armando, Griffith Rean, Joseph Anthony D, Katz Randy H, Konwinski Andy, Lee Gunho, Patterson David A, Rabkin Ariel, Stoica Ion, Zaharia Matei (2010) A view of cloud computing. Commun. ACM 53(4):50–58
3. Atallah MJ, Frikken KB, Blanton M (2005) Dynamic and efficient key management for access hierarchies. In: ACM conference on computer and communications security, pp 190–202, 7–11 Nov 2005
4. Barreto PSLM, Naehrig M (2005) Pairing-friendly elliptic curves of prime order. Selected areas in cryptography, vo 3897 of Lecture notes in computer science, Springer, Berlin, pp 319–331, 11–12 Aug 2005
5. Bell DE, LaPadula LJ (1975) Secure computer systems: mathematical foundations and model. Technical Report M74–244, MITRE Corporation, Bedford, MA
6. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. IEEE symposium on security and privacy, IEEE Computer Society, pp 321–334
7. Boneh D, Boyen X, Goh EJ (2005) Hierarchical identity based encryption with constant size ciphertext. EUROCRYPT, Lecture notes in computer science, vol 3494. Springer, Berlin, pp 440–456. 22–26 May 2005
8. Boneh D, Gentry C, Waters B (2005) Collusion resistant broadcast encryption with short ciphertexts and private keys. CRYPTO, Lecture notes in computer science, vol 3621. Springer, Berlin, pp 258–275, 14–18 Aug 2005
9. Boneh D, Hamburg M (2008) Generalized identity based and broadcast encryption schemes. In: ASIACRYPT, Lecture notes in computer science, vol 5350. Springer, Berlin, pp 455–470, 7–11 Dec 2008
10. Chase M (2007) Multi-authority attribute based encryption. In: TCC, Lecture notes in computer science, vol. 4392. Springer, Berlin, pp 515–534. 21–24 Feb 2007
11. Chase M, Chow SSM (2009) Improving privacy and security in multi-authority attribute-based encryption. In: ACM conference on computer and communications security, pp 121–130
12. Cheung L, Newport C (2007) Provably secure ciphertext policy abe. In: ACM conference on computer and communications security, pp 456–465
13. Crampton J (2005) Understanding and developing role-based administrative models. In: ACM conference on computer and communications security, pp 158–167. 7–11 Nov 2005
14. Crampton Jason, Loizou George (2003) Administrative scope: a foundation for role-based administrative models. ACM Trans. Inf. Syst. Secur. 6(2):201–231

15. Crampton J, Loizou G (2002) Administrative scope and role hierarchy operations. SACMAT, pp 145–154. 3–4 June 2002
16. Delerablée C, Paillier P, Pointcheval D (2007) Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In: Pairing, Lecture notes in computer science, vol 4575. Springer, Berlin, pp 39–59
17. Di Vimercati SDC, Foresti S, Jajodia S, Paraboschi S, Samarati P (2007) A data outsourcing architecture combining cryptography and access control. In: Proceedings of the 2007 ACM workshop on Computer security architecture, pp 63–69, 2 Nov 2007
18. Di Vimercati SDC, Foresti S, Jajodia S, Paraboschi S, Samarati P (2007) Over-encryption: management of access control evolution on outsourced data. In: Proceedings of the 33rd international conference on Very large data bases VLDB, pp 123–134. 23–27 Sept 2007
19. Emura K, Miyaji A, Nomura A, Omote K, Soshi M (2009) A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In: ISPEC, Lecture notes in computer science, vol 5451. Springer, Berlin, pp 13–23. 13–15 April 2009
20. Ferraiolo DF, Kuhn DR (1992) Role-based access controls. In: 15th national computer security conference, vol 1–2. National Institute of Standards and Technology, National Computer Security Center, pp 554–563. 13–16 Oct 1992
21. Fiat A, Naor M (1993) Broadcast encryption. In: CRYPTO, Lecture notes in computer science, vol 773. Springer, Berlin, pp 480–491. 22–26 Aug 1993
22. Garay JA, Staddon J, Wool A (2000) Long-lived broadcast encryption. In: CRYPTO, Lecture notes in computer science, vol 1880. Springer, Berlin, pp 333–352. 20–24 Aug 2000
23. Gentry C, Silverberg A (2002) Hierarchical id-based cryptography. In: ASIACRYPT, Lecture notes in computer science, vol 2501. Springer, Berlin, pp 548–566
24. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: ACM conference on computer and communications security, pp 89–98. 30 Oct–Nov 3 2006
25. Halevy D, Shamir A (2002) The lsd broadcast encryption scheme. In: CRYPTO, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 47–60. 18–22 Aug 2002
26. Hassen HH, Bouabdallah A, Bettahar H, Challal Y (2007) Key management for content access control in a hierarchy. Comput Netw 51(11):3197–3219
27. Hu L, Liu Z, Cheng X (2010) Efficient identity-based broadcast encryption without random oracles. JCP 5(3):331–336
28. Ibraimi L, Tang Q, Hartel P, Jonker W (2009) Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In: ISPEC, Lecture notes in computer science, vol 5451. Springer, Berlin, pp 1–12. 13–15 April 2009
29. Lin H, Cao Z, Liang X, Shao J (2008) Secure threshold multi authority attribute based encryption without a central authority. In: INDOCRYPT, Lecture notes in computer science, vol 5365. Springer, Berlin, pp 426–436
30. McLean J (1988) The algebra of security. In: IEEE symposium on security and privacy, pp 2–7. IEEE computer society, 18–21 April 1988
31. Miklau G, Suciu D (2003) Controlling access to published data using cryptography. In: 29th international conference on very large data, Bases, pp 898–909, Sep 2003
32. Miyaji A, Nakabayashi M, Takano S (2001) New explicit conditions of elliptic curve traces for fr-reduction. IEICE Trans Fundam E84-A(5):1234–1243
33. Oh S, Sandhu R, Zhang X (2006) An effective role administration model using organization structure. ACM Trans Inf Syst Secur 9(2):113–137
34. Oh S, Sandhu R (2002) A model for role administration using organization structure. SACMAT, pp155–162
35. Sandh R, Bhamidipat V, Munawer Q (1999) The arbac97 model for role-based administration of roles. ACM Trans Inf Syst Secur 2(1):105–135
36. Sandhu RS, Coyne EJ, Feinstein HL, Youman CE (1996) Role-based access control models. IEEE Comput 29(2):38–47
37. Sandhu R, Ferraiolo D, Kuhn R (2000) The nist model for role-based access control: towards a unified standard. In: ACM workshop on role-based access control, RBAC00, pp 47–63

38. Sandhu R, Munawer Q (1999) The arbac99 model for administration of roles. In: Computer security applications conference, (ACSAC'99) proceedings. 15th annual, pp 229–238
39. Shamir A (1984) Identity-based cryptosystems and signature schemes. In: CRYPTO, Lecture notes in computer science, vol 196. Springer, Berlin, pp 47–53
40. Zhou L, Varadharajan V, Hitchens M (October 2011) Enforcing role-based access control for secure data storage in the cloud. Comput J 54(13):1675–1687
41. Zhou L, Varadharajan V, Hitchens M (2012) Trusted administration of large-scale cryptographic role-based access control systems. In: TrustCom, pp 714–721. 25–27 June 2012
42. Zhou L, Varadharajan V, Michael H (2011) A flexible cryptographic approach to secure data storage in the the cloud using role based access control. Int J Cloud Comput
43. Zhu Y, Hongxin H, Ahn GJ, Wang HX, Wang SB (2011) Provably secure role-based encryption with revocation mechanism. J Comput Sci Technol 26(4):697–710

Accountability-Based Compliance Control of Collaborative Business Processes in Cloud Systems

Jinhui Yao, Shiping Chen and David Levy

1 Introduction

The correctness of the inter-organizational collaboration in the Cloud relies on the individual correctness of all participants. That is, if the collaborator is compliant to the pre-defined business process, or Service Level Agreement (SLA). It follows that, the viability of this paradigm and the willingness of new participators to join collaboration highly depend on the trustworthiness of the behaviors of all collaborators. Here we adopt the definition of trustworthiness on IETF [25]: a trustworthy system is a system that is already trusted, and continues to warrants that trust because the system's behaviors can be validated in some convincing way.

It is a challenging task to preserve trustworthiness in such a dynamic cross-domain environment, as each participator is usually an independent entity with his own priorities and interests. Given that admission to violations may lead to penalties in some form, it is conceivable that they may intend to deceive and hide this fact. Therefore, a mechanism to detect and prove incompliance is needed for this computing paradigm to prosper.

Cloud computing promotes the practice of running services in a platform located outside of the owner's administrative domain. There are many ways things can go wrong for these services. Being able to identify which party is responsible when problems occur is rather important for cloud computing. In a sense, the Cloud is a computing environment to execute business processes. If service providers can

J. Yao (✉) · S. Chen
Information Engineering Laboratory, CSIRO ICT Centre, Canberra, Australia
e-mail: jinhui.yao@gmail.com

S. Chen
e-mail: Shiping.Chen@csiro.au

J. Yao and D. Levy
School of Electrical and Information Engineeering, University of Sydney, Sydney, Australia
e-mail: dlevy@ee.usyd.edu.au

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_12,

behave willy-nilly during the execution and will not be penalized for their faults, the Cloud will become a chaotic environment where business processes may lead to random outcomes. This poses a great threat to the cloud security as the participants as well as the clients of such business processes, that is, the users of the Cloud may bear loss (i.e. financial or other forms) for unknown reasons.

As a solution, we have proposed to enforce strong accountability to enhance the trustworthiness [30, 32]. While this will be elaborated shortly in later sections, in short, accountability provides a means to verify compliance according to evidence in a provable and undeniable way.

In this chapter, we will discuss the concept of strong accountability with a brief overview of the current literature and our own insights. We will describe the overall architecture of using Accountability Service (AS) to aggregate evidence and verify compliance in the Cloud. Then we elaborate a novel model to represent the $horizontal$ and $vertical$ structures of the collaborative business processes. Using this model, we classify two types of compliance and determine the loggings needed for their verification. We detailedly analyse and reason about the extent to which those compliance types can be verified in a provable way, and how accountability services can conclude their *compliance interval* through $probing$ and $voting$. Then we evaluate the practical effectiveness of the model and methodology proposed by implementing them in a collaborative business process. We will present our findings in the simulation experiments and discuss their implications.

This chapter is structured as follows. Section 2 shows a literature review over the research field. Section 3 presents the motivating scenario and our understanding of accountability. Section 4 describes the modelling developed for collaborative business processes. In Sect. 5, we define two types of compliance and analyse their verification approaches. Section 6 elaborates the approach AS nodes conclude *compliance intervals*. Section 7 presents the results of our evaluations and experiments and Sect. 8 concludes the book chapter with a discussion about future research directions.

2 Background and Key Concepts

Accountability, aims to make all the entities accountable for their activities. this implies that the compliance level of that entity to certain regulations must be known. Different system settings require different approaches to verify the compliance of the involved entities. The verification process involves evidence analysis or/and system diagnosis. Depending on the complexity of the circumstances, compliance verification may be quite trivial in some cases and be extremely difficult in others. In this section, we will study some approaches to diagnose the system's compliance violations and analyze the accountability evidence. Often, accountability or compliance frameworks rely on techniques in other disciplines to conduct the verifications (e.g. data mining). Some of those techniques will also be briefly discussed in this section, as they are orthogonal to our study.

2.1 Compliance Diagnosis Model

The goal of compliance checking is to discover in an efficient way which part of the system is causing this incompliance. SOA often involve multiple service nodes, whereby different nodes may not even be aware of each other during the operation. This makes the issue far more complex than discovering the faulty components locally in a single service entity. The suitable diagnosis model for a system is to large extent determined by the topology and architecture of the system. Broadly, they can be classified into two categories: local diagnosis model and distributed (remote) diagnosis model.

Local diagnosis models are suitable for diagnosing faults within a single service entity. For example, consider a traditional web server hosted by a commercial company. When any exception is noticed, the company will examine the logs to pinpoint the faulty components. Since both the servers and the logs are available locally, this type of diagnosis is mainly based on the analysis of the system logs or debugging techniques.

Distributed diagnosis model, on the other hand, applies to the scenario where multiple service entities from different parties are involved in the system. These service parties, as well as the recorded accountability evidence, are remotely distributed. Before analyzing the evidence, those evidence will need to be collected from the service nodes.

In this section, we focus on the *distributed diagnosis model*, as it is more closely related to our study of accountability. Distributed diagnosis model requires certain entity (entities) to be nominated as diagnosor, which will gather the evidence to conduct the diagnosis. In a general sense, a system can assign a central diagnosor, or assign a number of peer diagnosors.

In the case of a *central diagnosor*, one special entity will diagnose the entire system on behalf of other service entities. Accountability evidence will be centrally aggregated to be analyzed by the diagnosor. This model can be found in many literatures, such as the PlanetFlow project [13]. But the concerns of this approach are obvious: the evidence stored in the service nodes may eventually grow to unacceptable size, collecting them and processing them centrally will have performance and scalability issues. Heuristic designs will be needed to minimize the costs.

The LLAMA project [16, 17, 36] is a framework developed to diagnose system compliance with less evidence to be collected. Their approach is based on collecting the evidence only from the most likely root cause locations for the detected system faults, in order to save diagnosis time and costs. This is achieved by transforming the service node in service network to random variable nodes in Bayesian Network. Bayesian Network (BN) [14] is a directed, acyclic graph model for reasoning under uncertainty. Given the fault, BN can be used to compute, for every service node, that the probability of being the cause of that fault.

In the case of *peer diagnosors*, all the peers in the system are responsible for diagnosing the system for any potential or detected violations or faults. CATS and PeerReview are examples of such systems, where the diagnostic tasks have been

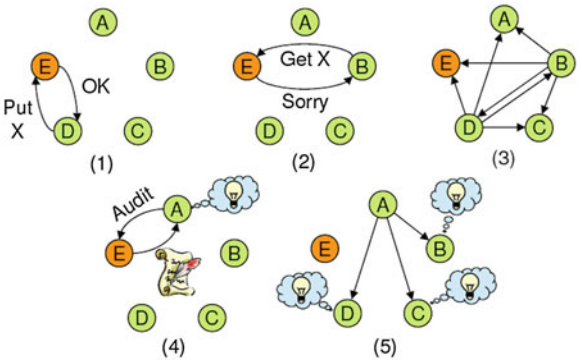


Fig. 1 Diagnosis by peers

spread over all the involved service nodes in the system. Thus, the communication and computing needs for diagnostic tasks of every node are much lower than for the central diagnosor, even though the overall cost might have increased.

Haeberlen et al. [10] consider the possible faults or incompliance in the system as Byzantine faults [15], i.e. the faulty or incompliant entity may exhibit arbitrary behaviors, such as corrupting local state or sending arbitrary messages. To detect those faults, every node will be associated with a number of witnesses who will send audit requests to collect and analyze its local evidence for diagnosing. Figure 1 describes a diagnosis scenario, where node E stores an object for client D (1) and then tries to hide it from client B (2). The diagnosis procedure is designed to make sure eventually, a peer node will discover and prove this incompliance, and then broadcast its findings.

As we mentioned, for peer diagnosis approaches, the number of the witnesses assigned to each service node is critical, for it is directly relate to the overall computation and transmission overhead introduced. In traditional study of Byzantine Faults, it is found that at least 3f+1 service replications are needed to tolerate f concurrent Byzantine faults [3]. However, the author found that in order to find the incompliant node, it only requires f+1 nodes, implying that every node needs to be assigned f+1 witnesses. This is due to the fact that, the last healthy node can always use the non-repudiable evidence to prove another node's incompliance to other entities. A detailed probabilistic analysis can be found in their techical report [11].

2.2 Evidence Analysis and Reasoning

Once the required evidence has been collected by the accountability authority (e.g. a diagnosor or a witness), it needs to be extensively analyzed to draw conclusions about the service node's compliance to respective service assurance. Sometimes, the conclusion is not directly drawn from the evidence collected; instead, it is drawn

from certain reasoning based on the evidence. According their different focuses, compliance analysis can be generally classified into two categories: performance compliance analysis and procedural compliance analysis. Below we will elaborate the two categories with their related research studies respectively.

2.2.1 Performance Compliance Analysis

Performance compliance analysis focuses on the verification of the performance aspects of the SLAs, such as transmission delays, availability, etc. One may consider such analysis to be trivial, by presuming it only involves simple arithmatic operations. However, the service quality from the client's perspective may be affected by a range of factors, some of which may not be under the service provider's control and responsibility. For instance, network traffic jam may make the response time longer, even though the reply message is actually sent in time.

Research studies conducted in this area identify the challenges of monitoring the performance compliance when high level of accuracy is required [2, 22, 27]. The measurement taken could be too heavyweight for practical implementation, or too lightweight to yield accurate results. Sommers et al. [26, 28] have proposed to use various mathematical methods to estimate the service provider performance based on the limited evidence collected (measured). In their experiments, applying adequate estimation techniques can only reduce the error of the resultant calculated performance, but not eliminate it.

Mathematical modeling and estimation techniques are out of the scope of this review. The point we should take from that is, calculating the service entity performance, according to the evidence logged by a monitoring entity, can be very difficult if extreme accuracy is essential. This fact must be noted when defining the SLAs and designing the analysis methods.

2.2.2 Procedural Compliance Analysis

Procedural compliance refers to the correct execution of the activities. A compliant service entity should only execute the activities that are defined in the workflow and execute them in the correct order. Verification of such compliance generally relies on the workflow modeling languages and the activity event traces generated during the execution. One example of such modeling languages is BPEL. The workflow can be expressed in BPEL in terms of different types of activities, and the event traces emitted by the workflow engine can be used to restructure the sequence of the activities executed [18, 24]. The consistency between the two shows the compliance.

Aalst et al. [29] attempted to utilize Petri net to analyze the procedural compliance (referred to as the conformance by the authors). Petri net [19] is a mathematical modeling language suitable for describing processes. The authors chose Petri net, as it is simpler to check the conformance of a formal model than a complicated language like BPEL.

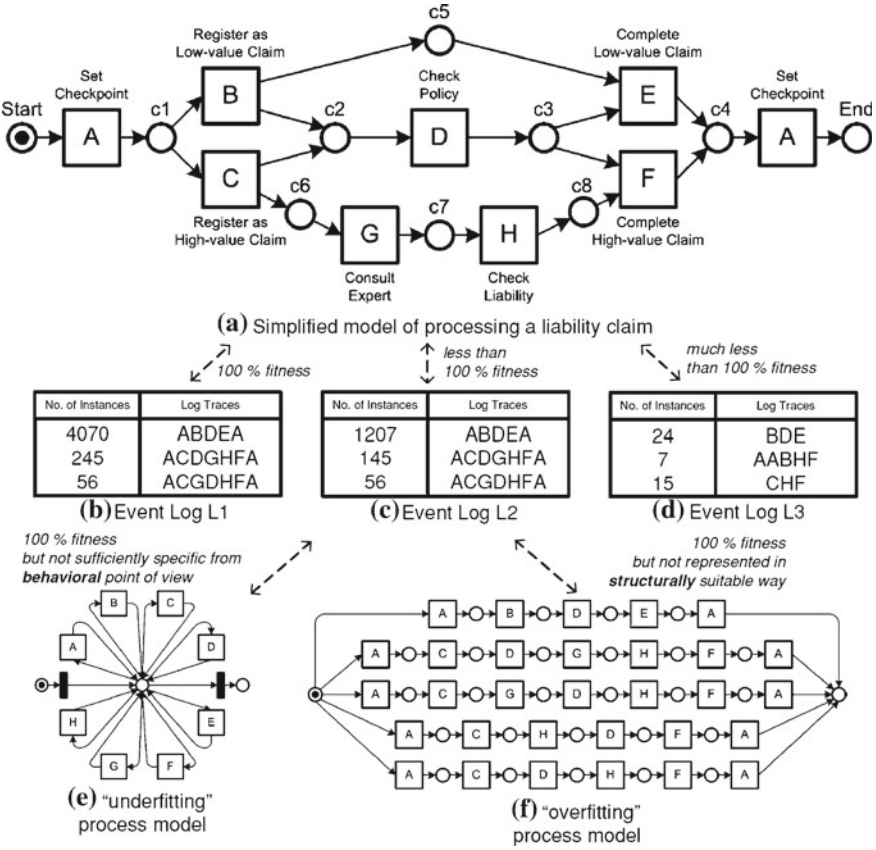


Fig. 2 Conformance checking through Petri-Net

Figure 2 presents a liability claim process translated into Petri net from BPEL (a). The authors identified two dimensions of conformance: fitness, whether the execution complies with the process model; and appropriateness, whether the model describes the process in a suitable way.

The fitness is verified by examining the log traces shown in Fig. 2b, c, and d. In the examples, only the log traces in (b) demonstrate the correct execution. Appropriateness can be measured according to the model's size and generality. Two extreme cases are shown in the example, with (e) being small but too general and (f) being very specific but too large. Details about the checking algorithms can be found in [23].

Our work, on the other hand considers a more hostile environment where all service entities are expected to behave in any possible manner and deceive for their own benefit. Approaches like [6, 34] share this point with us that cryptographic techniques are employed to achieve provability. Haeberlen [8] promotes accountability as one important aspect of cloud environment and [9] proposed to incorporate accountability mechanisms into the virtual machines for evidence recording.

There are many existing modeling of business process, popular ones like BPEL and XPDL.[1] They are designed solely to describe or define the action sequence of the process. Whereas our approach aims to capture the trust relationships and deployment details for compliance analysis and reasoning. In fact, it is common that approaches for compliance analysis will develop a model based on similar concept to BPEL and XPDL. Approaches like [5, 29] model the process as a sequence of event traces emitted by the service nodes. Compliance is verified through matching the patterns of the event or mine the event traces [1]. Approaches like [4, 10], model such execution as a sequence of service state changes, with the assumption that states are all preserved, compliance is verified by examining the causality between the states. In a similar way, Petri-net [19] has been used to model the actions and the state changes in a process [7, 21]. Our modeling method differs from those approaches in the way that we model the different domains and implementation infrastructures, and explicitly define the difference between the actions happened and the actions observed by the AS node.

Inference and reasoning involved in compliance assurance mostly focus on verifying the logical consistency and causality of the events. As in [12, 31], correctness of certain action is proved by looking up previous actions to check if the actor has been properly authorized. In our approach, the AS node first reason about the credibility of the events observed, then analyse the extent to which the compliance can be proved by this observation, and engage in a $probing$ and $voting$ process if needed.

3 Service Compliance: a Motivating Scenario

We use a one-stop loan application service as the running example in this book chapter. As shown in Fig. 3, the process requires the collaboration of five entities. First, a one-stop loan application service allows customers to lodge the application and fill in their personal information. His/her personal information will first be used to obtain a credit score from the credit rating authority, and then the score is attached

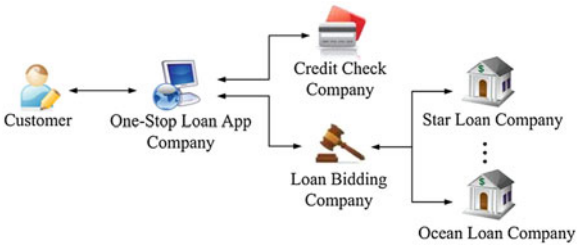


Fig. 3 Online one-stop loan application service composition

[1] XML Process Definition Language (XPDL) http://www.wfmc.org/xpdl.html

with other personal information to be sent to the loan bidding company. The bidding company forwards the application to multiple loan companies (Star Loan and Ocean Loan), and select the cheapest offer available (if any) to return to the applicant.

In this typical collaboration scenario, the overall correctness of the system depends on the correctness of all individual participants. As every of them may be interested to violate the collaboration rules for their own benefit or/and deceive to avoid possible penalties, the causer of a failure may be extremely difficult to determine. It follows that, a mechanism is required for the Cloud to prevent this denial of failure. This mechanism or service is essential to control the correctness of a business process established through collaboration. It helps the Cloud to discover untrustworthy or low quality service providers and protects the potential clients or collaborators from them.

Accountability is a concept to make the system accountable and trustworthy by binding each activity to the identity of its actor [35]. Such binding should be achieved under the circumstance that all actors within the system are semi-trusted. That is, each identified actor may lie according to their own interest. Therefore the bindings must be supported by provable or non-disputable evidence. In our approach, accountability can be incorporated into activity based process by requiring the actor (conductor) of the process to log non-disputable evidence about the activities in a separate domain from the domain of its own. The logging operations require the employment of PKI in all involved service entities. They are as follows:

1. The logger—A signs the evidence E with his private key K_{a-} to create a digital signature of the evidence $\langle E \rangle_a$.
2. The evidence and its signature are then logged in a separate entity—B.
3. When received, B signs $\langle E \rangle_a$ with his private key to create a receipt $\langle \langle E \rangle_a \rangle_b$.
4. Lastly, the receipt is sent back to the logger in the reply.

Proposition 1 *Without colluding (this will be relaxed later), after the evidence logging both entity—A and entity—B can independently prove the fact that, evidence E has been produced by A at time t.*

Proof Without colluding, we assume the digital signature is un-forgeable. B can use $\langle E \rangle_a$ to prove A produced E at time t. A can use $\left\langle \langle E \rangle_a \right\rangle_b$ to prove a separate entity—B has accepted $\langle E \rangle_a$ in the past, which is the evidence produced by A at time t.

To record evidence for the business process, we propose to use dedicated accountability services (*AS*) to enforce accountability on all the participating business services (*BS*), as shown in Fig. 4. The service space in the Cloud has been split into two domains: the Accountability Service Domain (*ASD*) and the Business Service Domain (*BSD*). In the *BSD*, business services (*BS*) compose with each other to conduct complicated business processes. Each service in the *BSD* keeps a close association with the accountability services (*AS*) in *ASD* so as to ensure that the *BS* are held accountable. In this setting, the *AS* nodes continuously receive logs and analyse

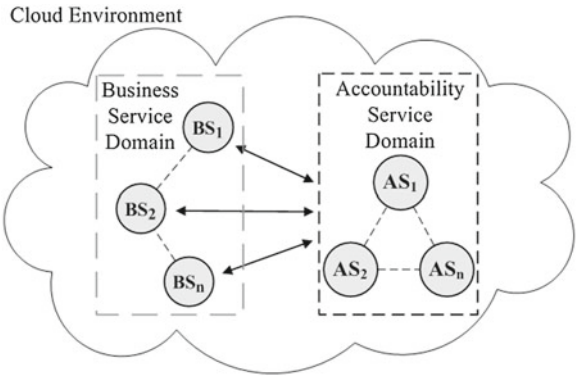


Fig. 4 Overall system design

the evidence to monitor the compliance of all the underlying participating services in the collaboration.

The accountability services can be provided by the Cloud as a mechanism to enforce strong accountability among all the collaborating web services; or they can be provided by other neutral service entities and deployed in the Cloud. We believe that, in order to create a trustworthy computing environment, the employment of such accountability mechanism is essential. With the accountability services, the collaborating service entities can take comfort from the knowledge that their partners will try their best to comply with the requirements they are bond to in the workflow, and hence establish the trust among them.

4 Modeling the Collaboration

A collaborative business process may involve many services provided by different entities (e.g. companies). To clearly describe the settings of a collaboration, one needs to look at both its *horizontal* and *vertical* structures. With respect to a participating service, in the *horizontal* structure of the collaboration, this service interacts with all other participating services according to the pre-defined business logic. Whereas in the *vertical* structure, this service may first, belong to a specific service entity (e.g. a company), and second, have its physical service node(s) deployed in an infrastructure provided by other entities (apart from this company). This *vertical* structure contains much information essential for verifying one's compliance. For instance, the actual service node may need to be contacted when verifying certain complex disputes. Our modelling intends to capture both the *horizontal* and *vertical* structure of the collaboration.

Different service providers collaborate with each other to form business processes. And small processes are integrated to form massive ones. We model the business process (P) formed through collaboration as a tuple

$$P = (N,\ P,\ V) \tag{1}$$

where N is the service node involved, P is the sub-process and V is the directed edge connecting them. The building block of a business process is the service nodes involved. The "node" here refers to the physical computing instance where the service application is deployed at. A service node is modelled as

$$N = (D_{in},\ D_{out},\ F,\ Spec) \tag{2}$$

where D_{in} and D_{out} are the input and output data of the service during execution (if there is any). $Spec$ is the specification about the computing instance that is hosting the service node. An instance can be a physical computer, part of a physical computer or several physical computers combined. $Spec$ intends to capture some information about the deployment of the service node so that it can be queried when problems occur. It contains information like the location of the node, the operating system, and its operation status. This brings certain transparency even if the $Spec$ only contains a virtual IP address which links to a computing instance that is constantly scaled and moved within the cloud environment. F refers to the function of this service node - the internal computational logic, which is a sequence of activities (A). Generally, we can further assume that the first action of a function (F) is to take in the input data (A^F_{in}) while the last action is to send away the output data (A^F_{out}), leaving the other actions between them as local computations within this function

$$F = \prod_{i=0}^{n} A^F_i = A^F_{in} * \prod_{i=1}^{n-1} A^F_i * A^F_{out} \tag{3}$$

The product symbol ($\prod$) and the multiply symbol ($*$) here refer to multiplications between different actions, which serve as a symbolic expression of the constitution of a function in terms of the actions involved. Due to the nature of the task, a function may include actions that are commutative or sometimes in parallel, for simplicity, we here only consider functions consists of linear compositions of actions and the multiplications in Eq. (3) is non-commutative. Another simplification we made here is that every service node is dedicated to only one function instead of many. These assumptions aim to keep our modelling adequately generic.

Every participator (e.g. company, organization) in the collaboration is regarded as service entity (En) which owns a group of service nodes, that is,

$$En = \{N_1,\ N \ldots N_3) \tag{4}$$

The inter-entity interactions within the business process are described through unidirectional edges (V). An edge could be connecting the business processes, service entities and functions. When the interaction occurs, one entity (at one end of the edge) will send specific data to another (the other end of the edge). The edge is modelled

as:

$$V = (Src, Dest, Cond, D_{out}) \tag{5}$$

Where Src and $Dest$ are the send and receiver in the interaction, $Cond$ is the condition to trigger this interaction (e.g. when output is available), usually expressed in terms of internal or global states, and D_{out} is specification of the data that will be transmitted during the interaction. The directed edges connecting services are denoted by arrows ($\longrightarrow$). For example, in a business process, node a (N_a) is defined to invoke node b (N_b) after finishing its own computation. This connection can be expressed as $N_a \longrightarrow N_b$. Alternatively, this can be expressed as a connection between their respective output activity and input activity: $A_{out}^{Na} \longrightarrow A_{in}^{Nb}$.

We can elaborate this model with our running example. For instance, the One-stop Loan App Company is our En_1, which has only one service node deployed in Amazon EC2. It can expressed as $En_1 = N_1$ and the node

$$N_1 = \{D_{Application}, D_{Result}, F_{Loan}, EC2 - 184.72.253.241\} \tag{6}$$

Similarly, let us suppose the Credit Check Company is our En_2 whose only service node is deployed in Microsoft Azure, it can be expressed as $En_2 = N_2$ and the node

$$N_2 = \{D_{Person}, D_{Rating}, F_{Credit}, Azure - 192.78.24.33\} \tag{7}$$

Here in the $Spec$ only the names of the cloud providers and the IP addresses are depicted for simplicity, certainly much more information about the service node can be shown. Suppose the two entities are in the different domain, the edge between them is then

$$...F_{Loan}^{N1} * A_{out}^{N1} \longrightarrow A_{in}^{N2} * F_{Credit}^{N2}... \tag{8}$$

Where A_{out}^{N1} and A_{in}^{N2} are the actions in the two services to send (output) and receive (input) data respectively. Note that the functions F_{Loan}^{N1} and F_{Credit}^{N2} also include output and input actions, they should not be confused with the actions of the service to interact with a partner. It is possible that a function of a service directly sends its output to a function in another service and therefore omitting A_{out}^{N1} and A_{in}^{N2} , in order to clearly and intuitively illustrate our modeling, we here assume the functions (F) are all encapsulated in the service which will manage and relay their input and output data.

A complete example is shown in Fig. 5. The vertical structure of a collaboration is presented in different views. Process view displays the horizontal structure of the collaboration. Service nodes from different service entities interact with each other to form the business process. The deployment structure of the services is shown in the infrastructure view. It displays the locations of the deployed computing instances in different computing provisioning clouds—Amazon Web Service and Windows Azure in our example. By capturing both horizontal and vertical aspects of the collaboration, many specific details can be taken in to consideration when analyzing the

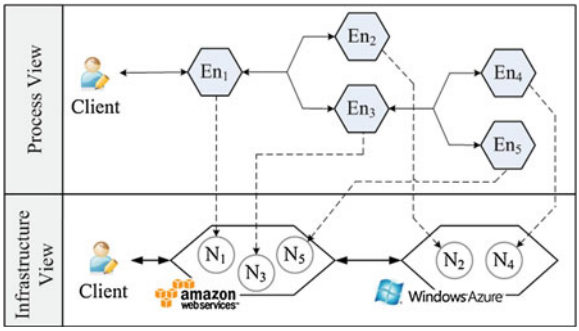


Fig. 5 Modelling the horizontal and vertical structure of collaboration

compliance of participants. For instance, the location of the deployment may assist the diagnosis process in which the actual computing instance can be probed to test its availability.

5 Applying for Service Compliance

Compliance is the correctness of the activities conducted by entities. The compliance of a certain function can be saved by logging evidence before the activity(s) and/or after them. However, as compliance requirements could have different forms, the evidence needed and the verification process can be quite apart. In order to systematically analyze compliance, one has to identify the different types of compliance and tackle them accordingly. Here we have classified compliance into two broad types, they are: *business logic compliance* and *QoS compliance*. For each of them, based on the collaboration model described in last section, we provide a formal definition and specify the logging method and the evidence content for their verification. Then we in detail analyze the method to prove the compliance of the service entities.

5.1 Compliance Type Definitions

Business Logic Compliance aims to verify if the conductance of the workflow or business process is correct, or more generally, if the interactions between the participating services coincide with the sequence or order defined in the business logic. It looks at the the horizontal structure or the process level and verifies the compliance of the service entities. Suppose we have a business process P consists of two entities - $Entity1$ and $Entity2$. $Entity1$ contains node a (N_a) with function F_x and $Entity2$ contains node b (N_b) with function F_y. For simplicity, we here use the two service nodes N_a and N_b to represent the two entities. Assuming the set of the local

and global states is S, the business logic compliance is defined as

Definition 1 Business logic compliance for node a ($Comp^{Na}_{logic}$) entails that, in a process P, given $V^{Na,Nb}$, if $\exists D^{Fx}_{out,t0}$ at time t_0 and $Cond \in S_{t0}$, there must $\exists A^{Na}_{out,t1}$ with $D^{Fx}_{out,t0}$ and $\exists A^{Nb}_{in,t2}$ with $D^{Fy}_{in,t2}$ where $D^{Fx}_{out,t0} = D^{Fy}_{in,t2}$ and $t_2 > t_1 > t_0$

Let us illustrate this definition through the expression below describing the business process only involving N_a and N_b. In this simple composition, the business logic is defined as that, N_a needs to send the output of F_x to N_b for further processing in F_y. To verify the compliance to this business logic, when the output of F_x is available, two actions need to be observed $A^{Na}_{out,t1}$ and $A^{Nb}_{in,t2}$, and if the data they are associated with are consistent, the defined business logic thus has been correctly carried out.

$$P : \left\{ ... * F^{Na}_{x} * A^{Na}_{out,t1} \right\} \longrightarrow \left\{ A^{Nb}_{in,t2} * F^{Nb}_{y} * ... \right\} \tag{9}$$

To verify business logic compliance, certain evidence needs to be recorded. According to the definition, first we need to know (1) when the output data ($D^{Fx}_{out,t0}$) is available; and (2) how the following two actions ($A^{Na}_{out,t1}$ and $A^{Nb}_{in,t2}$) are conducted. These can be addressed by requiring the services to log input and output actions (this include input/output actions of functions F). The evidence should contain sufficient information to describe the action that it is associated with. Its form could be various, to make a simple example, N_a can log signed evidence $E^{Na}_{logic,tx} = \langle processId,\ D^{Na}_{out,t1} \rangle_{Na}$ at time t_x for $A^{Na}_{out,t1}$; and N_b can log signed evidence $E^{Nb}_{logic,ty} = \langle processId,\ D^{Nb}_{in,t2} \rangle_{Nb}$ at time t_y. Time t_x and t_y should be within the allowed time frame (threshold time) after the occurrence of the corresponding actions. To elaborate, in our example, if the evidence is not received from N_b before the threshold time after the moment N_a claims it has invoked N_b, N_a will be suspected incompliant to the defined action to invoke N_b.

The notion of business logic compliance here just implies the conformance of services' activities to the defined business logic. The execution correctness of the functions of a service (e.g. if the credit score issued by the CreditCheck is reasonable) however, will be a totally different topic. Automatic verification of the local execution correctness often require domain specific solutions which fall out of the scope of this article. Nonetheless, the accountability mechanisms provide a means to record undeniable evidence associated with the actions conducted by services, which can be used by other tools to conduct such verification.

QoS Compliance focuses on the performance issues of the business process. It exploits the vertical structure of the business process and assesses the compliance of the computing instances in the infrastructure level. QoS requirements are usually expressed through service level agreements (SLA) by the service entities , which often define or provide reference to the metric functions (MF) for their verification and to the target computing instances. There are different types of QoS compliance include response time, throughput, availability, etc. The verification of them would

require certain performance data (PD) to be recorded during the operation, to be used by the verification MF. To give a generic definition of QoS compliance, we here take response time (RT) as our example subject, however the concept can be applied to other types of QoS compliance. Suppose a service entity - $Entity1$ contains node a (N_a)

Definition 2 QoS compliance on response time of N_a entails that, in a business process P, $Entity1$ guarantees that, given $T_{RT}^{Na} = \{t_1, t_2...t_n\} \in PD^{Na}$ it requires $MF_{RT}^{SLA}\left(T_{RT}^{Na}\right) \leq T_{RT}^{SLA}$

Expression (7) describes the internal operation of a service node N_a. Most intuitively, the response time of this node to the request from external invokers can be calculated through $T_{RT}^{Na} = t_1 - t_0$. It thus seems the logging strategy applied to *business logic compliance* is also valid for *QoS compliance*, the services need to log its input and output actions. But since the *QoS compliance* is verified according to the time elapsed between the two actions, the evidence for *QoS compliance* requires the declaration of the time when the action is carried out by the service entity. Developed based on the evidence structure for *business logic compliance* N_a can log signed evidence $E_{QoS,t0}^{Na} = \langle processId,\ D_{in,t0}^{Na},\ t_0\rangle_{Na}$ for $A_{in,t0}^{Na}$ and $E_{QoS,t1}^{Na} = \langle processId,\ D_{out,t1}^{Na},\ t_1\rangle_{Na}$ for $A_{out,t1}^{Na}$. Unlike for *business logic compliance*, for *QoS compliance* the evidence must be logged immediately after the occurrence of the corresponding actions. These extra stringencies are to help the AS nodes, as external third parties, to determine if the evidence and the time enclosed within the evidence logged by the concerned services are truthful.

$$\rightarrow \left\{A_{in,t0}^{Na} * F_x^{Na} * A_{out,t1}^{Na}\right\} \rightarrow \tag{10}$$

The truthfulness of evidence logged for $QoScompliance$ is much more difficult to verify compared to that of evidence for *business logic compliance*. A deceitful service may manifest the time in the evidence to reduce its response time observed by AS. To detect that, probability reasoning needs to be applied, to figure out how likely this evidence is true. In some cases, other service nodes also need to submit evidence to provide reference for the reasoning process. For instance, if the invoker of N_a logs the time when it invoked N_a, AS can use this reference to assess the credibility of the time N_a claims it receives the request invocation. Details about this will be discussed shortly in next section.

5.2 Compliance Analysis

Clearly, even though the participating service providers are instructed to submit evidence to demonstrate their compliance, it is highly likely they may (i) choose not to submit certain evidence or (ii) submit bogus evidence to avoid possible penalties.

In our design, we make very limited assumption on the logging and the truthfulness of the evidence submitted. Before a service node has been concluded for being compliant for a specific action, the AS node constant hypothetically presume it is incompliant. The confidence of such hypothesis—incompliance hypothesis confidence $Conf \in [0, 1]$ is a continuous value, with 1 being definitely incompliant and 0 being definitely compliant. It approaches 1 as more and more evidence suggest incompliance. The *AS* nodes continuously update *Conf* for all observed activities that are happening in the collaboration.

For *Business Logic Compliance*, to prove if node *a* has invoked the node *b*, *AS* node needs to receive the evidence logged by node *a* and node *b*. These evidence unambiguously prove if node *b* has been invoked with the data sent by node *a* , that is

$$E^{Na}_{logic,tx}, E^{Nb}_{logic,ty} \Rightarrow A^{Na}_{log,tx}, A^{Nb}_{log,ty} \Rightarrow \begin{matrix} A^{Na}_{out,t1}, A^{Nb}_{in,t2} \\ D^{Na}_{out,t1}, D^{Nb}_{in,t2} \end{matrix} \Rightarrow Comp^{Na}_{logic}$$

where '⇒' symbol refers to material implication. Based on this inference, we can define the condition for the confidence for business logic incompliance ($Conf^{Na}_{\neg logic}$) of N_a to be 1 and 0, that is

$$Conf^{Na,Nb}_{\neg logic} = \begin{cases} 0 & if\ \exists E^{Na}_{logic}, E^{Nb}_{logic} \bigwedge D^{Na}_{out,t1} = D^{Nb}_{in,t2} \\ 1 & if\ \exists E^{Na}_{logic}, E^{Nb}_{logic} \bigwedge D^{Na}_{out,t1} \neq D^{Nb}_{in,t2} \end{cases} \tag{11}$$

But when the required evidence is not present, we cannot conclude that the required action is not conducted, as there is always a chance that the communication channel is blocked. The most reasonable assumption *AS* can make is that, the longer time it waits for specific evidence to be received, it becomes more and more confident that the service failed to carry out the defined action, the more $Conf^{Na}_{\neg logic}$ approaches one. Following this, the value of $Conf^{Na}_{\neg logic}$ can be adjusted according to the deviation between the expected time to receive certain evidence and the current time. That is

$$Conf^{Na,Nb}_{\neg logic} = \begin{cases} f_{conf}\left\{t - \varepsilon\left(t_x\right) - \varepsilon\left(T_{Na,AS}\right)\right\}\ if\ \neg\exists E^{Na,out}_{logic} \\ f_{conf}\left\{t - \varepsilon\left(t_y\right) - \varepsilon\left(T_{Nb,AS}\right)\right\}\ if\ \neg\exists E^{Nb,in}_{logic} \end{cases} \tag{12}$$

where t is the current time and $\varepsilon\left(t_x\right)$ and $\varepsilon\left(t_y\right)$ are the expected time N_a and N_b should log (the threshold time), and $\varepsilon\left(T_{Na,AS}\right)$ and $\varepsilon\left(T_{Nb,AS}\right)$ are the expected transmission latency to the *AS* node. f_{conf} is a function to increase the confidence according to the unusual delay that has occurred.

The expected transmission time can be estimated according to the historical loggings (e.g. the time N_b logs can be estimated by adding the time N_a claims it invoked N_b to the average transmission latency between them). It is reasonable to assume a service provider to be interested to display both good processing latency and fast data transmission speed, in a bid to remain competitive. It will be unwise to systematically

log bogus data that yield a very slow transmission history so it can hide a real performance deterioration in the future, as its performance will appear unattractive and suspicious at the first place. In some cases, the expected transmission time can be estimated from available measuring of the internal and external transmission speed of the cloud, for instance [20].

QoS compliance is more difficult to be verified compared to the business logic compliance, because it has a bigger focus on the timing of the actions conducted. Nevertheless, the inference steps are similar to that of business logic compliance. Taking again our previous response time example, the steps of inference to verify N_a's response time compliance is

$$E^{Na,in}_{QoS,t0}, E^{Na,out}_{QoS,t1} \Rightarrow A^{Na}_{in,t0}, A^{Na}_{out,t1} \Rightarrow RT^{Na} = t_1 - t_0 \Rightarrow Comp^{Na}_{QoS}$$

However, the times t_0 and t_1 in the evidence are the claimed times by the issuer of the evidence N_a. The implications of this is that they may have been manifested by N_a to improve its QoS observed by AS. As an external party, it is difficult for AS to find out the true timing of the actions conducted internally in some services, since the network congestion may arbitrarily affect the transmission latency, this fact may be exploited by dishonest participators to hide incompliance. It follows that, certain means are needed to verify the truthfulness of the evidence effectively. We here describe an approach to assess the validity of the evidence based on two key aspects: individual evidence genuineness, and inter-evidence coherence.

Individual evidence genuineness refers to the validity of a piece of evidence on its own. AS nodes based on their knowledge about the current states of the system, evaluate the possibility that the evidence submitted is true. This possibility, which is essentially the confidence over the evidenceâs genuineness, can be evaluated as

$$Conf_{\neg Gen}(e) = f_{Gen}\left(e, S_{ref}\right) \tag{13}$$

where e is the evidence under assessment and S_{ref} is a set of local state parameters used as reference. Referring back to our response time scenario, AS node can record the time when the evidence is received and use it to determine the credibility of the claimed action time enclosed in the evidence. For example, if t_0 is provided in the evidence as the time N_a claims when $A^{Na}_{in,t0}$ happened, the evidence will reach the AS node at t^{AS}_0 where $t^{AS}_0 = t_0 + T^{Na,AS}$ with $T^{Na,AS}$ being the transmission latency between N_a and AS. Since the evidence is required to be logged immediately after the occurrence of the activity, the confidence of evidence $E^{Na,in}_{QoS,t0}$ being bogus is

$$\begin{aligned} Conf_{\neg Gen}\left(E^{Na,in}_{QoS,t0}\right) &= f_{Gen}\left(E^{Na,in}_{QoS,t0}, t^{AS}_0\right) \\ &= f_{conf}\left\{t_0 - \left(t^{AS}_0 - \varepsilon\left(T^{Na,AS}\right)\right)\right\} \end{aligned} \tag{14}$$

where $\varepsilon\left(T^{Na,AS}\right)$ is the expected (average) transmission latency between N_a and AS. In normal case, the value of $t_0 - \left(t_0^{AS} - \varepsilon\left(T^{Na,AS}\right)\right)$ should be relatively small. But if N_a attempts to reduce its response time deceptively by logging $t_0 + \triangle$ instead (so $t_2 - t_0 - \triangle \ll RT^{Na}$), this will result

$$t_0 + \triangle - \left(t_0^{AS} - \varepsilon\left(T^{Na,AS}\right)\right) \gg t_0 - \left(t_0^{AS} - \varepsilon\left(T^{Na,AS}\right)\right)$$

and yield a larger confidence value about t_0 being bogus. Whilst this is an example of applying the genuineness test, the concept can be applied to other scenarios.

Inter-evidence coherence, as its name suggests, focuses on the coherence among the different evidence submitted (possibly by different service nodes). More essentially, it checks if the actions depicted by difference evidence raise conflicts or suspicion. To verify if the action described in certain evidence is coherent to some other related actions conducted (possibly by other entities), the evidence about those actions need to be logged (if not already) and brought together for analysis to discover possible conflicts. This can be expressed as

$$Conf_{\neg Coh}(e) = f_{Coh}\left(e,\ e_1^{ref},\ e_2^{ref} \ldots e_n^{ref}\right) \tag{15}$$

where $e_1^{ref},\ e_2^{ref} \ldots e_n^{ref}$ are the relevant evidence used as reference, and f_{Coh} is a function to evaluate their coherence with each other.

Referring back to our response time scenario, dishonest service nodes can exploit the possible transmission congestion at its input/output edge. A dishonest node may log evidence a moment after it has started processing a job or log evidence before the output can be sent to the next node. In this way, they can reduce its processing time observed by AS yet passing the genuineness test as the moment of logging is changed accordingly, i.e. the bogus evidence $t_0 + \triangle$ arrives at AS at time $t_0^{AS} + \triangle$ resulting the same confidence value as when the evidence is genuine (Eq. 14).

This type of fraud can be detected by logging the input/output actions of the invoker of the service, say N_p, and the recipient of the output of N_a, say N_q, and analyse their coherence with the evidence logged by N_a. Let us expand expression 10 to include three nodes, N_p, N_a and N_q.

$$\{\ldots A_{out,tp}^{Np}\} \rightarrow \{A_{in,t0}^{Na} * F_x^{Na} * A_{out,t2}^{Na}\} \rightarrow \{A_{in,tq}^{Nq} \ldots\} \tag{16}$$

Here N_p and N_q are requested to log their input/output actions as well. In the normal case, AS will observe that N_p logs $E_{QoS,tp}^{Np,out}$ at t_p and N_a logs $E_{QoS,t0}^{Na,in}$ at t_0, resulting the transmission latency between N_p and N_a being $T^{Np,Na} = t_0 - t_p$. Assuming the history transmission latency measures are used to derive the expected transmission time $\varepsilon\left(T^{Np,Na}\right)$, if N_a delays its moment of logging to $t_0 + \triangle$, AS will immediate notice $t_0 + \triangle - t_p \gg \varepsilon\left(T^{Np,Na}\right)$ and suspect the truthfulness of $E_{QoS,tp}^{Np,out}$ and $E_{QoS,t0}^{Na,in}$ as they appear to be incoherent to each other and to the history

measurements. The coherence of $E^{Na,in}_{QoS,t0}$ is then evaluated as

$$Conf_{\neg Coh}\left(E^{Na,in}_{QoS,t0}\right) = f_{Coh}\left(E^{Na,in}_{QoS,t0}, E^{Np,out}_{QoS,tp}, \varepsilon\left(T^{Np,Na}\right)\right) = f_{conf}\left\{t_0 - t_p - \varepsilon\left(T^{Np,Na}\right)\right\} \tag{17}$$

The larger the deviation to the expected (usual) transmission latency between the two nodes, the more confident AS can be about the incoherence between the evidence. Same method can be applied to evaluate the coherence of $E^{Na,out}_{QoS,t2}$ with the evidence $E^{Nq,in}_{QoS,tq}$ logged by N_q.

Unlike genuineness test, although AS can detect incoherence in the evidence, it is difficult to find out which service is likely to be the party that logged bogus evidence to improve its QoS. In other words, even AS observes $t_0 + \triangle - t_p \gg \varepsilon\left(T^{Np,Na}\right)$, it cannot be sure if $\triangle$ is introduced by N_p or N_a as both of them have the incentive to deceive. This particular issue will be discussed in Sect. 5 when we introduce the concept of Compliance Interval.

Therefore, the confidence of QoS compliance $Conf_{\neg QoS}$ is computed by considering both the genuineness $Conf_{\neg Gen}$ and coherence $Conf_{\neg Coh}$ of the evidence submitted. This process can be expressed as

$$Conf_{\neg QoS} = \sum_{i=0}^{n} Conf_{\neg Gen}\left(e_i\right) + \sum_{i=0}^{n} Conf_{\neg Coh}\left(e_i\right) \tag{18}$$

where the confidence functions are assumed to be linear for simplicity. In our response time scenario, the response time of N_a can be calculated through $RT^{Na} = t_2 - t_0$, the confidence is developed as the accumulation of the confidence levels of the genuineness and coherence of the evidence. This confidence reflects how reliable the calculated response time is, the higher the non-compliance confidence, the less the QoS value calculated can be trusted.

The confidence function could have various forms. A simple example could be a linear function which increases when the deviation to the expected value increases, i.e. $f_{conf} = a(x - \varepsilon(x))$. More complex calculation methods should be adopted with domain knowledge when targeting some specific problems. One may notice the $Conf^{Na}_{\neg QoS}$ calculated according to Eq. (18) can be equal to or exceed 1. Certainly, the probability of an event can never exceed one hundred percent, in the practice, scaling may be applied to keep the value under one. The idea we are demonstrating through this simple accumulation equation is that, the individual confidence values calculated for different evidence, need to be aggregated to form the overall confidence value which is used to judge the QoS compliance.

6 Concluding Compliance Through Voting and Probing

Preserving system trustworthiness entails credible compliance verification. This approach implies that the verification process is conducted under the assumption that the underlying services are semi-trusted, whereby their behaviour is governed by their own interests or priorities. Intuitively, this assumption is valid for accountability services as well. So far, it has been assumed that AS nodes are provided by third parties, who do not benefit from the compliance performance of the underlying business services, and act as a neutral monitoring agent to maintain trustworthiness. However, this assumption is rarely true in practice. Although the monitoring agent may not have prior connections to the services being monitored, once the monitoring relationship is established, there is potential for colluding between the monitor and the monitoring subject which has to be considered when verifying accountability.

For example, the evidence logging protocol described in Sect. 3 is built on the premise that the private keys of the logger (*BS*) and the *AS* are kept confidential and are never disclosed to one other; thus, the evidence log can serve as a definitive proof of the action it is associated with. However, in a likely scenario that an *AS* colludes with a *BS* and obtains the private key assigned to the *BS*, the *AS* can literally manifest any evidence and sign on the *BS*'s behalf to improve its compliance performance. The presence of such collusion and corruption can significantly hinder the trustworthiness provided by the accountability service. It naturally follows that, instead of one, multiple accountability services must be employed to monitor the process compliance. Although colluding is still possible, the scenario whereby multiple *AS* nodes collude with the same *BS* is highly unlikely, and this likelihood diminishes further when the number of accountability services involved increases.

In this approach, several *AS* nodes may be monitoring the same *BS* node at the same time, which will determine its compliance according to their respective evidence received. A *voting* process will be conducted among different *AS* nodes, during which each *AS* node sends its own conclusion (called opinion) to the others. A *BS* node will be found compliant or non-compliant only when the majority of the *AS* nodes provide that opinion. Ideally, all *AS* nodes should have a consistent view upon the underlying *BS* nodes; however, in the event of exceptions (e.g. a particular *AS* node may not receive certain evidence due to network problems), the majority of the *AS* nodes can still verify its compliance or lack thereof. Such a *voting* method thus tolerates some scope for disputes and failures. Further, it makes verification process conducted in the *accountability service domain* more reliable as it helps to address the concerns that an *AS* node could also act erroneously, either mistakenly or deliberately.

Probing refers to the process whereby the states of the service nodes are actively tested when the confidence of the non-compliance hypothesis reaches a critical level (e.g. $Conf_{\neg logic} > Conf_{threshold}$). The *AS* nodes will, in such cases, send special probing messages (e.g. ping) to every concerned *BS* node in order to challenge their compliance. Consequently, in order to prove their proper conduct, the probed *BS* nodes need to respond to the probing messages in the correct manner and the majority opinions of the *AS* nodes after probing would give a conclusion about which node(s)

is faulty, and may be penalized or replaced. For example, if after probing, most of the *AS* nodes confirm that certain *BS* node is no longer active, this node is therefore found to be inactive despite of the fact that certain *AS* node may claim the *BS* node has correctly responded to its probing message.

Nonetheless, voting and probing to some extend rely on the unbiased judgment by the *AS* nodes. Whether an individual *AS* node is colluding with some *BS* nodes is very difficult to prove unless strong and sometimes unrealistic assumptions about the system are made a priori. Still, the potential for, and the effect of, such collusion are appreciably diminished when multiple *AS* nodes are involved as individual monitoring agents. While the inter-service colluding is not provable, the conclusion drawn from the voting and probing process is. Thus, the multiple *AS* nodes acting in tandem provide trustworthiness to the system.

6.1 Defining the Compliance Interval Through Probing and Voting

For *business logic compliance*, *probing* is usually required when certain evidence is not received from a node within the allowable time frame, leading to a high confidence value. A simple probing procedure is shown in Algorithm 1. When the confidence of non-compliance exceeds the threshold value, the *AS* node will send probing invitation to other *AS* nodes monitoring the suspicious service nodes, in conjunction with the supporting evidence. When the majority of the *AS* nodes believe that *probing* is needed, each *AS* node sends a probing message to the node and waits for its response. If the reply is not received within the required time frame, the node is considered inactive and therefore faulty. Alternatively, the reply message is received and it contains the evidence the node provides in order to resolve the matter, and thus establishes its compliance.

Given that, in response to the probing message, different evidence may be received by different probing *AS* nodes, a voting is subsequently conducted, whereby all involved *AS* nodes exchange their decisions (opinions on the matter) and evaluate the case. A simple voting process is described in Algorithm 3.2. When an opinion and its supporting evidence are received from another *AS* node, an assessment is carried out to evaluate its genuineness and coherence with other locally available evidence. The opinion will only be taken into consideration if the confidence values obtained from such assessment are adequately low. Finally, the conclusion about the compliance of the service nodes under suspicion is drawn according to the majority of the opinions collected. Moreover, all probing as well as voting messages are signed by the respective issuer before sending and their recipient will keep them for future reference. This log system makes the compliance verification process traceable and verifiable.

Inter-service invocation must involve two service nodes, say N_a and N_b, as discussed previously, both need to log evidence for AS to verify such invocation has been successfully carried out. If E_{logic}^{Nb} is not received, assuming the network infrastructure handles the transmission loss detection and retrying, then either (i) N_a did not

invoke N_b and therefore is non-compliant; or (ii) N_b did not respond to the invocation received and therefore is non-compliant. As we have established, non-compliant services may lie in order to avoid penalties. Probing may find both of the service nodes claiming compliant. In this case, such event will be concluded as $conflicting$, which means one of the services must be non-compliant yet both claiming compliant. It is difficult for AS to determine which party is more likely to be deceitful unless assisted with domain knowledge or by making further assumptions.

It follows that, a fair and provable way for AS to express one's business logic compliance is through a *Compliance Interval*. The business logic compliance interval for a service node a (CI^{Na}_{logic}) is defined as

$$CI^{Na}_{logic} = \left[\frac{|A^{Na}| - |A^{Na}_{\neg logic}| - |A^{Na}_{conflict}|}{|A^{Na}|}, \frac{|A^{Na}| - |A^{Na}_{\neg logic}|}{|A^{Na}|} \right] \quad (19)$$

where $|A^{Na}|$ is the cardinality of the set of all activities conducted by Na observed by AS, $|A^{Na}_{\neg logic}|$ is the cardinality of its set of activities found non-compliant with business logic, and $|A^{Na}_{conflict}|$ is the cardinality of its set of activities found conflicting by AS. The lower bound is determined by the percentage of the compliant actions if all conflicting actions are assumed non-compliant, whereas the upper bound is determined by the percentage of the compliant actions if all conflicting actions are assumed compliant. Thus, he closer those bounds get to one, the more compliant the service is. For example, if an accountability service witnessed 100 actions conducted by a service within a business process, of which 20 actions were definitely non-compliant, 20 actions are conflicting (possibly non-compliant) and 60 are definitely compliant, then, according to Eq. (17), the compliance interval will be [60, 80 %].

Algorithm 1: Probing suspicious service nodes

Input: a set of AS nodes $\{AS\}$, two suspicious nodes N_a, N_b
Output: Probing results Res_{Probe}

Initialize the set of AS replies $AS_{rep} \longleftarrow \{\}$;

if $Conf^{Na}_{\neg logic} > Conf^{Threshold}_{\neg logic}$ **then**
  foreach $N_{AS} \in \{AS\}$ **do**
    send probing invitation and relevant evidence to N_{AS};
    receive reply from N_{AS}: Rep_{Nas} with evidence;
    $AS_{res} \longleftarrow Rep_{Nas}$;
  end
  if *No. positive replies*$> |\{AS\}|/2$ **then**
    probe $N_a, N_b \Rightarrow Res_{Probe}$;
    return Res_{Probe};
  else
    return null;
  end
end

Although the compliance interval explicitly expresses the compliance performance of the services being monitored, without any attempt to solve the possible disputes between the services (which can be problematic), it is still useful indicator of a service's compliance and trustworthiness. For example, a service that consistently claims that others are responsible for its own faults will result a large set of conflicting actions, i.e. $A^{Na}_{conflict}$, and lead to a large compliance interval, thus strongly suggesting its possible lack of trustworthiness. In general, despite all the domain-related facts, the larger this interval is, the less trustworthy the service node appears to be. In other words, when compared to a service with compliance interval [60, 80 %], another service with compliance interval [75, 80 %] clearly seems to be more reliable and trustworthy even though they might actually have the same true compliance percentage.

In terms of establishing *QoS compliance*, *probing* is usually not needed or ineffective. A service entity may forge seemingly unreal evidence, resulting high $Conf_{QoS}$, but probing the service can reveal little truth if the entity is committed to deceive. Therefore, for QoS data logged with high $Conf_{\neg QoS}$ ($Conf_{\neg QoS} > Conf^{Threshold}$), AS nodes should propose the likely QoS value according to their observations, i.e. adjust the QoS value claimed by the service based on the knowledge obtained from the genuineness and coherence tests. Once the confidence value exceeds the threshold, AS should propose an adjustment $\triangle^{AS}$ to the evidence received 'e' so that

$$Conf_{\neg Gen}\left(e+\triangle^{AS}\right) < Conf^{Threshold}_{\neg Gen} < Conf_{\neg Gen}(e) \quad and/or$$
$$Conf_{\neg Coh}\left(e+\triangle^{AS}\right) < Conf^{Threshold}_{\neg Coh} < Conf_{\neg Coh}(e)$$

where $Conf^{Threshold}_{\neg Gen}$ and $Conf^{Threshold}_{\neg Coh}$ are the threshold confidence for evidence genuineness and coherence respectively.

Algorithm 2: Voting among AS nodes for verifying compliance

Input: a set of AS nodes $\{AS\}$, two suspicious nodes N_a, N_b
Output: compliance conclusion $Comp$

Initialize the set of AS votes $votes \longleftarrow \{\}$;
Analyse $Res_{Probe} \Rightarrow Opinion$

foreach $N_{AS} \in \{AS\}$ **do**
  send signed $Opinion$ and supporting evidence to N_{AS};
  receive N_{AS} opinion $Opinion_{Nas}$ with evidence E_{Nas};
  if $Conf_{Gen}(E_{Nas}) < Conf^{Threshold}_{Gen}$ *and* $Conf_{Coh}(E_{Nas}) < Conf^{Threshold}_{Coh}$ **then**
    $votes \longleftarrow Opinion_{Nas}$
  end
end
majority of $votes \Rightarrow Comp$

Once again, the response time QoS (RT) will serve as an example scenario. Response time compliance is usually determined by its average deviation to the SLA (T_{SLA}). Hence the average value of the claimed response times form the lower bound of the interval, whilst the average value of the adjusted response times (i.e. some claimed response times would have been adjusted by AS nodes) form the upper bound of the interval. The compliance interval about the response time of N_a observed by an AS node (AS_i) is given by:

$$CI_{Qos}^{Na,ASi} = \left[\frac{\sum_{k=0}^{n} RT_k^{claim}}{n * T_{SLA}} , \frac{\sum_{k=0}^{p} RT_k^{accept} + \sum_{k=0}^{q} RT_k^{adjust}}{n * T_{SLA}} \right] \quad (20)$$

where RT^{claim} is the response time claimed by the service, RT^{accept} is the response time found plausible by AS_i and RT^{adjust} is the adjusted response time based on the suspicious evidence received. The lower bound is determined by the ratio of the average value of the response time samples claimed by the service to the response time SLA. Similarly, the upper bound is equivalent to the ratio of the average value of the (partially) adjusted response time samples to the SLA. Therefore, the smaller the upper and lower bounds are the better performance the service has. In contrast to the compliance interval of business logic compliance, where the upper and lower bounds must be lower than one (100 %), those associated with the compliance interval for QoS compliance may be equal or greater than one. For example, 80 % will suggest that the service response time is 20 % faster than the SLA it guarantees, whereas 120 % implies 20 % slower performance and certain penalty should be applied.

The QoS compliance intervals established by different AS nodes may not be the same due to the randomness introduced by the network environment. The average QoS compliance interval of a service is determined through a voting among the involved AS nodes. Each AS node forwards the compliance interval it has calculated to all other AS nodes, and when the intervals established by others are received, they are aggregated to derive the average lower and upper bounds for the average compliance interval, which is given by:

$$CI_{Qos}^{Na} = \left[\frac{\sum_{i=0}^{n} B_{lower}^{ASi}}{n} , \frac{\sum_{i=0}^{n} B_{upper}^{ASi}}{n} \right] \quad (21)$$

Where B_{lower}^{ASi} is the lower bound of $CI_{Qos}^{Na,ASi}$ and is B_{upper}^{ASi} the upper bound of $CI_{Qos}^{Na,ASi}$. In other words, the average value of the compliance intervals computed by all AS nodes involved is used as the overall compliance interval of the BS nodes.

6.2 Colluding in the Accountability Service Domain

Possible violations incurred by any BS nodes as well as AS nodes can be regarded as Byzantine faults [15]. The faulty node under this category may exhibit arbitrary behaviours, such as being non-responsive or sending faulty messages. Early study of Byzantine faults [10, 11] has found that for the diagnostic system to find the non-compliant node, a system requires $f + 1$ nodes, where f is the number of possible concurrent Byzantine faults. This is the case when solid evidence is available at that last healthy node, which can unarguably prove the violations of the others. However, when conspiracy is involved, the conspiring AS node may be able to forge fake yet seemingly genuine evidence, as a bid to prevent certain BS nodes from being penalized.

Certainly, it is not likely an AS node will be able to forge evidence to claim something the conspiring BS node has not done. For instance, to claim BS_1 has sent a message to BS_2 (which it has not), the AS node will need the private key of BS_2 in order to forge the log submitted by BS_2 after receiving the message. But in some other cases, when the private key of the conspiring BS node is all what it needs, an AS node will be able to generate seemingly undeniable evidence. For example, an AS node can always claim a BS node is not inactive, by showing the forged signed response from the BS node for its probing message.

Figure 6 shows an example of a probing process. In the example, three BS nodes are composed to form a small business process, whereas three AS nodes have been assigned to monitor them. In Fig. 6a, a possible fault is noticed when BS_2 failed to send its output to BS_3 in the required time frame. All three AS nodes then send probing message to BS_2, and find out the node is overloaded by the requests from BS_1 (Fig. 6b). In Fig. 6c, disregarding the votes of the other AS nodes, AS_1 forges evidence to claim to BS_1 that BS_2 is working fine, so as to gain more job requests for BS_2 (although they will be processed slowly) while AS_2 tells the truth. At this moment BS_1 is not able to decide if it should forward more job requests to BS_2. Finally, in Fig. 6d, AS_3 also notifies BS_1 of BS_2 being faulty. BS_1 is thus convinced and send job requests to alternative BS nodes.

Proposition 2 *In the worst case, where AS nodes may collude for the benefit of some services by forging seemingly genuine evidence, the minimum number of AS nodes (N_{AS}) required is $N_{AS} \geq 2f_{AS} + 1$, where f_{AS} is the number of possible concurrent colluding AS nodes, so that the system can behave justly.*

Proof According to Algorithm 1, the $voting$ result is determined by the majority of the opinions of AS nodes. Suppose $N_{AS} < 2f_{AS} + 1$, then $f_{AS} \geq N_{AS}/2$, which means the votes of healthy AS nodes will never be the majority.

In the practise, it may not be easy to determine N_{AS} or the approach to choose the AS nodes with the smallest chance of conspiracy. Domain experts may be needed

to make such decisions. An alternative approach may be looking at the historical performance of the system and apply adjustments which eventually tune the system to its proper settings. In particular, one may be interested to study the cases when *AS* nodes make conflicting conclusions. In those cases, the ratio between the number of correct conclusions to the number of incorrect ones, indicates the overall correctness of the system and the degree of dominance of the healthy *AS* nodes.

7 Simulation Studies Through Fault Injection Experiments

We have implemented an experimental loan application business process in Amazon EC2—a computing resource provisioning service that charges the user according to the CPU usage. As shown in our running example, we deployed 5 *BS* nodes and 1 *AS* node on six standard computing stances in EC2—these are virtual machines with computing power equivalent to 1GHz CPU and 1.7GB memory. Our implementation used Apache Tomcat 5.5 as our Servlet container, and Axis2 1.5 as our web service engine. We used BPEL to orchestrate the service nodes to form our loan application business process. Apache ODE (http://ode.apache.org) has been used to conduct the business process defined in BPEL.

7.1 Performance Evaluation

The BPEL scripts deployed in each service node have been transformed to log evidence at *AS* when a service node receives an input and when it invokes another

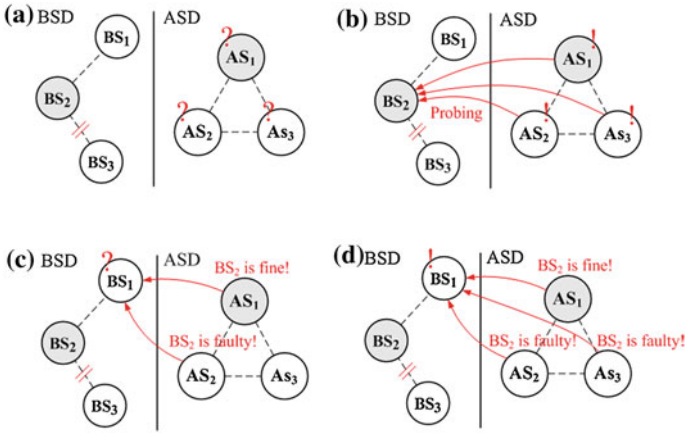


Fig. 6 Probing and voting with colluded *AS* node

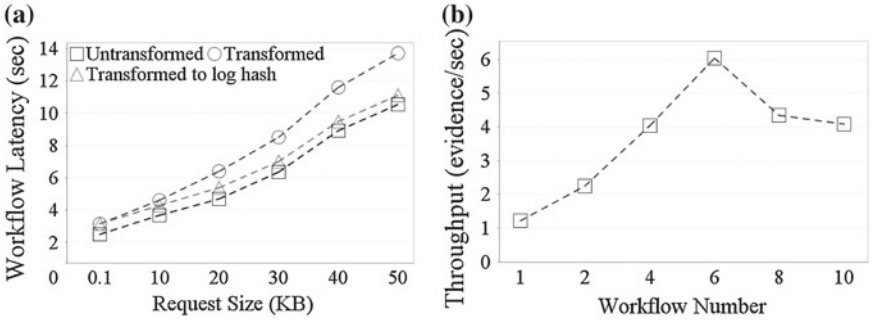


Fig. 7 Performance testing. **a** Latency introduced by incorporating accountability. **b** Throughput of AS under different loads

node with its output (refer to [33] for details). We first conduct testing to evaluate the latency introduced by incorporating accountability. Figure 7a shows the overall latency to finish the workflow with untransformed BPEL scripts and with transformed ones. For the process that logs the entire input/output messages (the serie marked with "circles"), the latency introduced compared to the untransformed one (the serie marked with "squares") grows as the request message becomes larger. In percentage terms, on average we observed a 30 % increase in the overall latency. Intuitively, this latency is substantial; however it can be largely improved through the use of collision-resistant hash functions. We can see in the graph, the extra latency is significantly reduced if the BPEL scripts are transformed only to log the hash of the evidence (the serie marked with "triangles"). In fact, the extra latency almost remains constant regardless of the size of the request message, so it becomes more and more negligible when the message size increases. If a collision-resistant hash function (e.g. VSH) is used to create digital certificates of evidence objects, the certificates are undeniably linked with the evidence and can be used later to request the real evidence objects from the loggers.

As aforementioned, one AS node may be monitoring multiple *BS* nodes, or multiple business processes. Naturally, it is interesting to find out the processing capability of individual *AS* nodes. To evaluate this, we replicated the business process we have implemented, and invoke multiple business processes concurrently. With this setting, we evaluate the processing throughput of an AS node when it is under different loads. We tested this with messages of size 50KB, the processing operations conducted by AS involves both business logic and QoS compliance validating without probing and voting. Figure 7b shows the testing results. In the figure we can see that, the processing throughput of the AS improves as the number of workflows increments, it reaches its peak when the AS is monitoring 6 workflows, and then it decays gradually if more workflows are involved. The peak value may be different if tested with different computing environment, in the real practise, compromises need to be made to balance the working load and the complexity of compliance monitoring logic.

Based on above observations, we believe it is reasonable to conclude that the latency introduced by incorporating accountability into existing business processes is acceptable and adjustable. An *AS* deployed in a small computing instance with limited resources in the cloud is capable of monitoring the compliance of a number of workflows each of which consists of multiple service nodes.

7.2 Fault Injection Experiments

In order to demonstrate the effectiveness of the system, we inject some faults into the business process we have implemented, and let the *AS* nodes conclude the upper and lower bounds of the services' compliance intervals CI. Each service of the 10 business processes implemented has been asigned (i) an error rate, which determines the probability the service node will be incompliant to the business logic; and (ii) the tendency to deceive, expressed as percentage to determine the probability the service will deny the incompliance caused by the internal error that has occurred. Three *AS* nodes are involved to conclude the CI of the services.

The experiment is divided into 30 rounds. In each round, one random business process structure is generated for every business process. Then all processes are executed for a hundred times and the compliance interval as well as the actual compliance percentage are recorded. The purpose of randomly restructuring the business processes in every round is to avoid the case that the services next to a deceitful service will be consistently involved into many conflicts even though they seldom lie. It is also to coincide with the fact that, in practise, a service is usually provided to a large group of clients rather than a small number of fixed clients.

In Fig. 8 we show the business logic compliance intervals concluded for two very representative services: S_1, a relatively honest and healthy service, with 10 % error rate and 10 % deceit rate (Fig. 8a); and S_2, a relatively deceitful and defective service, with 30 % error rate and 30 % deceit rate (Fig. 8b). In the two figures, the upper/lower bound curves are formed by the upper and lower bounds of the compliance intervals concluded in 30 rounds of the experiment. The actual compliance curves (the serie marked with "squares") represent the true compliance percentages (rates) of the services which are unknown to the *AS* nodes.

Interestingly, due to the different error rate and degree of honesty, the two figures exhibit distinctive characteristics. First of all, apart from the random fluctuations, the upper/lower bounds of S_1 are much higher than that of S_2, suggesting S_2 has a higher error rate. Secondly, the compliance intervals of S_1 (the distance between the upper and lower bound) are much smaller than that of S_2, which indicates S_2 is much more likely to deceive compared to S_1. Presumably, the real compliance rate of a deceitful service will be close to the lower bound of its interval, as it is responsible for most of the conflicts it has been involved with. Oppositely, an honest service's compliance rate should be close to its upper bound. Considering these, we can see the compliance intervals in the two figures can help to make very reasonable estimations about the true compliance rate of the services as well as their trustworthiness. Services with

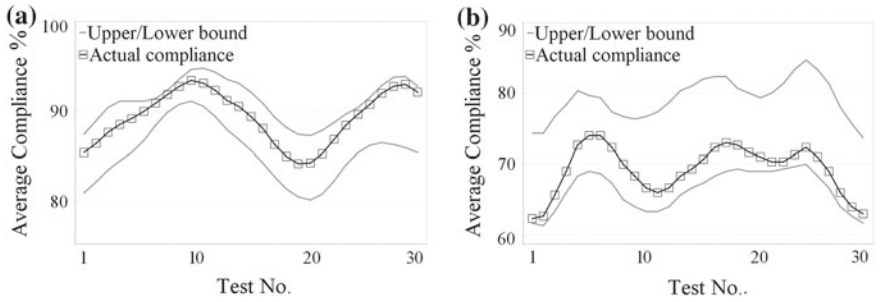


Fig. 8 Business logic compliance intervals. **a** Service with low error rate and tendency to deceive (S1). **b** Service with high error rate and tendency to deceive (S2)

'narrow' compliance interval bands are more trustworthy than the ones with 'wide' bands. While this rule may not be used mechanically to conclude one's compliance, it provides many insights for the evaluation.

8 Conclusions and Future Research Directions

In this chapter, we have looked at many different aspects of accountability-based approaches to verify compliance in collaborative business processes in the Cloud. We described a novel modelling of the collaborative business process, which captures both the *horizontal* structure (process level) and the *vertical* structure (infrastructure level) of the collaboration. With this model, we defined two generic types of compliance and thoroughly analyse the evidence and the logging required for their verification. We in depth discuss the extent to which compliance of a service can be proved and elaborate our approach to conclude one's *compliance interval* through *voting* and *probing*. We evaluate the costs of the evidence logging in an experimental business process deployed in Amazon EC2, the overhead observed is acceptable. Through fault injection experiments, we have demonstrated that *compliance intervals* concluded based on undeniable evidence, can help to make reasonable estimations about the services' true compliance.

As the research on accountability is still rapidly developing, there are many issues that remain open in this realm. Below, we identify a few potential research directions in accountability, together with our own view over some desirable properties of those research directions.

Generic versus domain specific. Compliance requirements are in diverse forms, thus their respective verification methods can be very different. The accountability mechanism designed for one domain may not be applicable to another. Hence the usability and value of such design is often limited. Generic accountability models, on the other hand, may be applicable to a range of scenarios; however, due to the lack of domain support, such models may be far too abstract for practical deployment. In

this case, compromise needs to be made to design the accountability for a practical use scenario, while having a generic model in a higher level of abstraction.

Incorporate accountability into remote systems. In many cases, remote systems (i.e. local systems of service providers) may be involved in a composed service network deployed in the Cloud. Modifying such remote systems to incorporate accountability can be very difficult. Rather than having accountability as a dedicated component during the design phase of the system, in this scenario, accountability needs to be planted into the already implemented remote systems (e.g. as middleware). Such approach is likely to suffer from the reduced stringency of accountability, as certain internal states of the system will not be captured.

A unified accountability framework. Achieving strong accountability requires several different functional blocks to be integrated and function together. Those blocks include: requirement definition, evidence management and system diagnosis/reasoning. The research studies we have reviewed often only focus on one functional block and make assumptions on the others. Since the researchers have different views and design goals, it is difficult to combine several sources to form a unified framework due to the inconsistency. In future research of accountability, a (relatively) more consistent understanding of accountability needs to be shared among different research groups. Certain flexibility should be allowed in the designed components, so they can be integrated together to form

References

1. Abraham T (2006) Event sequence mining to develop profiles for computer forensic investigation purposes. In: Australasian workshops on Grid computing and e-research. pp 145–153
2. Barford P, Sommers J (2004) Comparing probe-and router-based packet-loss measurement. IEEE Tans Internet Comput 8(5):50–56 (sep 2004)
3. Castro M, Liskov B (2002) Practical byzantine fault tolerance and proactive recovery. ACM Trans Comput Syst 20(4):398–461
4. Chase J, Yumerefendi A (2004) Trust but verify:accountability for network services. In: ACM SIGOPS European Workshop
5. Daniel F, Casati F, D'Andrea V (2009) Business compliance governance in service-oriented architectures. In: International Conference on Advanced Information Networking and Applications. pp 113–120
6. Druschel P, Haeberlen A, Kouznetsov P (2007) Peerreview: practical accountability for distributed systems. In: ACM SIGOPS symposium on Operating systems principles. pp 175–188
7. Du Y, Jiang C, Zhou M (2009) A petri net-based model for verification of obligations and accountability in cooperative systems. IEEE Trans Syst Man Cybern 39(2):299–308
8. Haeberlen A (April 2010) A case for the accountable cloud. In: ACM SIGOPS International Workshop on Large-Scale Distributed Systems and Middleware, vol 44(2), pp 52–57
9. Haeberlen A, Aditya P, Rodrigues R, Druschel P (2010) Accountable virtual machines. In: USENIX conference on Operating systems design and implementation. pp 1–16
10. Haeberlen A, Druschel P, Kouznetsov P (2006) The case for byzantine fault detection. In: Conference on Hot Topics in System Dependability. pp 5–10
11. Haeberlen A, Kouznetsov P, Druschel P (2007) Peerreview: Practical accountability for distributed systems. Technical report, Max Planck Institute for Software Systems (Mar 2007)

12. Hartog J, Dekker M, Corin R, Etalle S, Cederquist J (2005) An audit logic for accountability. In: International Workshop on Policies for Distributed Systems and, Networks. pp 34–43
13. Huang M, Peterson L, Bavier A (20006) Planetflow: maintaining accountability for network services. In: ACM SIGOPS Operating Systems, Review. pp 89–94
14. Korb K, Nicholson A (2004) Bayesian artificial intelligence. Chapman & Hall/CRC, London
15. Lamport L (1983) The weak byzantine generals problem. J ACM 30(3):668–676
16. Lin KJ, Chang S (2009) A service accountability framework for qos service management and engineering. Inf Syst E-Bus Manag 7:429–446, 10.1007/s10257-009-0109-5
17. Lin KJ, Panahi M, Zhang Y, Zhang J, Chang SH (2009) Building accountability middleware to support dependable soa. IEEE Trans Internet Comput 13(2):16–25 (mar 2009)
18. Mulo E, Zdun U, Dustdar S (2009) Monitoring web service event trails for business compliance. In: International Conference on Service-Oriented Computing and Applications. pp 1–8
19. Murata T (1989) Petri nets: properties, analysis and applications. Proc IEEE 77(4):541–580
20. Palankar M, Iamnitchi A., Ripeanu M, Garfinkel S (2008) Amazon s3 for science grids: a viable solution. In: Workshop on Data-aware, distributed Computing. pp 55–64
21. Rik E, Juliane D (2003) Reactive petri nets for workflow modeling. In: International Conference on Applications and Theory of Petri Nets. pp 296–315
22. Roughan M (2005) Fundamental bounds on the accuracy of network performance measurements. In: ACM SIGMETRICS international conference on Measurement and modeling of computer systems. pp 253–264
23. Rozinat A, Aalst W (2006) Conformance testing: measuring the fit and appropriateness of event logs and process models. In: Workshop on Business Process, Intelligence. pp 163–176
24. Schumm D, Leymann F, Ma Z, Scheibler T, Strauch S (2010) Integrating compliance into business processes. In: Multikonferenz Wirtschaftsinformatik
25. Shirey R (2007) Trustworthy system definition, at IETF RFC 4949 (Aug 2007), http://tools.ietf.org/html/rfc4949
26. Sommers J, Barford P, Duffield N, Ron A (2007) Accurate and efficient sla compliance monitoring. SIGCOMM Comput Commun Rev 37(4):109–120
27. Sommers J, Barford P, Duffield N, Ron A (2005) Improving accuracy in end-to-end packet loss measurement. In: Conference on Applications, technologies, architectures, and protocols for computer communications, pp 157–168. ACM, New York, NY, USA
28. Sommers J, Barford P, Duffield N, Ron A (2010) Multiobjective monitoring for sla compliance. IEEE/ACM Trans Netw 18(2):652–665 (Apr 2010)
29. Van der Aalst WMP, Dumas M, Ouyang C, Rozinat A, Verbeek E (2008) Conformance checking of service behavior. ACM Trans Internet Technol 8(3):1–30
30. Wang C, Chen S, Zic J (2009) A contract-based accountability service model. In: IEEE International Conference on Web Services. pp 639–646
31. Xiong Q, Zhang H, Meng B (2005) The practical detailed requirements of accountability and its application in the electronic payment protocols. In: IEEE International Conference on e-Technology, e-Commerce and e-Service. pp 556–561
32. Yao J, Chen S, Wang C, Levy D, Zic J (2010) Accountability as a service for the cloud. In: IEEE International Conference on Services, Computing. pp 81–90
33. Yao J, Chen S, Wang C, Zic J, Levy D (2010) Accountability as a service for the cloud: from concept to implementation with bpel. In: World Congress on Services (SERVICES). pp 91–100
34. Yumerefendi AR, Chase JS (2007) Strong accountability for network storage. ACM Trans Storage 3(3):11
35. Yumerefendi A, Chase J (2005) The role of accountability in dependable distributed systems. In First Workshop on Hot Topics in System Dependability
36. Zhang Y, Lin KJ, Hsu J (2007) Accountability monitoring and reasoning in service-oriented architectures. SOCA 1:35–50, 10.1007/s11761-007-0001-4

Reputation as a Service: A System for Ranking Service Providers in Cloud Systems

Wassim Itani, Cesar Ghali, Ayman Kayssi and Ali Chehab

1 Introduction

Today cloud service providers guarantee the quality of their services by defining a set of Service Level Agreements (SLAs) with their customers. SLAs binds the provider to a set of service level metrics typically related to service reliability, availability, performance, security, and billing. Generally, the SLA formally specifies the minimum expected service metrics that the provider is committed to supply and that the customer agrees to accede. A detailed description on SLA terms, levels, and the various legislations and conditions that accompany their specification is comprehensively presented in [34]. Unfortunately, SLAs typically lack any technical means of enforcement which leaves the customer's data and software processes under the total control of the cloud service provider. Any failure to meet the SLA terms and obligations will have disastrous effects on the cloud customer and provider. The effects range from losing reputation and client trust to legal compliance, and financial penalties that may lead to putting an end to the entire business. This fact will put pressure and responsibility on the customers when selecting a particular cloud service provider for running their business processes and storing data. The severity of this selection is further aggravated when we estimate the serious losses incurred when dealing with "misbehaving" cloud providers or the technical difficulties,

W. Itani (✉) · C. Ghali · A. Kayssi · A. Chehab
Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon
e-mail: wgi01@aub.edu.lb

C. Ghali
e-mail: csg04@aub.edu.lb

A. Kayssi
e-mail: ayman@aub.edu.lb

A. Chehab
e-mail: chehab@aub.edu.lb

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_13,

financial losses, and service downtimes accompanying the process of switching between service providers. Terabytes of data migration tasks over expensive communication links, software reconfiguration and adaptation, and data leakage and privacy implications are some factors that render the migration process highly expensive.

To alleviate customers concerns, and to aid them in selecting the appropriate cloud service provider at the outset, we believe that a cloud reputation service should be developed to rank service providers based on performance, security, and pricing criteria. Although other criteria may be incorporated, performance, security, and pricing criteria represent the "axis of evil" when dealing with misbehaving cloud providers and are really the major building blocks in the development of the initial customer-provider trust relationship. The advantages of a cloud reputation service would benefit both customers and providers. The cloud customer will be able to take better selection decisions when choosing a cloud infrastructure guided by a set of measurable and quantified reputation scores. On the other hand, the reputation service will encourage the cloud provider to enhance its provided services and offerings to attract a larger customer base which results in the development of a healthier and more competitive cloud computing market.

A major design goal to be achieved in the development of the reputation service is for it to be secure and accountable. Hence it should be built on top of a trusted cloud computing infrastructure for supporting the secure feedback processes and the tamper-proof logging mechanisms. The recommendation and reputation scores should be verified by a trusted third party trusted by the cloud customer and provider. This prevents the providers from denying responsibility of SLA incompliance.

In this chapter, we present RaaS, a secure and accountable reputation system for ranking service providers in cloud computing architectures. RaaS leverages previous research in the fields of cryptographic coprocessors, software division, and secure audit logging to provide a secure reputation reporting system whose results and recommendations can be published as a service and verified by trusted third parties or by the cloud service providers themselves. The reputation service is based on an assortment of ranking criteria ranging from multilevel performance and quality of service measures to security and pricing assessments. This makes RaaS a valuable IT component in supporting verifiable and accountable compliance with service-level agreements and regulatory policies, encouraging competition among cloud providers for better security and quality of service, and providing new and existing cloud customers with valuable advice for selecting the appropriate cloud service provider(s) that suffice their performance, budgeting, and security requirements. The RaaS reputation system does not rely on subjective feedback from cloud customers but rather carry out the reputation calculation based on observable actions extracted in real-time from the computing cloud itself. This feature allows the system to be impervious to attacks such as slandering, oscillation and Sybil attacks, which represent a major concern when designing reputation systems [11]. Moreover, the RaaS real-time feedback processes executed in the cloud (close to the transactions rather than in the customer address space) aids the reputation service to adapt dynamically to changes in the customer load and to resource variations in the provider infrastructure. This ensures the accurate representation of the reputation ranking schemes and suits the "elastic" and

complex nature of the virtualized cloud infrastructure. RaaS is implemented in a real cloud computing architecture using the VMware vSphere 4 cloud operating system [30]. The protocol implementation demonstrates the dynamics of reputation calculation and shows that the incorporated protocol operation imposes minimal overhead on the overall system performance.

The rest of this chapter is organized as follows: Sect. 2 provides a literature survey of the main models and protocols related to the proposed work. In Sect. 3 we discuss the trust model assumed in this work. Section 4 provides a brief overview of a proposed hardware-based security infrastructure in cloud computing. Section 5 presents the architecture of the RaaS reputation system. In Sect. 6, a proof of concept implementation of the RaaS protocols is evaluated and analyzed. Section 7 presents a brief economic study and feasibility analysis of the reputation system components. Section 8 summarizes the security axioms assumed in the system design and discusses a set of security exceptions. Conclusions and future directions are presented in Sect. 9.

2 Background

A large number of reputation assessment systems and trust establishment models have been introduced in service oriented architectures and the Internet. These systems aim at ranking providers based on the quality of their services and establishing the necessary trust models that govern the interaction among entities and agents in such open architectures. A common property shared by available service-oriented reputation systems is that they base the reputation calculation on the consumers' feedback. Since this form of feedback information maybe, in many cases, subjective, biased, or even malicious, the results and recommendations provided by this category of reputation systems is characterized by incompleteness and inaccuracy and thus cannot be fully trusted. The work in this chapter focuses on developing accountable protocols to extract objective feedback based on observable events generated by the computing cloud itself. Situating the source of reputation calculation within the address space of the cloud provider nearest to where the customer transactions are executed enhances the credibility of the reputation metrics and provides RaaS with a major advantage over existing reputation systems. In Ref. [20], Mármol and Pérez present some of the key challenges and threats facing the process of reputation calculation in distributed systems. According to Ref. [20], differentiating among honest and dishonest clients and handling malicious peers and information collectives are on top of the list of challenges and risks to be tackled when designing distributed reputation systems. Some of the proposed service-oriented reputation systems are presented in Refs. [2, 6, 12, 19, 21, 32, 36]. In Ref. [16], Jøsang et al. provide a comprehensive survey of trust and reputation systems in distributed systems and the Internet. A similar survey, but with less discussion and analysis, is presented in Ref. [18].

Research on accountability in service-oriented distributed systems has received a lot of attention after the proliferation of the cloud computing model. The emphasis on the importance of having accountable cloud computing platforms sprouts form the fact that in the computing cloud all the customer's hardware and software resources are under the full and direct control of the cloud service provider. In Ref. [22], Pearson and Charlesworth shed light on the importance of accountability as a means for resolving security and privacy risks in cloud computing. They thoroughly discuss the key elements that can be enriched and supported by a cloud accountability service. Of these they mention: policy compliance, responsibility determination, user trust assurance through privacy policies and contractual measures, and transparency in dealing with customer's private data and computations.

In [9], Haeberlen discusses the key requirements for establishing an accountable computing cloud and suggests the presence of an "Audit" primitive function that enables the customer to check the compliance of the provider with the service agreements. The recommendations and requirements provided in [9] are not accompanied with a technical solution. This chapter is viewed as a "call for action" for further research in this field as stated by the author.

A similar approach is presented by Yao et al. in [37] where the authors propose the design of a cloud accountability service by the addition of explicit logging service invocations in the business logic script. This method assumes the business logic script runs in a trusted domain to support the generation of secure log entries. We believe that this assumption does not acceptably comply with the cloud computing model where all the customer business logic is under the direct control of the service provider.

In [17], Li et al. present a set of benchmarking tools for estimating the performance and costs of deploying a customer cloud application on different cloud providers. This work suffers from a set of limitations that RaaS overcomes by design. These limitations are summarized in the following points:

1. It does not take into consideration the validity of the benchmarking results when possibly dealing with malicious cloud providers.
2. The performance measurements produced represent a snapshot in time. This makes the validity of the measurements affected by variations in customer's workloads or by any modification in the software, hardware, or network infrastructure. Moreover, changes in the provider's pricing model may impact the cost results.
3. It represents a client-side estimate of the provider's performance.
4. Last but not least, the work in [17] does not consider any security evaluation metrics. Security is a major requirement that should be considered when selecting a cloud provider. This fact becomes more evident if we know that the main factor hindering the adoption of cloud computing, particularly in healthcare and financial applications, is the lack of security and compliance support in current cloud service implementations.

3 RaaS Trust Model

3.1 RaaS Main Players

RaaS operates in a traditional cloud computing environment consisting of the two main communicating entities, namely a cloud service provider and a cloud customer. The provider manages and operates a cloud infrastructure of on-demand storage and processing services. The customer consumes the cloud services provided by the service provider and uses the Internet as the main communication medium for data exchange.

A cloud service provider could be an individual, a business enterprise, or a governmental or federal organization. In the same sense, a cloud customer could be any of the above mentioned entities.

The cloud applications managed by the provider abide with the Software-as-a-Service (SaaS) 3-tier enterprise architecture [4]. The RaaS reputation service targets this type of cloud applications since they currently dominate over 50 % of the cloud market share [23].

3.2 Customer–Provider Trust Levels

The trust model we assume in RaaS is principally governed by the significance and sensitivity of the cloud customer's data. Employing this approach makes the proposed reputation service comply with prominent cloud security and privacy protocols such as that presented in [14]. Based on this trust categorization, RaaS supports the following three trust levels in the services published by the cloud provider:

1. *Ultimate Trust*: this trust level is established when the provider is fully trusted for storing and processing customer's data with relatively low degrees of significance.
2. *Compliance-based Trust*: this trust level is established when customer data needs to be stored securely to achieve legal compliance regulations. The data security is mainly accomplished by encryption (by the service provider) using a provider-specific cryptographic key.
3. *No Trust*: in this trust level, the customer exhibits no trust in the cloud service provider. This is mainly due to the private nature of the data to be stored and processed in the computing cloud controlled by the provider. In the *No Trust* level, the customer is responsible for encrypting the sensitive data with trusted cryptographic keys before sending it to the cloud. Moreover, the software processing operations are carried out in isolated execution containers to protect the privacy of sensitive customer data. The isolated containers are implemented using tamper-proof cryptographic coprocessors that are configured, maintained, and distributed by a trusted third party as will be illustrated in Sect. 4.

Based on the trust levels presented above, customer data in the envisioned cloud computing environment is classified into three categories, before being transmitted for storage and processing in the computing cloud.

1. Non Sensitive (*NS*): This attribute marks non sensitive data that needs no security protection on the provider side. The cloud provider is trusted in this case to store the data in plaintext format without any form of encryption. Secure SSL [7] sessions could be optionally employed to secure the network transmission of *NS* data to the cloud.
2. Compliance Storage (*CS*): This privacy attribute marks customer data that should be stored encrypted in the cloud to comply with regulatory policies and recommendations. The encryption mechanism is the responsibility of the cloud service provider using a provider's specific cryptographic key. When sending *CS* data to the provider, the customer is forced to secure it over an SSL session to achieve network data confidentiality and integrity. The cloud provider is forced by the SLA to extract the SSL secured network data and store it encrypted. The encryption algorithm to be used and the ciphering key strength could be some parameters enforced by the regulatory policy or specified by the SLA.
3. Highly Sensitive (*HS*): Data marked with this attribute exhibits high sensitivity and privacy and should be concealed from the cloud service provider. For this reason, *HS* data is encrypted at the cloud customer side using a customer-trusted cryptographic key (K_{CID}) before being sent for storage in the cloud. This type of data can only be accessed and processed in the address space of the crypto coprocessor which shares with the customer the key K_{CID}.

4 Hardware–Based Security in the Cloud: A Proposed Technical Overview

For a reputation system to be trustworthy, i.e. capable of producing reliable, objective, and indisputable ratings and scores, it has to rely on a trusted cloud computing infrastructure that supports the accountability and credibility of the system protocol operation discussed in Sect. 5. RaaS relies on secure cryptographic coprocessors to enforce the accountability of the reputation calculation protocols. This section provides a brief overview of a proposed hardware–based security infrastructure in the cloud as inspired from our previous work on privacy-aware data storage and processing in cloud computing [14].

4.1 Secure Coprocessor Background

A cryptographic coprocessor is a tamper-proof piece of hardware that interfaces, mainly via a PCI-based interface, to a main computer or server. The chief security

property supported by a crypto coprocessor is its ability to provide a secure and isolated execution environment in the computing cloud [38]. A crypto coprocessor is a full-fledged computing system with its own processor, RAM, ROM, battery, network interface card, and persistent storage. However due to economic reasons, the coprocessor is usually less capable in terms of processing and memory resources than the main server it interfaces to, and thus cannot replace it. The main property that gives a crypto coprocessor its secure capabilities is the tamper-proof casing that encloses it and makes it resistant to physical attacks. A secure coprocessor tamper-resistance or tamper-responding mechanisms should reset the internal state of the coprocessor (RAM, persistent storage, processor registers) upon detecting any suspicious physical activity on the coprocessor hardware.

The only logical interface to the functionality of the coprocessor is done through a highly-privileged root process burned at manufacturing into the ROM of the coprocessor. This process represents a minimal operating system for the coprocessor. More on the root process is presented in Sect. 4.3. The input/output access to the cryptographic coprocessor can be either done locally via the main server system bus, or remotely via the coprocessor network card.

4.2 Coprocessor Authoritative Configuration and Distribution

The main authoritative entity responsible for configuring the crypto coprocessors and distributing them to cloud service providers is a trusted third-party (TTP) trusted by the cloud provider and customer. The resources of the crypto coprocessors installed in the computing cloud are shared among the cloud customers registered in the provider's storage and processing services. This resource sharing mechanism supports the economic feasibility of the solution and, most importantly, complies with the general cloud computing vision and paradigm. The TTP is responsible for securing the provider rating process and analyzing the rating records to generate the provider's absolute reputation score. Technically, the TTP loads a set of private/public key pairs into the persistent storage of the crypto coprocessor. Every public/private key pair (PU_{CID}/PR_{CID}) is to be allocated to a single customer when the latter registers with the cloud service provider. Upon registration, the cloud customer securely receives a copy of its public/private key pair. This can be achieved through an offline transaction or through a secure electronic session.

The PU_{CID}/PR_{CID} key pair set can be remotely updated by the TTP even after the crypto coprocessor is installed in the computing cloud. This remote key update mechanism is very important to support the registration of new customers and the service revocation of existing customers.

In addition to loading the customer's PU_{CID}/PR_{CID} key pair, the TTP also loads its own secret key K_{TTP} into the persistent storage of the crypto coprocessor. This key is needed by the TTP to remotely authenticate to the crypto coprocessor and to securely execute commands against it. Moreover, K_{TTP} is used to secure the integrity

and confidentiality of log records produced by the RaaS reputation protocols on the cloud provider's side as will be presented in Sect. 5.1.4.

The use of trusted third parties in security protocols is in many cases unavoidable to satisfy a set of security or application-specific requirements. This fact is supported by a wide set of successful protocol implementations that rely on trusted entities to deliver its security and trust commitments. A paragon example is the well-known PKI infrastructure used in the Internet today where all the communicating parties hierarchically trust one or more root Certification Authorities (CAs) to authenticate the principals' public keys. Add to this the signals received from the cloud computing industry itself where the notion of a trusted authority is getting more hype in the cloud supported by practical products proposed by big names such as RSA as indicated in the Cloud Trust Authority [3].

4.3 Coprocessor Process Model and Software Division

The crypto coprocessor process structure abides by the ABYSS [33] model. The customer applications are logically isolated using a highly-privileged root process that we refer to as the *RP* daemon. In addition to software isolation, The *RP* daemon also controls the access to the cryptographic keying material stored in the coprocessor's non-volatile storage and authenticates data, software, and remote configuration connections from the cloud customers and the TTP.

RaaS supports the economic and performance feasibility of the reputation system by adopting the software division concept. This concept urges the cloud customer to logically divide its software application components into two categories: protected and unprotected. The protected classification indicates that the logical component should be executed in a protected process in the address space of the cryptographic coprocessor. On the other hand, the unprotected classification indicates that the logical component can be executed in a traditional process on the main server's processor. The protected software part is digitally signed using PR_{CID} and encrypted by a customer-trusted key K_{CID} shared by the customer and the cryptographic coprocessor. The key sharing mechanism is achieved by executing an authenticated version of the Diffie-Hellman (D-H) key exchange protocol [5]. The authenticated D-H version authenticates the communicating parties and prevents man-in-the-middle attacks on the system. Whenever the software application is to be executed in the cloud, the *RP* daemon on the crypto coprocessor parses the protected software part and verifies its authenticity and integrity by checking the digital signature. Then it decrypts this software part using K_{CID} and executes it within the address space of the crypto coprocessor.

The software division mechanism is also applied to the RaaS performance evaluation protocols to ensure the accountability of the feedback logging mechanism on the cloud provider's side. In the same sense, the protocols execution steps are divided into secure (to be executed in the crypto coprocessor address space) and non-secure (executed on the main server processor) based on their role in the

performance evaluation process. This will be clearly illustrated when describing the details of the RaaS secure event monitoring and auditing protocols in Sect. 5.1.

5 Reputation System Architecture

RaaS provides a secure and accountable reputation service that does not rely on subjective feedback from cloud customers (although this kind of information could be integrated in the reputation service). The main source of input feeding the reputation calculation mechanism resides in the computing cloud itself at the provider side. The reputation service utilizes a trusted crypto coprocessor to provide a secure execution environment in the computing cloud, and thus produces authentic event logs that constitute the basis for the reputation score calculation.

The provider's reputation score is computed by the TTP and consists of three main components: a security reputation score, a pricing reputation score, and a performance reputation score. The performance reputation score is further subdivided into two sub scores: the data retrieval reputation score and the data processing reputation score. Basing the performance reputation score on the data retrieval and processing software operations is adopted since these operations represent the general atomic primitives upon which all higher level cloud transaction constructs are built.

It is worth mentioning here that other types of reputation components can be incorporated into the reputation calculation process, such as reputation components based on feedback from cloud clients or from third party auditing organizations. Cloud providers can also be ranked based on statically-analyzed criteria such as customer traction, technical background, and management track record. In this work we mainly focus on the security, performance, and pricing criteria due to the belief that they represent the top-of-the-list concerns in today's cloud applications. The ability to objectively monitor the variations in performance, security, and pricing is considered a highly challenging problem in today's cloud computing market. Moreover, striving to carry out the feedback processes in the computing cloud itself, close to the transactions rather than in the customer address space, aids the reputation service to adapt dynamically to changes in the customer load and to resource variations in the provider infrastructure. This ensures the accurate representation of the reputation ranking schemes and suits the "elastic" and complex nature of the virtualized cloud infrastructure.

The reputation service consists of three main phases:

- The Secure Event Monitoring and Auditing phase: this phase is carried out at the cloud provider side and is responsible for providing a trusted feedback mechanism to support the calculation of the performance reputation score components. It results in the generation of secure event logs by relying on the trusted coprocessor entity. The event logs are periodically delivered to the TTP site to be processed and analyzed.

- The Reputation Score Calculation phase: In this phase, the TTP calculates the performance, security, and pricing reputation scores. In the performance reputation score calculation, the TTP analyzes the providers' event logs and studies their compliance with the contractual terms and specifications promised in the SLAs. The reputation score will be proportional to the degree of provider's compliance with the SLAs. The process of calculating the performance reputation technically requires the existence of a set of secure and accountable protocols to accurately measure the performance of customer transactions on the provider side. For this reason, RaaS dedicates four secure performance monitoring and logging protocols to achieve this aim. A detailed description of these protocols is presented in Sect. 5.1.
  The security reputation score is calculated based on two main qualities:

1. The static security specifications specified in the SLA, mainly the degree of provider's compliance with regulatory policies, the network and storage encryption and authentication algorithms supported by the provider, and the strength of the cryptographic keys employed.
2. A dynamic penetration testing and vulnerability assessment mechanism executed by the TTP at random points in time.

Finally, the process of generating the pricing scores is achieved by relatively ranking the service prices specified by the different providers using a simple order statistics algorithm. More on the reputation score calculation is provided in Sect. 5.2.

- The Service Publication phase: this phase involves the online publication of the providers' reputation scores and all the management mechanisms for resolving disputes and complaints. The TTP provides through this phase the necessary procedures that allow the cloud service provider to check the validity and coherency of its reputation scores. More details on this phase are presented in Sect. 5.3.

5.1 Secure Event Monitoring and Auditing

Depending on the type and sensitivity of operations requested by the cloud customer, four performance evaluation and logging protocols are developed to securely generate the event logs about customer transactions. These event logs will be used later by the TTP to calculate the performance reputation score of the cloud provider. The RaaS performance evaluation protocols are presented below:

5.1.1 The Bulk Data Fetch Protocol

This protocol is executed whenever the cloud client sends a query for fetching bulk database/file data from the cloud storage facility. The requested data could belong to any of the data categories presented in Sect. 3.2. The main goal behind running this

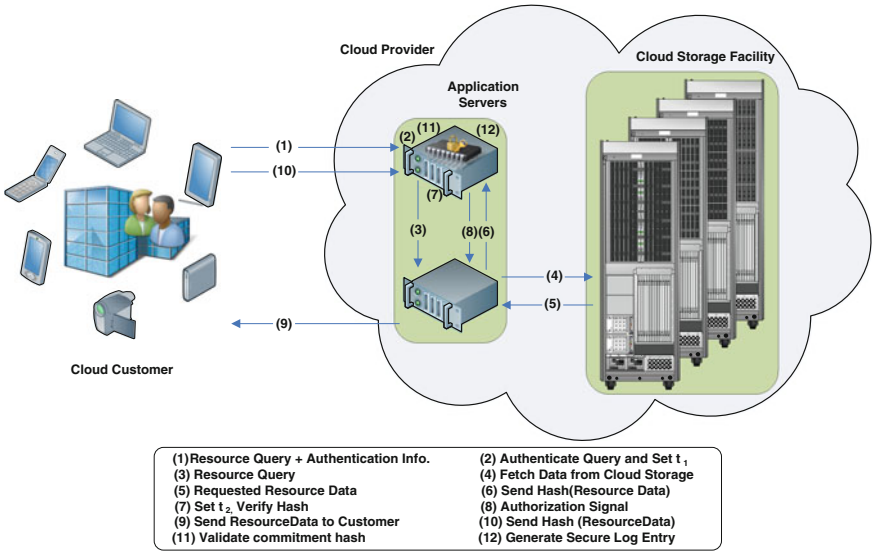


Fig. 1 The BDF protocol execution steps

protocol is to securely and accurately measure the time needed by the provider to retrieve the requested data from the cloud storage. Figure 1 presents the interaction between the trusted coprocessor and the main server processor to execute the Bulk Data Fetch (*BDF*) protocol steps. The 12 steps of the protocol are described as follows:

1. The cloud customer sends a database/file query to the crypto coprocessor. The query is accompanied by implementation-specific authentication information to authenticate the client request and the integrity of the query. This is mainly done by incorporating a cryptographic digital signature on the query structure using the client's private key PR_{CID}
2. The crypto coprocessor authenticates the client query and initiates a performance timer at time t_1.
3. The crypto coprocessor relays the query to the main server processor.
4. The main server processor executes the query against the cloud storage facility.
5. The main server receives the query results from the storage.
6. The main server processor sends a collision-resistant hash value of the query results to the secure coprocessor. The hash message has a double purpose. Firstly it represents a signaling message from the main server processor to the crypto coprocessor indicating that the data fetch process is accomplished. Secondly, the hash value constitutes a commitment that binds the cloud service provider with the results it fetched from the cloud storage. This commitment scheme prevents the cloud provider from rushing a fake and premature signal to the

secure coprocessor before the actual data fetching mechanism is really executed. This will be evident when discussing protocol step 11.

7. The secure coprocessor terminates the performance timer at time t_2 and verifies the validity of the hash value received in step 6.
8. The crypto coprocessor authorizes the main server processor to deliver the query data to the requesting cloud client.
9. The query data is delivered by the main server processor to the cloud client. The query data is authenticated by an implementation-dependent provider digital signature to ensure its authenticity and integrity.
10. The client validates the authenticity and integrity of the query results and sends the hash value of the query data received to the crypto coprocessor.
11. The crypto coprocessor compares the commitment hash value received from the main server processor in step 6 with the hash value received from the client in step 10. Equal hash values indicate that the cloud provider has fully accomplished the query request before sending the finish signal to the crypto coprocessor in step 6.
12. After validating the two hash values, the crypto coprocessor generates a secure log entry containing, most importantly, the value of the time interval (t2 – t1). A detailed description of the secure log generation mechanism is presented in Sect. 5.1.4.

5.1.2 The Data Fetch for Execution Protocol

This protocol is a variant of the *BDF* protocol with the exception that the fetched data is not sent to the cloud customer; instead it is consumed by internal software processes. If the software processes are secure, i.e. running in the address space of the crypto coprocessor, then steps 8, 10, 11, and 12 will not be needed since in this case the software processes receiving the data are already running on a trusted platform. Step 9 will be modified to deliver the requested data to a software process instead of to the cloud customer. This will be illustrated in the next section.

5.1.3 The Data Execution Monitoring Protocol

This protocol is executed after the Data Fetch for Execution (*DFE*) protocol. Its chief goal is to securely measure the time needed by the internal software processes on the cloud provider side to process the data retrieved by the *DFE* protocol. A very important property of the Data Execution Monitoring (*DEM*) protocol is that it operates on non sensitive customer data of the *NS* and *CS* types. Processing operations on sensitive data belonging to the *HS* category do not necessitate the presence of a dedicated performance evaluation protocol. This fact is illustrated as follows: Due to the high sensitivity of the processed *HS* data, the software processes handling it should be of the secure type; that is running in the address space of the secure crypto coprocessor. Since the processing platform is trusted and controlled by the

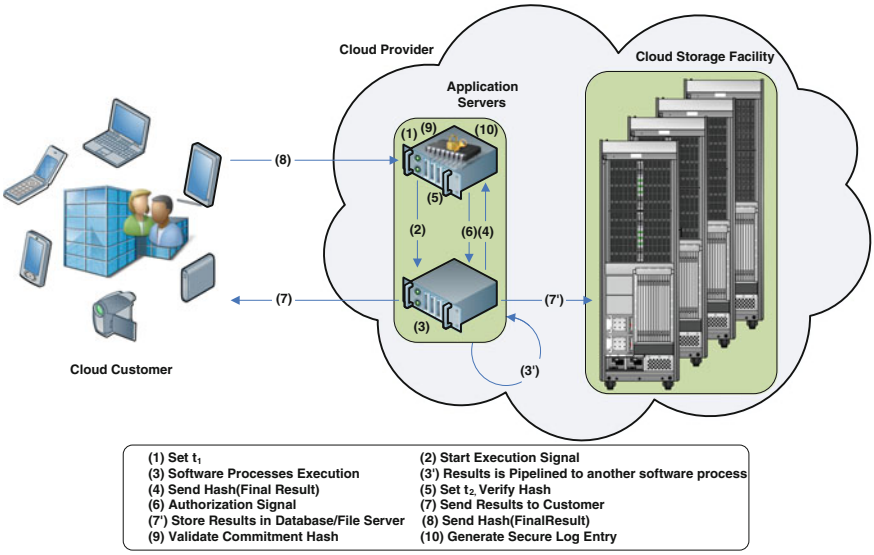


Fig. 2 The DEM protocol execution steps

TTP, no performance evaluation is carried out or required and thus this protocol has no direct effect on the reputation score calculation. Figure 2 illustrates the *DEM* protocol execution steps. The 10 steps of the protocol are described as follows:

1. The secure coprocessor initiates the performance timer at time t_1 which identifies the start of data processing.
2. A "start execution" signal is sent to the main server processor to start the data processing operations.
3. The software execution is carried out on the main server processor. Intermediate output results may be pipelined to other software processes before producing the final execution result.
4. The non secure processor sends a hash of the final result to the secure coprocessor. The purpose of this hash value is analogous to that described in the *BDF* protocol step 6 (execution completion signal and output result commitment).
5. Upon receiving the hash value in step 4, the secure coprocessor terminates the performance timer at time t_2 and checks the validity of the output result hash value.
6. The secure coprocessor authorizes the non secure processor to send the processing results to the cloud customer or to store them in the cloud persistent storage.
7. The data execution result is delivered to the customer or stored in persistent cloud storage.
8. This step is executed if the cloud customer is the intended recipient of the processing results (step 7). The cloud customer validates the authenticity and integrity

of the data received in step 7 and sends the hash of the processing results to the secure coprocessor.

9. The crypto coprocessor compares the commitment hash value received from the untrusted processor in step 4 with the hash value received from the client in step 8. Equal hash values indicate that the cloud provider has fully accomplished the data processing operation before sending the finish hash signal to the crypto coprocessor in step 4.
10. After validating the two hash values, the crypto coprocessor generates a secure log entry containing, most importantly, the value of the time interval $(t_2 - t_1)$.

5.1.4 The Secure Log Generation and Storage Protocol

In this section, we present the format of the log records generated by the secure performance evaluation protocols executed in the address space of the cryptographic coprocessor. This section also describes the protocol employed to guarantee the confidentiality, authenticity, and integrity of the log records when stored on the untrusted provider side. Storing event logs on untrusted storage is compulsory due to the limited capacity of the storage resources available on the trusted crypto coprocessor.

In RaaS, each log record entry consists of the following fields:

- *RID* : A unique identifier assigned to the log record.
- *KID* : The index or identification of the cryptographic key K_{TTP} installed by the TTP in the crypto coprocessor (see Sect. 4.2). K_{TTP} is used to generate the log message authentication codes as will be described later in this section. The field is required in the log since the TTP may periodically refresh K_{TTP} for security purposes. Thus the ID of the K_{TTP} key is included to unambiguously identify the correct key used.
- *CID* : The cloud customer identification number.
- *PID* : The cloud service provider identification number.
- *SID* : The application or service identification number.
- *SLAID* : An index to the SLA contract governing the transaction generating this log entry.
- R_{Query} : The cloud client resource query to the cloud storage facility. This field is null in the log records generated by the *DEM* protocol.
- β : The data size in Bytes retrieved/processed in the transaction.
- τ : The time required to retrieve/process the data in the transaction in milliseconds.
- *TS* : The timestamp of the transaction.

In addition to the above mentioned fields, application specific transaction fields may also be included.

To protect the confidentiality and integrity of the log records when stored on the untrusted cloud storage space, an authenticated and encrypted hash chain data structure [25] is employed. The hash chain cryptographically links the encrypted log records to assure their confidentiality and to prevent any undetectable malicious modification on their contents. This mechanism is illustrated in Fig. 3 where $Hash_i$

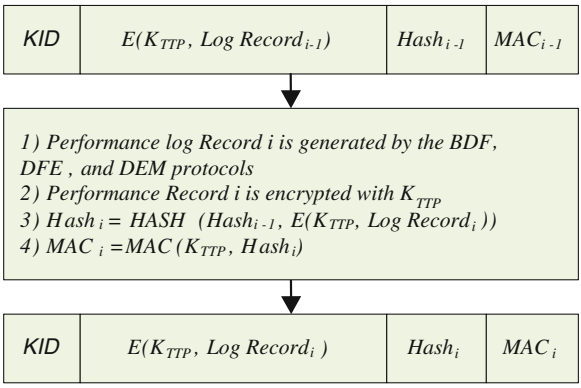


Fig. 3 The secure log generation and storage protocol (SGEN) protocol for securing log generation and storage

represents the hash chain value of the ith encrypted log record. $Hash_i$ is constructed by hashing the ith encrypted log record entry and the hash chain value of the previous encrypted log record $Hash_{i-1}$. Since $Hash_i$ includes $Hash_{i-1}$, it is possible to verify the integrity of all encrypted log records by only checking the authenticity of $Hash_i$. $Hash_i$ is authenticated using a message authentication code MAC_i generated using the cryptographic key K_{TTP}. Since the log record generation is entirely executed in the address space of the trusted crypto coprocessor and since K_{TTP} is securely stored in the coprocessor's persistent storage, the cloud service provider will not be able to decrypt the contents of the log records, or modify them without detection.

5.2 Reputation Calculation

This section describes the reputation score calculation by the TTP for providers registered in the reputation service. As mentioned previously, three main reputation score components are calculated: a performance reputation score consisting of data retrieval and processing sub scores, a security reputation score based on static and dynamic security assessments, and a pricing reputation score that relatively ranks the providers based on their offered service costs.

5.2.1 Reputation Score per Period

The reputation scores for providers are calculated periodically by the TTP which can carry out the calculation on a monthly, quarterly, semi-annual, or annual basis.

Performance Reputation Scores:

The securely generated logs are collected from the different provider sites and analyzed in order to calculate the provider performance reputation scores. The calculation is done as follows: for each log entry, the TTP utilizes the and fields to calculate the transaction data retrieval/processing rates and compares them to the rates promised in the SLA based on the following equations:

$$\Phi_{\mathrm{r},i}\,[\%] = \frac{\frac{\beta_i}{\Psi_\mathrm{r}} - \tau_i}{\frac{\beta_i}{\Psi_\mathrm{r}}} \times 100 \tag{1}$$

$$\Phi_{\mathrm{p},i}\,[\%] = \frac{\frac{\beta_i}{\Psi_\mathrm{p}} - \tau_i}{\frac{\beta_i}{\Psi_\mathrm{p}}} \times 100 \tag{2}$$

$\Phi_{\mathrm{r,i}}$ and $\Phi_{\mathrm{p,i}}$ respectively represent the percent improvement over the SLA retrieval rate Ψ_r and processing rate Ψ_p for the *i*th log record. The *SLAID* log field identifies the SLA contract governing the customer transaction producing the respective log record.

By applying averaging and normalization operations on the calculated Φ_r and Φ_p log entry improvement rates, the overall retrieval and processing reputation scores R_r and R_p can be computed. R_r and R_p are functions of the average retrieval/processing rates $\overline{\mathrm{X}}_\mathrm{r}$ and $\overline{\mathrm{X}}_\mathrm{p}$ and their standard deviations σ_r and σ_p , respectively.

$$\overline{X}_\mathrm{r}\,[\%] = \frac{1}{N_r} \times \sum_{i=1}^{i=N_r} \Phi_\mathrm{r}, i \tag{3}$$

$$\overline{X}_\mathrm{p}\,[\%] = \frac{1}{N_p} \times \sum_{i=1}^{i=N_p} \Phi_\mathrm{p}, i \tag{4}$$

$$\sigma_\mathrm{r}\,[\%] = \sqrt{\frac{1}{N_r} \times \sum_{i=1}^{i=N_r} \left(\Phi_\mathrm{r}, i - \overline{X}_\mathrm{r}\right)^2} \tag{5}$$

$$\sigma_\mathrm{p}\,[\%] = \sqrt{\frac{1}{N_p} \times \sum_{i=1}^{i=N_p} \left(\Phi_\mathrm{p}, i - \overline{X}_\mathrm{p}\right)^2} \tag{6}$$

$$R_\mathrm{r} = \left[\lambda + \left(\lambda \times \frac{\overline{X}_\mathrm{r}\,[\%]}{100}\right)\right] \pm \left(\lambda \times \frac{\sigma_\mathrm{r}\,[\%]}{100}\right) \tag{7}$$

$$R_\mathrm{p} = \left[\lambda + \left(\lambda \times \frac{\overline{X}_\mathrm{r}\,[\%]}{100}\right)\right] \pm \left(\lambda \times \frac{\sigma_\mathrm{p}\,[\%]}{100}\right) \tag{8}$$

where N_r and N_p are the number of retrieval and processing log entries per provider, respectively, λ and is a normalization constant that represents the middle value of the reputation score range.

Setting $\lambda = 5$, we get a retrieval and processing reputation scores range that lies in]0, 10[.

Numerical example:

This example demonstrates the calculation of the cloud provider retrieval reputation score resulting from the following sample log entries (the processing reputation score calculation follows analogous procedure).

SLA_6: Ψ_r = **6** [**Mbps**] Ψ_p = **15**[**Mbps**]

SLA_11: Ψ_r = **4** [**Mbps**] Ψ_p = **11**[**Mbps**]

From Eqs. (1) and (2) we calculate, for each entry, the percent improvement over the respective SLA. For instance, the improvement $\Phi_{r,1}$ resulting from the first record in the sample log is calculated as follows:

$$\Phi_{r,1}\,[\%] = \frac{\frac{50}{6} - 6.5}{\frac{50}{6}} \times 100 = 22\,\%$$

Repeating the same calculation for records 2 through 4 , we obtain the following improvement rates:

$$\Phi_{r,2}\,[\%] = 30.9\,\%$$

$$\Phi_{r,3}\,[\%] = 17.3\,\%$$

$$\Phi_{r,4}\,[\%] = -9.1\,\%$$

Applying Eqs. (3) and (5), we calculate the average and standard deviation of the obtained improvement rates as follows:

$$\overline{X}_r\,[\%] = 15.275\,\%, \quad \sigma_r\,[\%] = 9.1\,\%$$

Therefore, the retrieval reputation score based on Eq. (7) using $\lambda_r = 5$ (5 is the value that gives a reputation score in the]0, 10[range) is equal to:

$$R_r = 5.76 \pm 0.45$$

Security Reputation Score:

The security reputation calculation in RaaS is carried out in two main phases: a static analysis phase and a dynamic penetration testing phase. In the static analysis phase the TTP statically analyzes the provider's SLAs for security-related terms and specifications, classifies them into a set of security categories, and assigns a reputation

weight to each category. The category reputation weight is provided based on the quality of security this category represents.

Three security categories are currently supported in RaaS:

1. The degree of provider's compliance with regulatory policies and recommendations such as the Health Insurance Portability and Accountability Act (HIPAA) [10] for securing medical records and patient's information and the Gramm-Leach-Bliley Act [15] for ensuring the confidentiality of financial records and banking transactions for any institution providing a financial service. The more regulatory policies the provider supports, the higher is the reputation weight W_1 assigned to this security category.
2. The set of cryptographic protocols supported by the cloud service provider for providing the different network and storage security services. The weight W_2 assigned to this category is based on the security and performance properties of the cryptographic protocols supported. For example, a cloud provider using the AES encryption cipher for providing the storage confidentiality services, a SHA-512-based MAC algorithm for providing the storage integrity services, and the SSL v3.0 protocol for securing the network communication will definitely get a higher reputation weight on this category than a provider which supports DES for storage confidentiality, MD5-based MAC algorithms for storage integrity and which does not provide any form of network security.
3. The strength of the symmetric and asymmetric keys used in the cryptographic protocols. Larger key sizes in modern cryptographic algorithms provide higher grade security and are thus assigned a higher reputation weight W_3.

The list of security categories is implementation dependent and can be extended with additional classification groups as devised by the TTP. The general equation representing the security reputation score $R_s\,(static) \in [0, 10]$ resulting from the static analysis phase is given as follows:

$$R_s\,(static) = \frac{10}{\sum_{i=1}^{N} \max\,(W_i)} \times \sum_{i=1}^{N} W_i \tag{9}$$

N is the number of security categories and max (W_i) is the maximum weight value that can be assigned to the ith security category.

In the dynamic penetration testing phase the TTP performs a security assessment of the provider's site using advanced and up-to-date vulnerability scanning techniques. This assessment procedure is executed at a random point in time during each reputation calculation period. The dynamic penetration testing phase aims at scanning the provider's network resources and applications for known vulnerabilities to verify the immunity of the provider's system against possible security exploitations. Currently, there exist a plethora of highly accessible penetration testing tools capable of executing different types of network and system attacks. On top of the list are the Nmap [28] and Nessus [27] network scanners which support a large set of active and passive attacks to assess the security of networks and applications. The demon-

stration example introduced later in this section presents some of the attack types employed in the penetration testing phase. Note that carrying out the penetration tests at a contingent point during the relatively long reputation calculation period should in no way affect the quality of the provider services or the business of the cloud users.

Quantifying the dynamic security score $R_s\ (dynamic)$ is accomplished by assigning a prevention weight to each attack tested. The prevention weight is incorporated into the dynamic security score only if the provider was capable of preventing the respective attack. The value of the prevention weight is determined by the TTP based on the security implications that may result if the attack was successful. For example, Denial of Service (DOS) and buffer overflow attacks are given more weight than port scanning or OS fingerprinting attacks.

Let the set $A : \{A_1, \ldots, A_J\}$ represent the assigned reputation weights of the J attacks tested in the penetration testing phase and the set $B : \{B_1, \ldots, B_K\}$ represent the assigned reputation weights of the K attacks prevented by the provider $(K \leq J)$. The dynamic security score $R_s\ (dynamic) \in [0, 10]$ and is calculated by the following equation:

$$R_s\ (dynamic) = \frac{10}{\sum_{i=1}^{J} A_i} \times \sum_{i=1}^{K} B_i \tag{10}$$

The overall security reputation score R_s is calculated by taking the weighted average of the static and dynamic security scores according to the following equation:

$$R_s = \frac{a \times R_s\ (static) + b \times R_s\ (dynamic)}{(a + b)} \tag{11}$$

The parameters a and b are selected by the TTP to control the contribution of the static and dynamic components in the overall security reputation score calculation.

It is worth mentioning here that a high ranking in the static and dynamic testing phases does not guarantee that the provider site is completely secure (in fact no security testing scheme does). However, a high security reputation score provides a relatively high confidence level in the security capabilities, mechanisms, and measures implemented at the provider cloud.

Numerical example:

Tables 2 and 3 present a numerical demonstration for respectively calculating the static and dynamic security reputation scores. In Table 1 three security categories and their corresponding reputation weights are presented. The "supported" column indicates whether the particular security specification is supported by the example provider SLA. In each category, the weight of the individual security specification supported is added to get the category's reputation weight W_i. Using the W_i values of Table 2 in Eq. (9) we get:

$$R_s\ (static) = \frac{10}{(10 + 10 + 10)} \times (8 + 10 + 7) = 8.33$$

Table 1 Sample data retrieval log entries

RID	KID	CID	PID	SID	SLAID	R_{Query}	β(Mbits)	τ(msec)	TS
1	17	101	Serv_corp	19_Banking	6	Query_1	50	6,500	TS_1
2	17	101	Serv_corp	19_Banking	6	Query_2	40	4,600	TS_2
3	17	301	Serv_corp	23_Ledger	11	Query_3	60	12,400	TS_3
4	17	201	Serv_corp	41_MedServe	6	Query_4	55	10,000	TS_2

Table 2 Sample SLA security categories and respective reputation weights

Security category	Individual weight			Supported	Reputation weight $R_s \in [0, 10]$
Regulatory compliance	HIPAA		6	✓	
	Gramm-Leach-Bliley		2	✓	8
	Sarbanes-Oxley		2		
Cryptographic protocols	AES		5	✓	
	Twofish		4		
	Triple DES		2		
	DES		1		10
	SSL v3.0		5	✓	
	PCT		2		
	SHTTP		2		
Key strength	Symmetric	256	5		
		192	4		
		128	3	✓	7
	Asymmetric	2048	5		
		1024	4	✓	
		512	2		

Table 3 presents a sample penetration test scenario to dynamically asses the security of the cloud IT infrastructure. The attack prevention weight reflects the relative seriousness of the attack and the effect it has on normal system operation. The "prevented" column indicates whether the provider was capable of preventing or passing the particular penetration test. Using the weight values presented in Table 3 in Eq. (10) we get:

$$\begin{aligned} R_s\,(dynamic) &= \frac{10}{(10+3+10+10+10+10+8+7+9+2+2)} \\ &\quad \times (10+10+10+10+8+9) \\ &= 7.03 \end{aligned}$$

After calculating the static and dynamic security scores, we can find the overall security score using Eq. (11). Setting (the static and dynamic scores equally contribute to the security reputation score) we get:

Table 3 Sample penetration testing attacks and the corresponding prevention weights

Penetration attacks	Prevention weight $A_i \in [0, 10]$	Prevented
SYS flooding	10	✓
ICMP mapping	3	
Default accounts	10	✓
Buffer overflow	10	
SQL injection	10	✓
OS command injection	10	✓
TCP ISN guessing	8	✓
TCP RST attack	7	
Cross site scripting	9	✓
Port scanning	2	
Host fingerprinting	2	

Note that in real world implementations the list of security categories and penetration tests is extended with a wide variety of possible security specifications and attack scenarios.

Pricing Reputation Score:

The pricing reputation score is calculated by ranking a set of service providers according to the cost of the cloud services they provide. The ranking is achieved by employing a simple order statistics algorithm.

Assume that we have a set of n service providers $E_1, \ldots, E_n$ providing a cloud service S . Let the cost of S as specified by E_i be C_i . Applying the order statistics algorithm on the set $C : \{C_1, \ldots, C_n, \}$ results in ranking C in ascending cost order. The index r of the provider's cost C_i in the ordered set provides the pricing reputation score E_i of with respect to the cloud service S according to the following equation:

$$R_s^s = \begin{cases} 10, & r = 1 \\ \frac{10 \times (\mu - r)}{\mu}, & 2 \geq r \geq \mu \end{cases} \tag{12}$$

where is the number of distinct cost values in the set C. Note that $R_c^S \in [0, 10]$ In other words, the lower the relative price of the service delivered, the higher the pricing reputation score attained.

5.2.2 Cumulative Reputation Score

This section describes how the cumulative performance, security, and pricing reputation scores over the periods $P_1, P_2, \ldots, P_{i+1}$ are calculated.

The retrieval and processing performance scores are calculated by only analyzing the log entries generated in the period P_{i+1}. This is important for reducing the storage and processing requirements at the TTP site considering the extremely large number of log entries that need to be stored and analyzed for calculating the reputation scores

of the different registered cloud providers. Note that the equations presented in this section are generalized to satisfy the calculation of both the retrieval and processing reputation scores.

Let $\overline{X}\big|_{P_1}^{P_i}$ and $\sigma\big|_{P_1}^{P_i}$ be the total average rate and standard deviation of improvement over the SLA from period $P_1 \rightarrow P_i$, respectively. Let $\overline{X}_{P_{i+1}}$ and $\sigma_{P_{i+1}}$ be the average rate and standard deviation of improvement in period P_{i+1}. To calculate the cumulative average rate $\overline{X}_T$ and standard deviation σ_T over the period from P_1 to P_{i+1}, the following equations are used:

$$\overline{X}_T = \frac{\left(\overline{X}|_{P_1}^{P_i} \times N|_{P_1}^{P_i}\right) + \left(\overline{X}_{P_{i+1}} \times N_{P_{i+1}}\right)}{N|_{P_1}^{P_i} + N_{P_{i+1}}} \tag{13}$$

$$\sigma_T = \sqrt{\frac{1}{N|_{P_1}^{P_i} + N_{P_{i+1}}} \sum_{k=1}^{N|_{P_1}^{P_i} + N_{P_{i+1}}} \left(\Phi_k - \overline{X}_T\right)^2} \tag{14}$$

where $N\big|_{P_1}^{P_i}$ is the number of log entries processes in periods $P_1 \rightarrow P_i$ and $N_{P_{i+1}}$ is the number of log entries processed in the period P_{i+1}.

To facilitate the calculation of σ_T without the necessity of storing and reprocessing all the $N\big|_{P1}^{P_i}$ log entries, we assume that $\overline{X}|_{P_1}^{P_i} \approx \overline{X}_T$. This is considered a valid assumption as $N\big|_{P1}^{P_i}$ is significantly larger than $N_{P_{i+1}}$. Equation (14) can be rewritten as follows:

$$\sigma_T = \sqrt{\frac{1}{N|_{P_1}^{P_i} + N_{P_{i+1}}} \left\{ \sum_{i=1}^{N|_{P_1}^{P_i}} \left(\Phi_i - \overline{X}_T\right)^2 + \sum_{j=1}^{N_{P_{i+1}}} \left(\Phi_j - \overline{X}_T\right)^2 \right\}} \tag{15}$$

which simplifies to:

$$\sigma_T = \sqrt{\frac{1}{N|_{P_1}^{P_i} + N_{P_{i+1}}} \left\{ \left(\sigma|_{P_1}^{P_i} \times N|_{P_1}^{P_i}\right) + \sum_{j=1}^{N_{P_{i+1}}} \left(\Phi_j - \overline{X}_T\right)^2 \right\}} \tag{16}$$

Obtaining the values of $\overline{X}_T$ and σ_T, the cumulative retrieval and processing reputation scores and can be calculated based on Eqs. (7) and (8), respectively.

Calculating the cumulative security reputation score is straightforward. Let $R_s\big|_{P_1}^{P_i}$ be the cumulative security reputation score from period $P_1 \rightarrow P_i$ and $R_{S_i P_{i+1}}$ be the security score in period P_{i+1}. The cumulative security score $R_{T,s}$ from $P_1 \rightarrow P_{i+1}$ is calculated as follows:

The cumulative pricing reputation score $R_{T,c}^S$ can be calculated analogously.

5.3 Reputation Publication

After the TTP accomplishes the log analysis and reputation score calculation, it publishes the results online as a cloud service. The TTP also provides a set of procedures for resolving disputes and enabling the cloud service provider to check the validity and coherency of its published reputation scores. The main procedures employed include:

1. The design of a provider complaint form containing a detailed description of the problem or concern. The key elements of this form should include a specification of the reputation component disputed (mainly performance reputation scores are disputed since these are based on active protocol execution in the cloud), the time period in which the score was calculated and published, and the reason why the provider is concerned about the validity of the reputation score.
2. The design of an online complaint form submission system that handles the providers' contentions.
3. The specification of the maximum time period during which the provider is allowed to raise a dispute after the publication of the reputation scores. This is important for controlling the amount of storage needed at the TTP side to maintain the providers' reputation logs.
4. The specification of the extra charges incurred when the provider requests the recalculation of its reputation scores.
5. The design of a secure mechanism for delivering the event logs resulting in the disputed reputation score to the provider, together with the business logic necessary to analyze the event logs and calculate the reputation score.

6 RaaS Prototype Implementation

A prototype proof of concept of the RaaS reputation service algorithms and protocols is implemented on the VMware vSphere cloud computing platform. vSphere is a virtualization framework for building robust cloud computing applications and services. Currently, vSphere is considered the leading industry cloud operating system supporting a large set of server, storage, and network virtualization services. The main advantage in employing the VMware vSphere platform is its hardware and operating system-independent abstraction layer that allows the dynamic and on-demand hardware resource sharing and management by multiple client virtual machines. We created five client virtual machines (VMs) on the vSphere virtualization server to support the execution of the customer application business logic. The guest operating systems running on these machines are: 2 Windows XP SP3 VMs, 1 Windows 7 VM, and 1 Ubuntu 9.04 VM. The vSphere physical server is supplied with an Intel Core i7 CPU Q 720 running at 1.6 GHz and 4GB of RAM. We implemented 2 sample customer enterprise applications to run in the vSphere cloud: A Customer Relationship Management (CRM) application and a Human Resource

(HR) management application. The applications execute SQL queries on an SQL Server 2005 RDBMS running on a Windows 2003 R2 SP2 Server having two Intel Xeon CPUs running at 3.8 GHz and 2 GB of RAM.

The vSphere virtualization server communicates with the database server using a 100 Mbps LAN interface. The enterprise applications are developed using the C# programming language.

To implement the functionality of a secure cryptographic coprocessor, we assume that one of the core CPUs on the virtualization server is the secure crypto coprocessor while the other core CPUs are those of the main untrusted server. We believe this assumption provides a viable proof of concept sufficient for testing the system configuration, functionality, and reputation mechanisms.

The implementation focused principally on the realization of the RaaS performance monitoring and logging protocols and on the penetration testing for assessing the provider's dynamic security reputation score. This is due to the fact that these mechanisms represent the dynamic execution components in the reputation calculation system.

For evaluating the performance reputation score, a set of 2,000 data retrieval and processing transaction events is generated by the implemented performance monitoring protocols. The event records generated are evenly distributed over 4 virtual time periods of 5 days each. Roughly 100 transactions are executed on the database server per day. The data size, retrieved or processed, is homogenously distributed over the 10 KB–10 MB range. The transaction logs are analyzed and processed based on the equations presented in Sect. 5.2 to produce the RaaS performance reputation scores. Figures 4 and 6 respectively present the individual retrieval and processing reputation scores derived at each time period. The graphs in Figs. 4 and 6 are based on the retrieval and processing reputation parameters presented in Tables 4 and 6 respectively. In Figs. 5 and 7 the individual reputation scores in each period are linked together based on the equations presented in Sect. to generate the cumulative retrieval and processing reputation scores, respectively. The reputation graphs of Figs. 5 and 7 are based on the reputation data in Tables 5 and 7, respectively. The cumulative reputation results reflect a smooth continuous representation of the provider's reputation that can be extrapolated over the entire set of time periods. This cumulative representation provides cloud consumers with accountable reputation attributes that aid in anticipating the future behavior of service providers. To simulate a realistic operating environment for running the performance evaluation protocols, we introduced random load and stress factors on the application and database servers in each time period. This is achieved using the SQLIOSim tool [35]. The variation in the retrieval/processing reputation patterns presented in Figs. 4, 5, 6, 7 are directly related to the alteration in the intensity and duration of the load and stress factors on the system resources.

Employing the RaaS performance monitoring protocols and secure log generation mechanisms added minimal overhead to the overall application performance. This is primarily due to two main reasons: (1) the extremely fast inter-process communication and interaction on the local server machine and (2) the efficient design of the software division mechanisms that govern the interaction among the secure

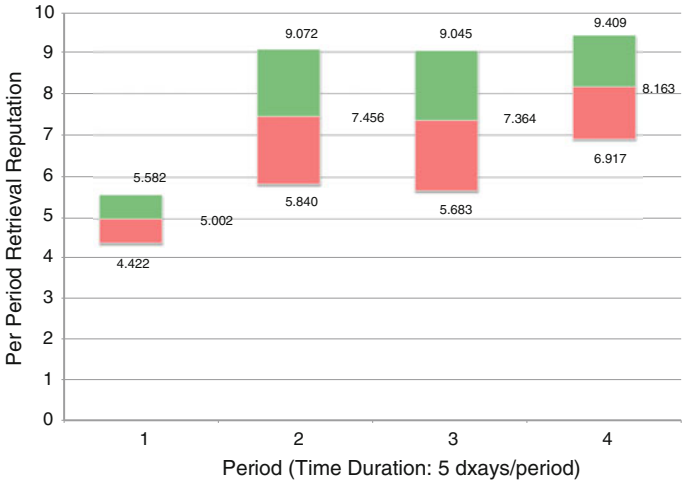


Fig. 4 Retrieval reputation calculation from the sample implementation

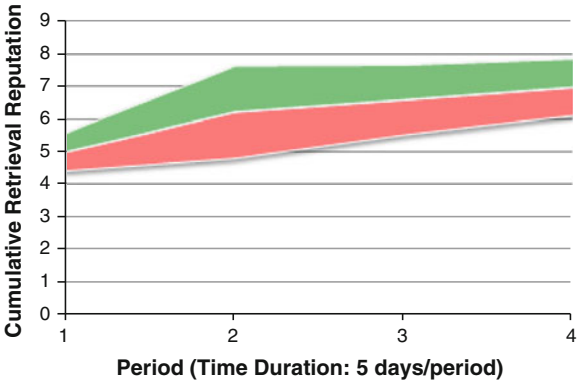


Fig. 5 Cumulative retrieval performance calculation

Table 4 Retrieval reputation parameters for Fig. 4

$\overline{X}_r$	0.053	49.135	47.287	63.275
σ_r	11.606	32.324	33.627	24.916
λ_r	5.0	5.0	5.0	5.0
R_r	5.002 ± 0.580	7.456 ± 1.616	7.364 ± 1.616	8.163 ± 1.616

coprocessor and the main server core processors. This fact is illustrated in Fig. 8 which presents the average time in seconds consumed by the data retrieval and processing operations with and without the application of the RaaS performance monitoring protocols. The overhead is roughly 9.4, 6.9, and 4.6 % for the BDF, DFE, and DEM protocols respectively. Pragmatically, this is considered minimal for a diverse set of enterprise cloud services when considering the wide range of query sizes operated on.

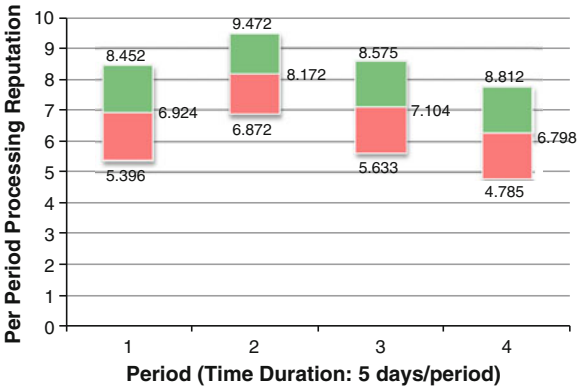


Fig. 6 Processing reputation calculation from the sample implementation

Table 5 Cumulative retrieval reputation parameters for Fig. 5

$\overline{X}_{T,r}$	0.053	24.594	32.158	39.937
$\sigma_{T,r}$	11.606	28.341	21.290	17.308
λ_r	5.0	5.0	5.0	5.0
$R_{T,r}$	5.002 ± 0.580	6.229 ± 1.417	6.607 ± 1.064	6.996 ± 0.865

Table 6 Processing reputation parameters for Fig. 6

$\overline{X}_P$	38.496	63.449	42.086	35.978
σ_P	30.5586	25.99226	29.42078	40.264
λ_P	5.0	5.0	5.0	5.0
R_P	6.924 ± 1.527	8.172 ± 1.299	7.104 ± 1.471	6.798 ± 2.013

Table 7 Cumulative processing reputation parameters for Fig. 7

$\overline{X}_{T,P}$	38.496	50.973	48.011	45.002
$\sigma_{T,P}$	30.558	20.347	17.302	20.454
λ_P	5.0	5.0	5.0	5.0
$R_{T,P}$	6.924 ± 1.527	7.548 ± 1.017	7.4 ± 0.865	7.250 ± 1.022

Thus, we believe that the cost realized is highly reasonable in return of the reputation service provided.

The calculation of the dynamic security reputation score is realized by executing a set of over hundred vulnerability testing attacks on the cloud infrastructure resources. The attack categories ranged from simple password cracking attacks to advanced distributed DOS and buffer overflow attacks. We employed 25 security assessment tools and network scanners for achieving this task. The majority of the security tools used is published in [26].

The complete source code of the cloud prototype implementation is available online at [24].

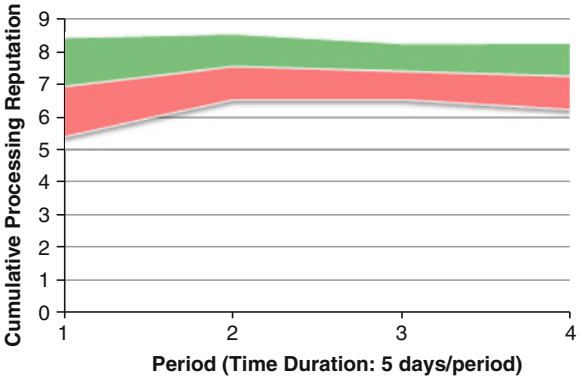
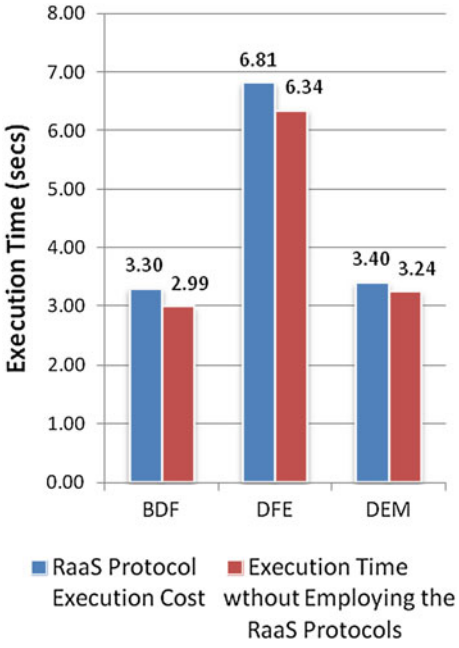


Fig. 7 Cumulative processing performance calculation

Fig. 8 Execution time analysis of the RaaS protocols



7 Economic and Feasibility Analysis

A brief economic study shows that commercial cryptographic coprocessors range in price from several hundred to several thousand U.S. Dollars. The cost of the coprocessor mainly depends on its processing and memory capabilities, the degree of physical security and tamper-resistance supported, its compliance with FIPS standards, and the crypto functionality (Hardware acceleration and cryptographic implementations)

provided. We believe that the cost of the reputation service presented can be greatly reduced based on a set of external economic factors as well as internal design choices related to the RaaS protocols architecture. These factors are summarized in the following points:

1. The increase in demand on cryptographically secure facilities to provide practical security solutions, particularly to computing clouds, will increase the competitiveness in the crypto coprocessor commercial market and will gradually result in a higher functionality/cost ratio.
2. The technological advancements in computing and memory hardware, as well as in physical packaging mechanisms, will result in delivering cost-effective cryptographic coprocessors.
3. The emergence of open-source cryptographic processor designs [8] will support the elimination of monopolies in the coprocessor market, and hence will lead to considerable price reductions.
4. The coprocessor sharing mechanism employed in RaaS (where more than one customer shares the resources of a particular crypto coprocessor) plays a major role in the cost-effectiveness of the security solution by greatly limiting the number of cryptographic processors that needs to be physically installed in the computing cloud.
5. The software division mechanism and the light-weighted nature of the RaaS performance monitoring protocols implemented supports a better utilization of the crypto coprocessor and avoids any unnecessary loads on its resources. This aids in reducing the resource requirements, and hence the price, of the coprocessor. It is true that the software division process adds considerable costs in restructuring legacy applications. However we believe that the reduction in the coprocessor cost resulting from this process outweighs the software partitioning cost. Moreover, we believe that the software industry will be able to accommodate the software partitioning design model using standard patterns and blueprints which will highly reduce the software division costs. From experience in this field we witness how open and flexible is the software development industry to accepting novel software development models. The rapid success of the procedural and Object-oriented development paradigms is a good example to illustrate this point.
6. Cloud computing security research is giving more attention to trusted hardware security approaches to provide technical solutions for solving several data privacy and integrity issues in the computing cloud. This fact is corroborated by the proposed work of the Trusted Computing Group [29] for developing a set of cloud security services and protocols based on their Trusted Platform Module (TPM) [1, 31].

8 Security Discussion

The main axiom we consider in this work is that a secure coprocessor is capable of ensuring the confidentiality and integrity of the data it possesses in its address space. In fact, every security protocol depends on a set of assumptions, which if respected, supports the proper fulfillment of the promised security properties and mechanisms. For instance, when designing cryptographic protocols, it is usually assumed that encryption algorithms are computationally-secure in resisting cryptanalysis and that a successful brute force attack on the key space is highly expensive considering current technological and computational capabilities. Thus, analogously, we believe that it may be possible to violate the physical security protections of a crypto coprocessor but this would require enormous effort and resources that are not currently possessed by attackers. Based on this, we do not associate the protocol designs we present in this work with any available crypto coprocessor design technology, but rather present them using a generic crypto coprocessor model that serves the physical security axiom we initially assumed. In other words, we believe that just as cryptographic algorithms can be evolved and modified to increase, beyond feasibility, the cost required to break them, tamper-proof security packaging technology can be enhanced to increase the cost and effort required to breach the crypto coprocessor physical security mechanisms.

In summary, the accountability of the RaaS protocols is guaranteed by relying on the tamper-proof capabilities of cryptographic coprocessors. This ensures the confidentiality and integrity of the log records even if stored on the nontrusted cloud side. The encryption of the log records ensures their confidentiality by preventing the cloud provider from reading the contents of these records. Moreover, the hash chain structure and its respective MAC values ensure the integrity of the log records by guaranteeing the detection of any unauthorized modification imposed on them.

Although we assume that the trusted entities of the security model (TTP and cryptographic coprocessors) are immune against system and network penetration attacks, we believe that it is beneficial to conclude the security analysis with a brief description of the effects that may be experienced if some of the axioms assumed in the RaaS system model are eradicated. The main security exceptions that may affect the system are described below:

1. Compromise of the TTP or its key K_{TTP}: an attack that compromises the TTP or its cryptographic key K_{TTP} will give the attacker unlimited network access to the different cloud crypto coprocessors configured and distributed by the victim TTP. The consequences of this attack are disastrous on the system since they result in unveiling all the encryption cryptographic keys stored in the cloud crypto coprocessors. This fact demolishes all the security mechanisms employed and leads to the collapse of the whole security architecture.
2. Breaking into a crypto coprocessor: successfully breaking into a particular crypto coprocessor allows the attacker to gain access to the TTP key K_{TTP}. This fact renders this attack a variant of the previously described attack in bullet 1.

3. Capturing a cloud provider site: assuming that the physical security axioms of the crypto coprocessor remain intact, capturing a cloud provider site would not affect the confidentiality and integrity of the reputation feedback records. In fact this is the main objective of the RasS reputation system which is to protect the confidentiality and integrity of the reputation data against any form of unauthorized access from an outside attacker or even from the cloud provider itself.

9 Conclusion and Future Directions

In this chapter we presented RaaS, a secure and accountable reputation service for ranking service providers in cloud computing platforms. RaaS builds on a set of integrity-assurance mechanisms and protocols to provide a secure execution environment for supporting the reputation calculation. The reputation calculation is based on securely-generated log entries at the provider-side. Dedicated light-weight performance evaluation protocols are established to secure the event log generation and storage mechanisms. A prototype implementation of the various RaaS algorithms and protocols is tested on the VMware vSphere cloud computing operating system. The incorporation of the RaaS protocols supporting the accountable reputation service added minimal overhead to the overall system performance.

Future extensions on this work are summarized in the following two points:

1. Devising secure and accountable auditing mechanisms for assessing the energy consumption patterns at cloud provider sites. Such mechanisms are very crucial to incorporate energy consumption reputation scores into the RaaS reputation system. Energy consumption reputation publication can aid in encouraging provider sites to follow green computing IT approaches.
2. Relying on accountable performance, security, and energy reputation scores to provide autonomic service routing protocols in collaborative cloud computing architectures supporting composite services. This aids in leveraging the collaborative cloud service model to maximize the performance and energy efficiency of advertised cloud services and to enhance their security mechanisms.

References

1. Bajikar S (2002) Trusted platform module (TPM)-based security on notebook PCs-white paper. Mobile Platforms Group, Intel Corp
2. Chang E, Dillon T , Hussain FK (2006) Trust and reputation for service-oriented environments. Wiley, London
3. Coveillo A, Elias H, Gelsinger P, Mcaniff R (2011) Proof, not promises: creating the trusted cloud, RSA white paper. http://www.rsa.com/innovation/docs/11319_TVISION_WP_0211.pdf
4. Cusumano M (2010) Cloud computing and SaaS as new computing platforms. Commun ACM 53(4):27

5. Diffie W, van Oorschot PC, Wiener MJ (1992) Authentication and authenticated key exchanges. Des Codes Crypt 2:107–125
6. Foussa F, Achbanyb Y, Saerens M (June 2010) A probabilistic reputation model based on transaction ratings. Elsevier Inf Sci 180:2095–2123
7. Freier A, Karlton P, Kocher P (1996) The SSL protocol version 3.0. Internet-Draft
8. Gutmann P, An open-source cryptographic coprocessor. In: Proceedings of the 9th USENIX security symposium, Denver, Colorado, August 2000, pp 97–112
9. Haeberlen A (2009) A case for the accountable cloud. In: Proceedings of LADIS
10. Health Insurance Portability and Accountability Act homepage: http://www.hipaa.org
11. Hoffman K, Zage D, Nita-Rotaru C (2009) A survey of attack and defense techniques for reputation systems, ACM Comput Surv 42(1)
12. Hwang K, Kulkareni S, Hu Y (2009) Cloud security with virtualized defense and reputation-based trust mangement. DASC'09, pp 717–722
13. Itani W, Ghali C, Kayssi A, Chehab A (2011) Accountable reputation ranking schemes for service providers in cloud computing. In: Proceedings of the 1st international conference on cloud computing and services science, CLOSER 2011, Noordwijkerhaut, The Netherlands, 7–9 May 2011
14. Itani W, Kayssi A, Chehab A, Privacy as a service: privacy-aware data storage and processing in cloud computing architectures. In: proceedings of the eighth IEEE international conference on dependable, autonomic and secure, computing, pp 711–716
15. Janger E, Schwartz P (2002) The Gramm-Leach-Bliley Act, information privacy, and the limits of default rules. Minn L Rev 86:1219–1261
16. Jøsang A, Ismail R, Boyd C (2007) A survey of trust and reputation systems for online service provision. Decis Support Syst 43(2):618–644
17. Li A, Yang X, Kandula S, Zhang M (2010) CloudCmp: shopping for a cloud made easy. In: Proceedings of the 2nd USENIX conference on hot topics in cloud, computing (HotCloud'10)
18. Lim S, Keung C, Griffiths N (2010) Trust and reputation. In: Agent-based service-oriented computing. Springer, London, pp 189–224
19. Malik Z, Bouguettaya A (2009) RateWeb: reputation assessment for trust establishment among web services. VLDB J 18(4):885–911
20. Mármol F, Pérez G (2009) Security threats scenarios in trust and reputation models for distributed systems. Comput Secur 28(7):545–556
21. Nepal S, Malik Z, Bouguettaya A (2011) Reputation management for composite services in service-oriented systems. Int J Web Service Res 8(2):29–52
22. Pearson S, Charlesworth A (2009) Accountability as a way forward for privacy protection in the cloud. HP labs technical report, HPL-2009-178. http://www.hpl.hp.com/techreports/2009/HPL-2009-178.pdf
23. Wainewright P SaaS will dominate your cloud strategy, Zdnet News, retrieved from: http:// http://www.zdnet.com/blog/saas/saas-will-dominate-your-cloud-strategy/1300
24. RaaS Prototype Implementation, Available online at https://www.dropbox.com/s/84l4skh89n08w49/RaaS_Impl.zip
25. Schneier B, Kelsey J (1999) Secure audit logs to support computer forensics. ACM Trans Inf Syst Secur 2(2):159–196
26. The Insecure.org website http://Insecure.org
27. The Nessus network scanner homepage http://www.nessus.org
28. The Nmap tool homepage http://www.nmap.org
29. The Trusted Computing Group homepage http://www.trustedcomputinggroup.org/
30. The vSphere 4 home page http://www.vmware.com/products/vsphere
31. Trusted Computing Group (2010) Expanded IF-MAP 2.0 addresses a broader set of applications, white paper
32. Wang Y, Vassileva J (2007) Toward trust and reputation based web service selection: a survey. Int Trans Syst Sci Appl J Spec Issue New Tendencies Web Serv Multiagent Syst 3(2):118–132
33. Weingart S (1987) Physical security for the mABYSS system. In: Proceedings of the IEEE computer society conference on security and privacy, pp 52–58

34. Wieder P, Butler JM, Theilmann W, Yahyapour R (2011) Service level agreements for cloud computing, Springer, Berlin/Heidelberg
35. Wort S, Bolton C, Langford J, Cape M, Jin JJ, Hinson D, Ji H, Mestemaker PA, Sen A (2008) Professional SQL server 2005 performance tuning, Wrox
36. Yahyaoui H, Maamar Z, Bentahar J, Sahli N, Elnaffar S, Thiran P (2008) On the reputation of communities of web services. In: International conference on new technologies in distributed systems, pp 1–8
37. Yao J, Chen S, Wang C, Levy D, Zic J (2010) Accountability as a service for the cloud. In: Proceedings of the IEEE international conference on services computing (SCC), Miami, USA
38. Yee BS, Tygar JD (1995) Secure coprocessors in electronic commerce applications. In: Proceedings of the 1st USENIX workshop on E-Commerce

Combating Cyber Attacks in Cloud Systems Using Machine Learning

Md Tanzim Khorshed, A. B. M. Shawkat Ali and Saleh A. Wasimi

1 Introduction

One of the crucial but complicated tasks is to detect cyber attacks and their types in any IT networking environment including recent uptake of cloud services. The common business practice of existing cloud providers is that they are not transparent when it comes to share security related logs and data with its consumers, which adds to the difficulty of detection by a cloud customer. The issue is addressed in this chapter in two parts. First, we demonstrate an easy technique on how cloud customers can collect performance data from their Virtual Machine (VM). Second, some thoughts are constructed on novel approaches to classify some of the widely discussed cyber attack types using machine learning techniques. We evaluate the techniques' performances using accuracy measure. The novelty of this rather rigorous analysis is in its ability to identify insider's activities and other cyber attacks using performance data. The reason for using performance data rather than traditional logs and security related data is that the performance data can be collected by the customers themselves without any assistance from the cloud providers. Therefore the aim of these series of experiments in a constructed cloud system is to give researchers, cloud providers and consumers additional insight and tools to proactively protect their data from known, or perhaps even unknown, security issues that have similar patterns.

The ultimate design objective of this chapter is to build a "Proactive Attack Detection" model for cloud computing users with three goals. Firstly, the model will be able to detect an attack when it starts or at least during the time of its perpetuation. Secondly, it can alert system/security administrators and data owner about the attack type with possible action needed. Thirdly, if cloud providers try to hide attack information from customers, this model will be able to tell customers about the kind of attack that happened by looking at the pattern of attack.

M. T. Khorshed (✉) · A. B. M. S. Ali · S. A. Wasimi
Central Queensland University, QLD 4702, Australia

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_14,

The chapter is organized as follows: Sect. 2 discusses some background information and related work for detecting cyber attacks in cloud computing using performance data. Section 3 describes the key concepts of cyber attack detection in cloud computing using machine learning techniques that includes experimental design and data collection. Section 4 narrates different classification algorithms that can be used for attack detection in cloud computing. Section 5 shows our experimental results, and finally, in Sect. 6 we summarise the limitations of present work and put forward the scope for future work.

2 Background and Related Work

As with any change in IT infrastructure where there are accompanying novel risks and opportunities, cloud computing is no exception. Shared, on-demand nature of cloud computing expose it to some unique risks that have not been experienced before. Cloud computing inherits all the security issues from existing systems, for instance grid computing, plus the security issues that have been created due to its unique architecture and features [22]. Despite our awareness on threats and our efforts to tackle them, cyber attacks are not vanquished, and we believe this is due mainly to the gaps. Rimal et al. [31] presented eight examples of outages in different cloud services with date and duration. Among those outages there are some providers of today's leading cloud services including Microsoft Azure and Google apps. Dahbur et al. [8] presented three other scenarios of cloud computing outage and data loss with the number of customers affected. It is not clear whether these outages were caused by attacks, but nevertheless, outages and data losses are surely basic security concerns and can be put into Cloud Security Association (CSA)'s data loss/leakage threat category [2]. Researchers at the University of California, San Diego and the Massachusetts Institute of Technology, Cambridge, and Ristenpart et al. [32] showed experimentally with Amazon Elastic Compute Cloud [1] that it is possible to map the internal cloud infrastructure and find out the location of a particular VM. They also showed how such findings can be used to mount cross-VM side-channel attacks to collect information from a target VM residing on the same physical machine. A recent research [33] showed how malicious insiders can steal confidential data. They demonstrated a set of attacks with attack videos, showing how easily an insider can obtain passwords, cryptographic keys and files etc. Chonka et al. [5] recreate some of the recent real world attack scenarios and demonstrate how HTTP-DoS and XML-DoS attack can take place in cloud computing.

In this chapter we only focus on one of the most hostile cyber attack types known as Denial of Services (DoS) and sometimes Distributed Denial of Services (DDoS). According to the United States Computer Emergency Readiness Team (US-CERT) DoS attack is a type of attack where an attacker attempts to prevent legitimate users from accessing network or computer resources. DDoS means, the attacker is using multiple computers to launch the denial-of-service attack [25]. However, there are several other types of DoS attacks and attack tools which are worth testing in an

experimental cloud environment. US-CERT [25] also listed few symptoms of DoS and DDoS attacks such as unusually slow network performance, unavailability of a particular website, inability to access any website, dramatic increase in the amount of spam, etc. We discovered some commonalities among these [5, 32, 33] attack models in that all of them used attack tools and followed organized attack procedures. We have attempted to design our experimental setup in the same pattern.

In some recent research works, cloud providers' reluctance to supply different security related data to its consumers has been revealed. This is a recurring pattern and does not appear to have an easy resolution as no one wishes to disclose their company secrets together with policy for hiring recruits [19–22]. We also note that some remediation measures are only initiated after a successful attack happens. Therefore, this chapter intends to inform cloud customers of some necessary ideas on how they can sense diverse types of cyber attacks with limited resources and access they have. In a related work Khorshed, et al showed that real life insiders' activities can be detected from the performance data in a hypervisor and its guest operating systems [19]. They also demonstrate that it is possible to detect DoS and DDoS based attacks on cloud computing using performance data which is generated in the hypervisor and its guest operating systems [21].

Through the above research works, the performances of several of the most popular machine learning techniques were used in a cloud environment for identification of these attacks and activities. A comparison on performances has also been made in identifying a particular attack or activity. A technique's performance has been evaluated through different performance evaluation matrices with the rigorous testing of 10-fold cross validation. The experimental outcome demonstrated that C4.5 [28] provided not only a better performance than other techniques, but also the level of performance is of acceptable standard. The other algorithms tested were Naive Bayes [18], Multilayer Perceptron [24], Support Vector Machine (SVM) [27] and PART [12].

The utility of above research works is that they are capable of recognizing insider's actions and further DoS/DDoS attacks with performance data. The uniqueness of this chapter is the use of performance data rather than the conventional logs and security related data with the distinct advantage that the performance data can be self-possessed by the customers without any help from cloud service providers.

3 Key Concepts/Technologies

Machine learning techniques can be used to investigate if there is an attack. If there is a known type of attack, supervised machine learning techniques can take proactive action to address the issue, and at the same time, notify systems/security administrators as well as the data owner. If an unknown type of attack happens, machine learning will still be able to detect it as an attack analyzing the performance variations from standard usage, and can notify the designated person with the closest type attack known to its database. That would make the security administrator's job

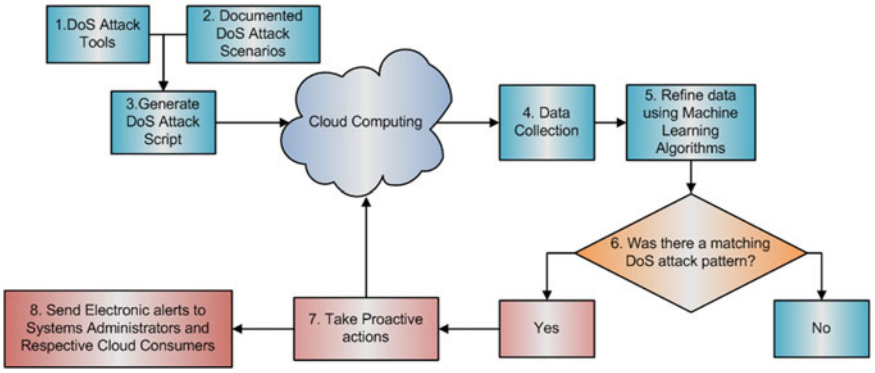


Fig. 1 Experimental design for attack generation, data collection and proactive actions for a DoS/DDoS attack that match attack pattern

easier to fight against unknown types of attacks. A few previous research works by the authors successfully identified Cloud insiders activities [19] and DoS/DDoS attacks [21] using machine learning techniques. These researches found that the rule based technique C4.5 is an efficient technique to solve the problem at hand. They evaluated a technique's performance through different performance evaluation matrices with the rigorous testing of 10-fold cross validation. The experimental outcome demonstrated that C4.5 provided not only a better performance than other techniques, but also the level of performance is of acceptable standard. The other algorithms tested were Naive Bayes, Multilayer Perceptron, SVM and PART [19, 21].

3.1 Experimental Design

For this experiment we first chose some of the very common DoS/DDoS attack tools, our aim was to train the machine with necessary patterns of DoS/DDoS attacks, we discuss about these basic tools in data collection section. We also studied some of the real world documented attack scenarios, such as [9–11, 16] then planned and generated attack script accordingly. Figure 1 shows our experiment designs. It should be noted that for complex environments, we recommend running step 1 and 2 first, train the machine and then run step 3 to get enhanced result.

Once attack script ran on Cloud Computing environment, we have collected Virtual Machine Managers (VMM) data in step 4. At step 5 machine learning algorithms were used to detect a DoS/DDoS attack type.

For this experiment we have chosen a HP ProLiant DL380 G4 Server [6] with Dual network interface cards. The main reason for selecting server hardware is not to make hardware limitation a bottleneck, which may provide inaccurate data. We also chose VMW are ESXi 3.5 [37] Hypervisor as VMM and Windows 7 [7] as

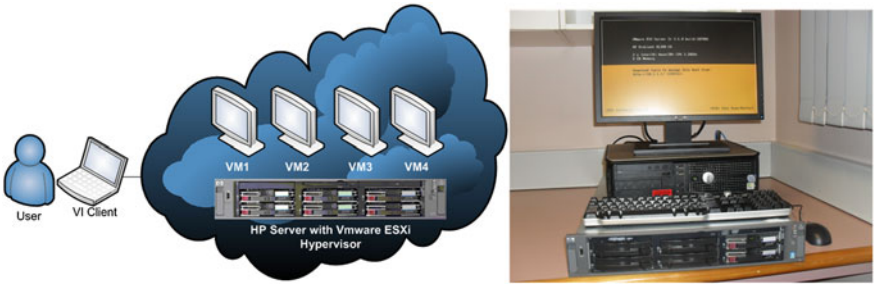


Fig. 2 Logical and physical diagram of our experiment design

guest Operating System (OS). Figure 2 shows a logical and physical diagram of our experiment design.

3.2 Data Collection

In this section we present pictures of the performance charts that are taken from the hypervisor during the attack and also performance plot generated by our data collection spreadsheet during the attack, the similarity between these two indicates how precisely data was collected. Accurate data collection is very important to achieve correct results by machine learning, and also, to make a distinction between an attack and an activity. It is to be noted that we have collected performance data of 20 different parameters of System, CPU, Memory and network. In this chapter we only included performance charts or plots of those that shows significant changes during an attack, however, to refine data using machine learning we used all 20 parameters at the same time irrespective of whether they made any noteworthy dissimilarity or not.

3.2.1 DoS Using Real-Time Disk Operating System

Real-Time Disk Operating System (RDoS) by Rixer [35] is one of the most easily available DDoS attack tool for web attack. This tool together with a port scanner can be very useful DDoS attacking tools for web resources. In our experiment we only used RDoS and did not use any port scanner as we created our own website on a virtual cloud environment and knew the port number already, which in this case was default HTTP port 80. Our HTTP servers Internet Protocol (IP) address is 10.1.1.1 and we ran RDoS tool from other VMs selecting victim's IP address 10.1.1.1 and port 80 from 4:30 to 4:40 a.m.

Figure 3 shows RDoS by Rixer tool running in our cloud environment and also victim's System performance chart showing important changes during the attack,

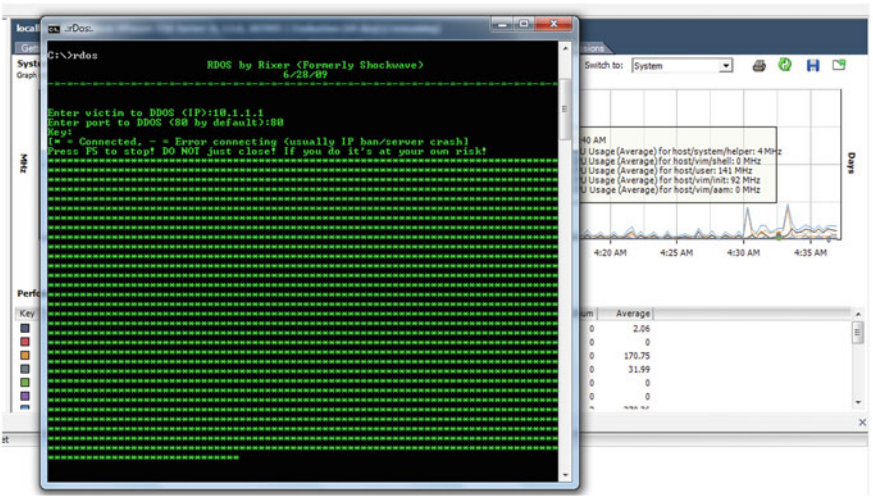


Fig. 3 DoS attack using RDoS running also victims system performance chart showing significant changes during the attack [21]

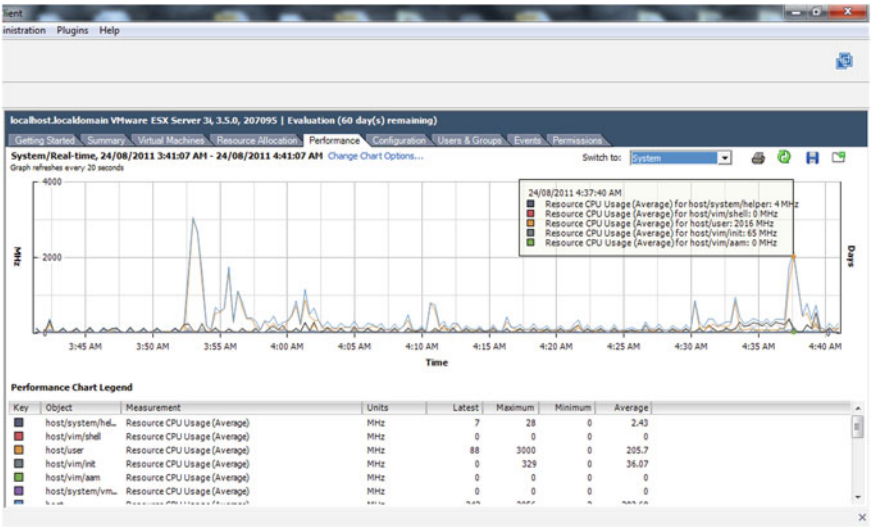


Fig. 4 System performance chart during RDoS attack happened between 4:30 and 4:40 a.m.

there is a notable change in performance chart from 4:30 a.m. onwards since we ran this tool.

Figures 4 and 5 represent system and CPU performance charts, respectively, after the attack happened between 4:30 and 4:40 a.m., significant changes in both System and CPU performances are noticeable during attack time.

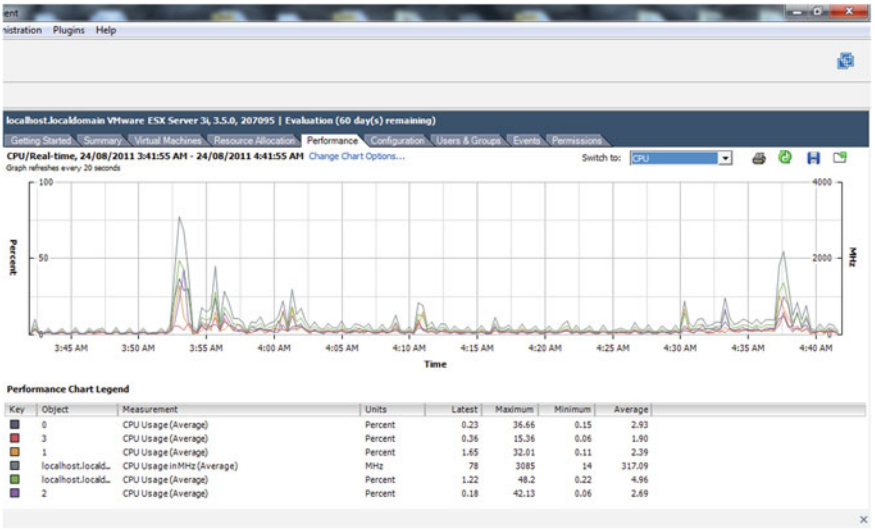


Fig. 5 CPU performance chart generated in hypervisor during the attack

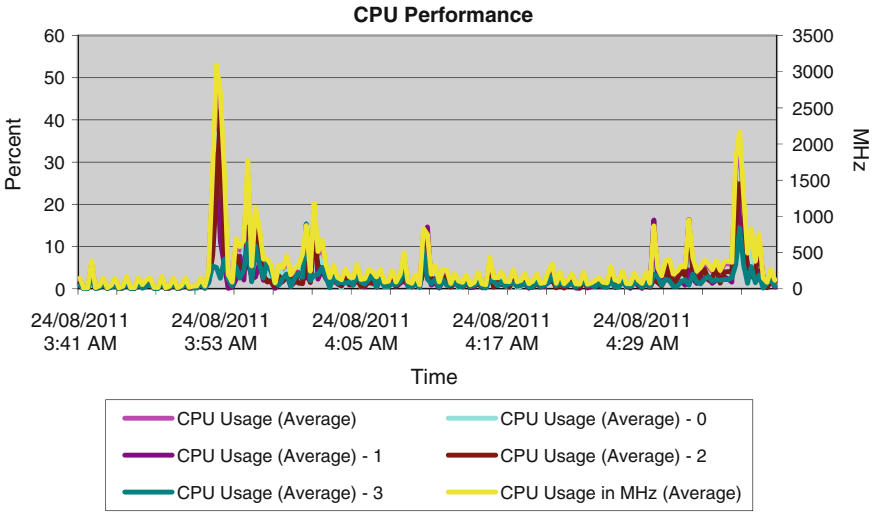


Fig. 6 CPU performance plot generated by data collected for our experiment showing similarity with the one generated automatically by the hypervisor

Next four pictures we have presented here are the plots taken from our data collection spreadsheet, these are exactly same as the hypervisor was showing in its performance charts during the time the attack happened, which indicates how accurately data can be collected in a cloud environment. Figures 6, 7, 8 and 9 show

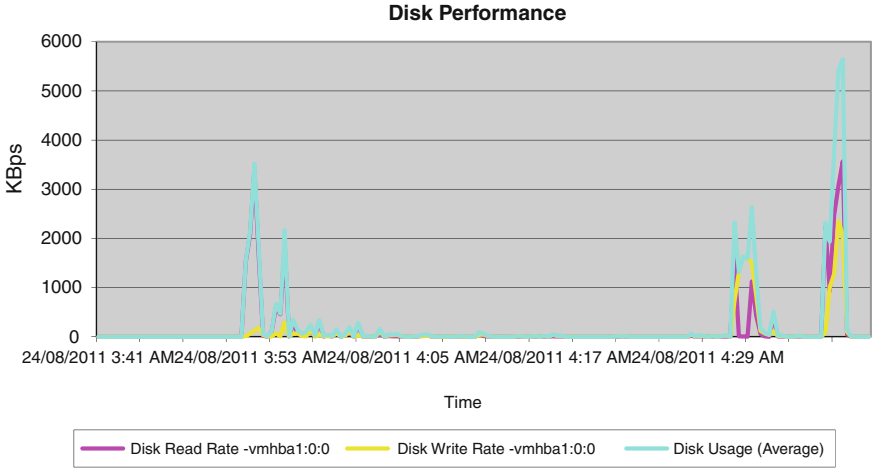


Fig. 7 Performance plot disk performance from the data collection spreadsheet

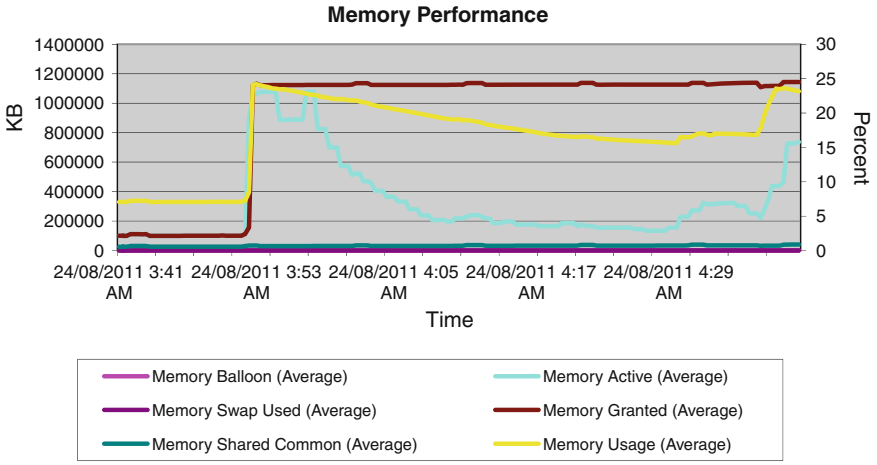


Fig. 8 Memory performance plot shows significant changes in active memory and memory changes during attack time

the performance plots of CPU, Disk, Memory and System, respectively, generated by our data collection spreadsheet.

3.2.2 HTTP-DoS Attack Using Low Orbit Ion Cannon

Low Orbit Ion Cannon (LOIC) is an open source network stress testing and DoS/DDoS attack application [3, 17]. An attacker can flood TCP/UDP packets with the intention of disrupting the service of a particular host. On December 2010, BBC

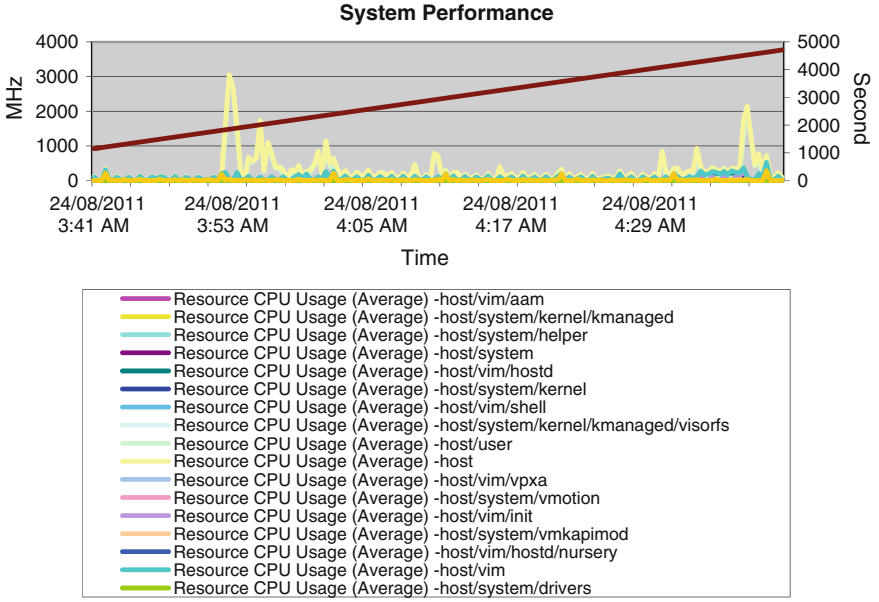


Fig. 9 System performance plot generated by data collected

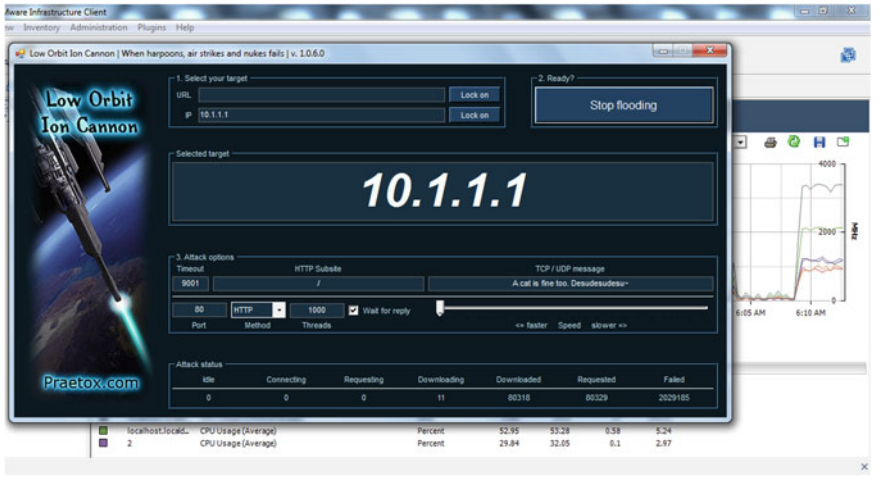


Fig. 10 HTTP-DoS attack using LOIC running, also showing CPU performance chart generated in hypervisor [21]

report entitled "Anonymous Wikileaks supporters explain web attacks" quoted security experts that well-written firewall rules can filter out most traffic from harmful DDoS attacks by LOIC [4]. However, in our previous research we discovered that these corporate firewalls are not very effective if the attacker resides or shares the

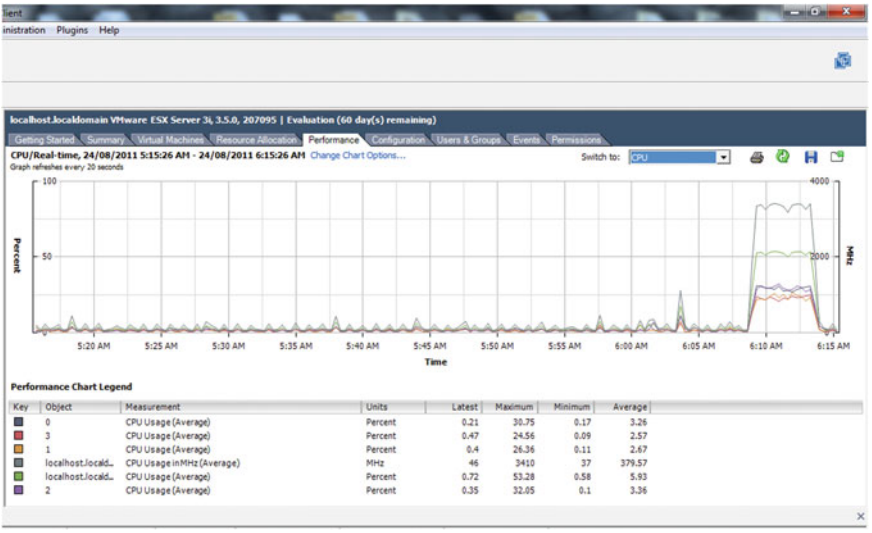


Fig. 11 CPU performance chart generated in hypervisor during attack showing significant changes during attack time

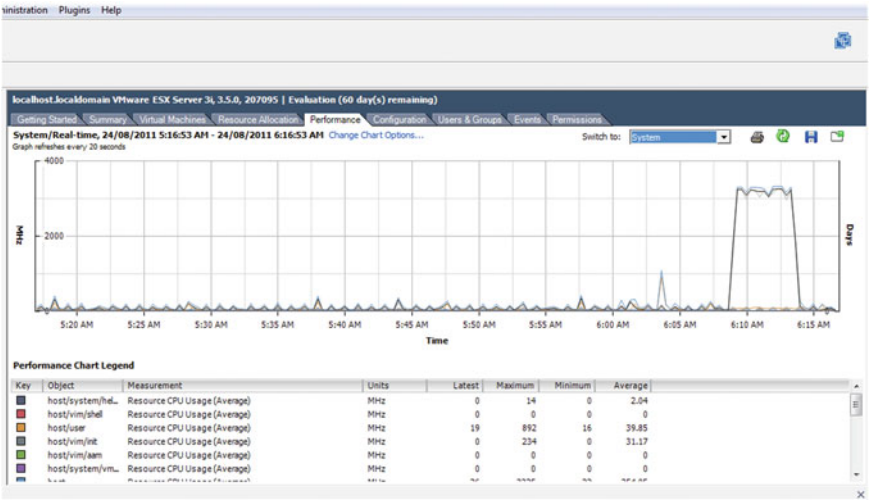


Fig. 12 System performance chart generated in hypervisor during HTTP-DoS attack

same physical hardware from same cloud provider [22]. For that reason, here, we attack a particular VM from other VMs that is sharing the same physical resources. We started HTTP-DoS attack on victim (IP 10.1.1.1) using LOIC at 6:08 a.m. and ended at 6:15 a.m. Figure 10 shows LOIC running from attacker VM with target IP 10.1.1.1.

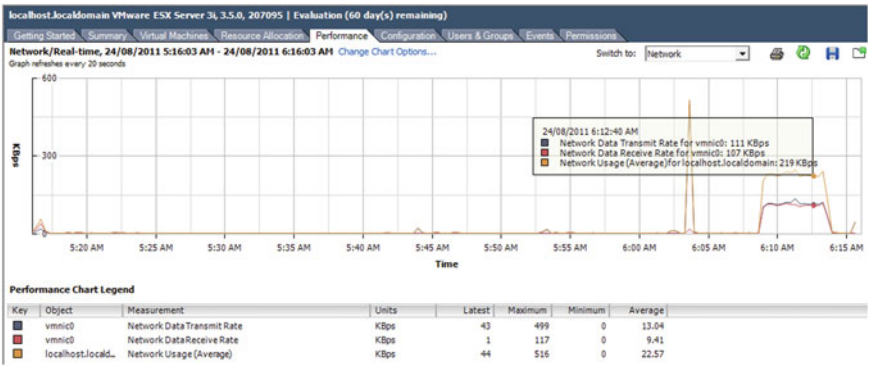


Fig. 13 Network performance chart generated in hypervisor during HTTP-DoS attack

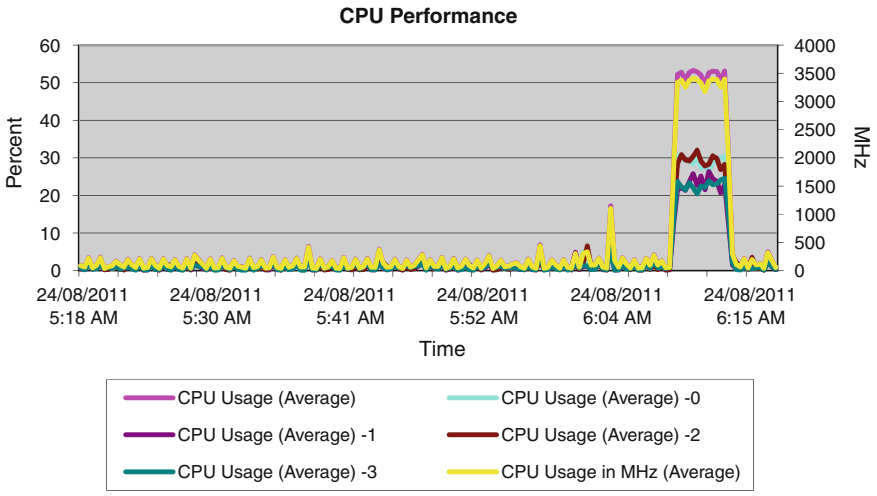


Fig. 14 CPU performance plot generated from the data collected [22]

Figures 11, 12 and 13 show the performance charts (taken from hypervisor) of CPU, system and network during the attack. Significant changes in charts are noticeable from 6:08 to 6:15 a.m. when the attack happened.

In Figs. 14, 15 and 16 we present the performance plots of CPU, network and system generated by our data collection spreadsheet that we collected from the VMM during the attack. Sudden increase in performance was noticed during the attack time (started at 6:08 a.m. and ended at 6:15 a.m.)

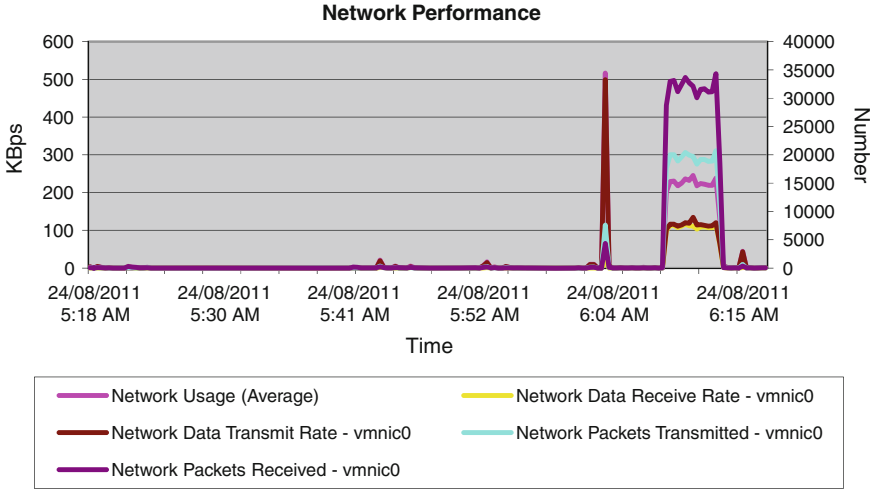


Fig. 15 Network performance plot generated from the collected data [22]

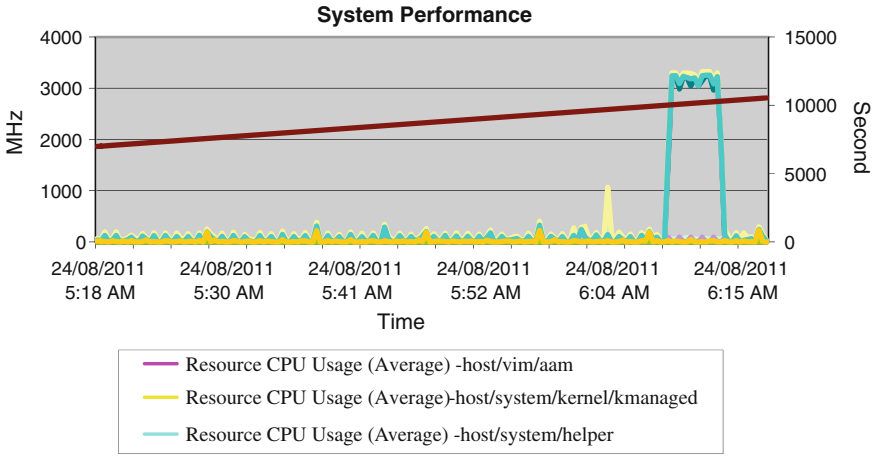


Fig. 16 System Performance plot generated from the data collected

3.2.3 Ping Flood Attack

Ping flood is another kind of DoS/DDoS attack where the attacker crushes the victim with Internet Control Message Protocol (ICMP) Echo Request (ping) packets. This method could be very successful when sending packets quickly without waiting for a response from the victim. If ICMP service is not disabled by the target host, it will flood the target host with large data segments [15, 26]. However, in our study and from work experience, we found organizations usually disable ICMP requests at firewall or in the router so that it can stop ICMP requests from external networks,

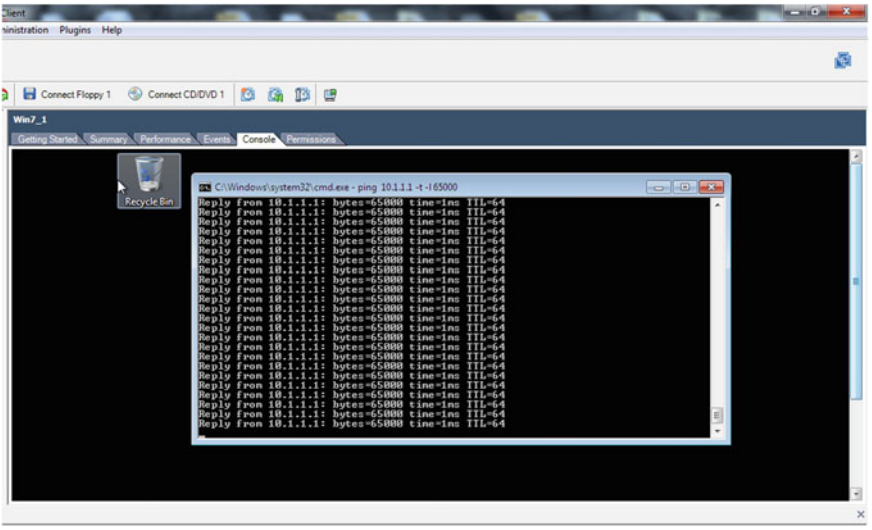


Fig. 17 From hypervisor console running ping flood attack from attacker VM to victim VM

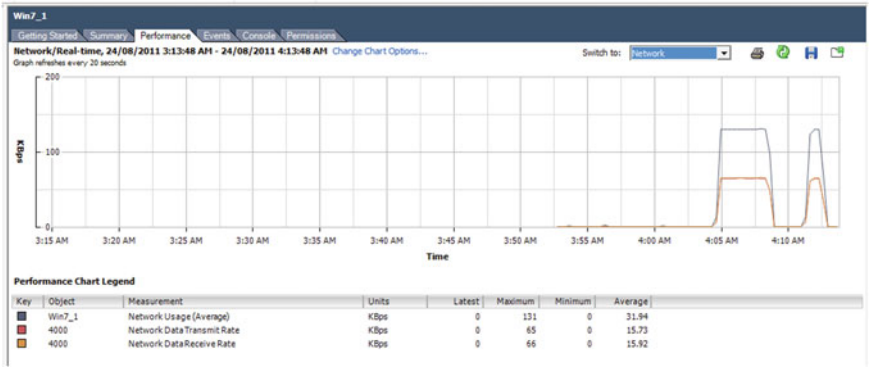


Fig. 18 Network performance chart generated from the hypervisor

but traditionally they keep ICMP open on hosts in their own internal networks, so that they can do network diagnostics. Our concern for cloud VMs is that an attacker could be residing on the same physical hardware or somehow can manage to hack into another low secured VM that is residing on same internal virtual network and, carry this kind of attack to a target VM.

A certain kind of ping flood attack in the past was named “ping of death” where an attacker deliberately used to send packets larger than the 65,536 bytes, many computer systems were not able to handle a ping packet larger than this maximum IPv4 packet size [34]. In our experiment we send ICMP packets from each attacker VM slightly lower than that, so that attacker VM itself does not get overwhelmed.

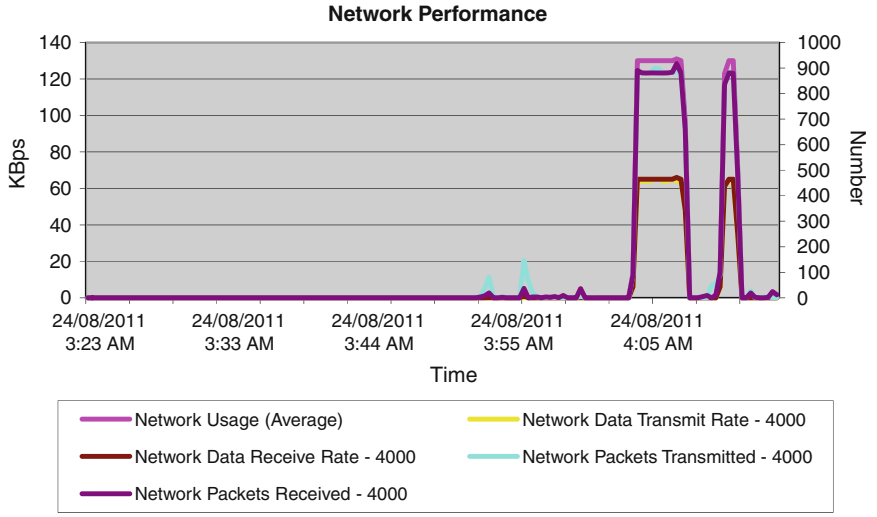


Fig. 19 Network performance plot generated from collected data

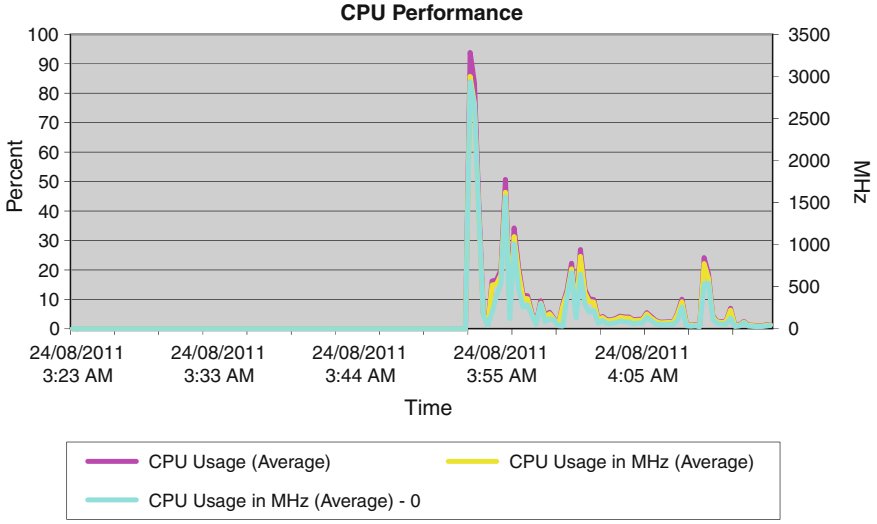


Fig. 20 CPU performance during attack time, and nothing significant noted

We ran "ping 10.1.1.1 –t -l 65000" command from each attacker VM. Here –t was used for repeated sending of echo messages and –l indicates the size of packet to be sent, in this case it was 65,000 bytes from attacker VM1 (we named it win7_1 as shown in Fig. 17).

Figure 17 shows hypervisor console running ping flood attack from attacker VM to Victim VM. Whereas Fig. 18 represents network performance chart of attack,

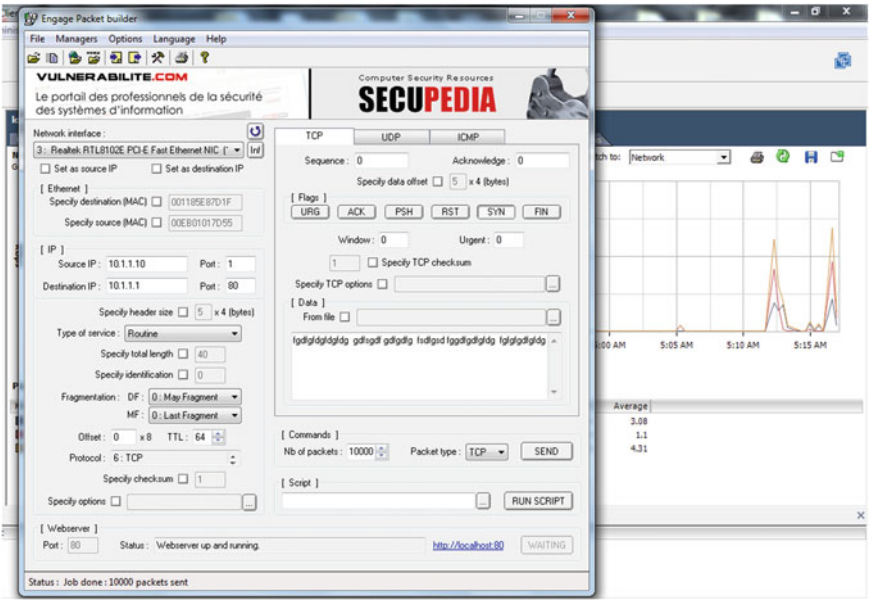


Fig. 21 Running SYN flood attack using engage packet builder and monitoring network performance from the hypervisor during the attack

Figs. 19 and 20 are the performance plots of network and CPU generated during the data collection that we capture from the VMM during the attack.

3.2.4 SYN Flood Attack Using Engage Packet Builder

A SYN flood attack is also another kind of DoS/DDoS attack where a network becomes overwhelmed by a series of SYN requests to a target's system [23]. Engage Packet Builder [39] is a powerful and scriptable packet builder with capability of packet injection starting from link layer (MAC address spoofing), it can also generate SYN-Floods by building "strange" packets [39]. We used Engage Packet builder to run SYN flood attack twice, at 5:13 a.m. and 5.17 a.m., Fig. 21 shows Engage Packet Builder running from attacker VM (IP 10.1.1.10) with target IP 10.1.1.1, Fig. 22 represents a performance chart (taken from hypervisor) of network during the attack. Figures 23, 24, 25 and 26 are performance plots of CPU, memory, network and system generated by our data collection that we collected from the VMM during the attack.

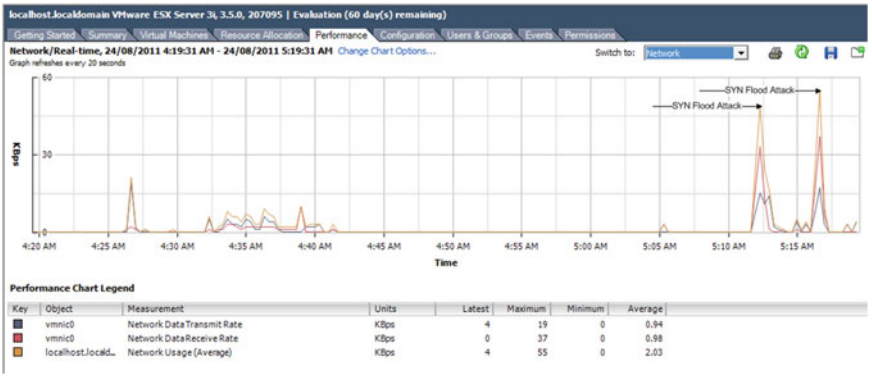


Fig. 22 Network performance chart collected from hypervisor after the SYN flood attack was over [22]

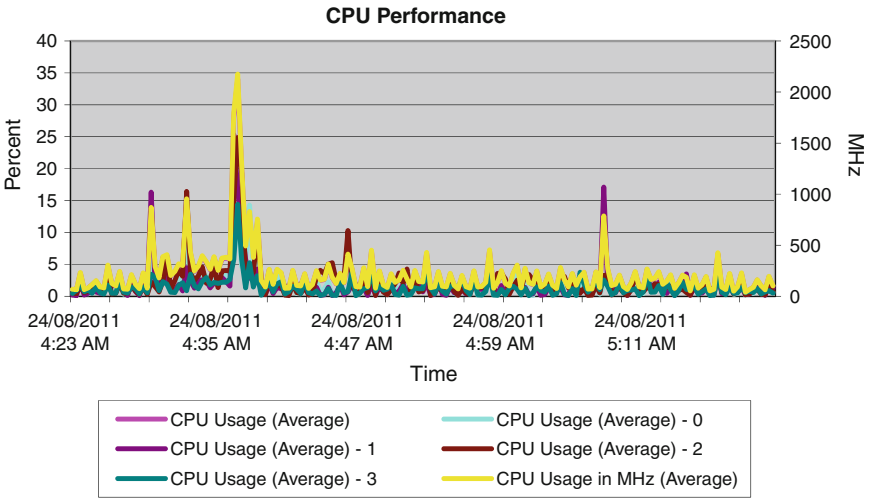


Fig. 23 CPU performance plot generated from the data collected during the SYN flood attacks

4 Classification Algorithms

Now-a-days, there are a series of algorithms available to do any classification task by meeting the desired accuracy. Among these, Boosting is a general and popular method to improve the classification accuracies of any weak learning algorithm [13]. In our implementation we used decision tree C4.5 (WEKA name is J48) [28] as a weak learner. One of the strong points of the Boosting algorithm is that one can identify an upper limit of the training error under the assumption of the weak hypothesis, i.e., "rules of thumb". The AdaBoost (adaptive boosting) algorithm was first proposed in 1995 by Yoav Freund and Robert Shapire as a general method for generating a

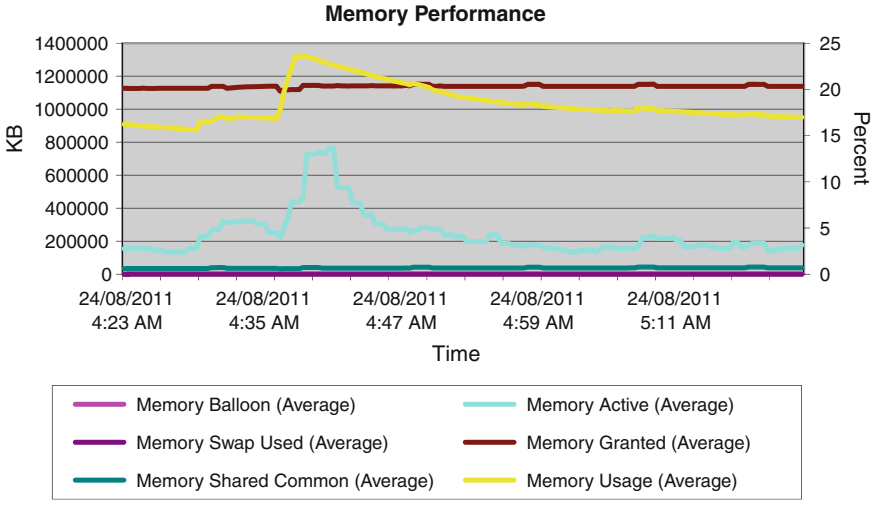


Fig. 24 Memory performance plot generated from the data collected during the SYN flood attacks

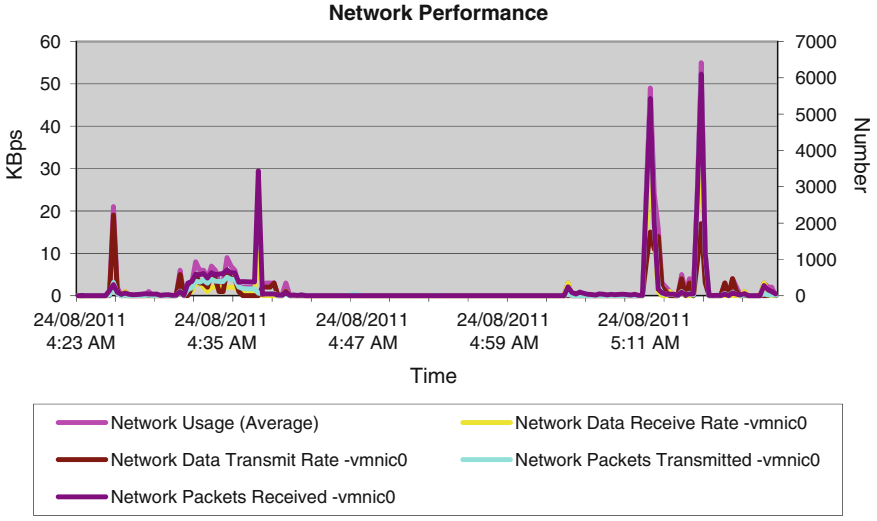


Fig. 25 Network performance plot generated from the data collected during the SYN flood attacks

strong classifier out of a set of weak classifiers [13, 14]. AdaBoost works even when the classifiers come from a continuum of potential classifiers (such as decision tree, linear discriminants, etc.).

Let us consider a DoS attacks dataset which contains N instances and M attributes (Steps 1 and 2 in Fig. 27). Basically the Mth attribute is holding the class variable of

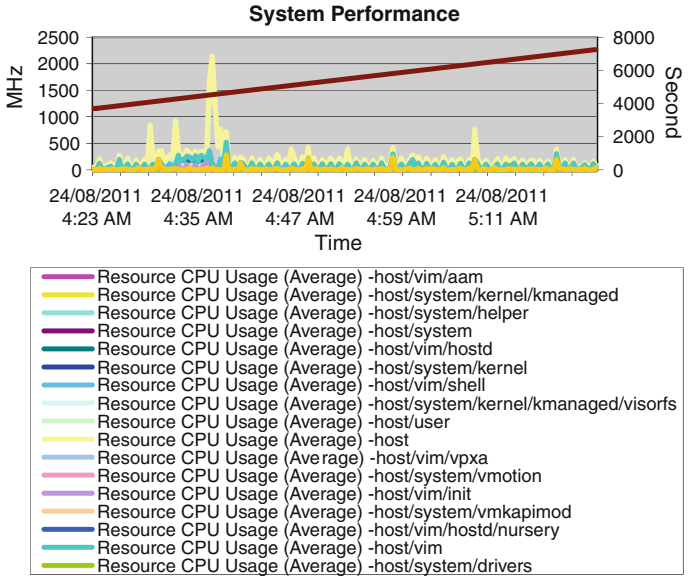


Fig. 26 System performance plot generated from the data collected during the SYN flood attacks

the Dos attack dataset. The independent attributes (N by $M-1$) is denoted by x in step 3 and dependent attribute Mth is indicated by y in step 4 below. We have presented the dependent attribute with the two symbols (+1 means attack and -1 means no attack). In step 5, we have assigned each instance with the same weight, $W_i = \frac{1}{N}$. In our implementation, the decision tree classifier uses the weights associated with each instance. In step 7, the model is prediction, the class values either is +1 or -1. For an ideal model, we should have $M_1(x_i) = y_i$. Of course the model is expected to be only slightly better than random so ms in step 8 is unlikely to be empty. The model M_1 relative error $\in_1$ is calculated as the relative sum of the weights of the misclassified instances in step 9. We use α_i to adjust the weights in step 11. To reach the expected level of classification accuracy, we adjust the weights by increasing or decreasing in the steps 11–13. We continue the model building and weight modification until the new model performs no better than random (i.e., the error is 50 % or more: $\in_1 \geq 0.5$) or is perfect (i.e., the error rate is 0 % and ms in step 8 is empty), or when model reach the maximum number of iterations. Finally, in step 14, the model M combines the other models using a weighted sum of the outputs of these models. In the same step, $\propto_j$ reflects the accuracy of each of the constituent models. The pseudo code of the AdaBoost algorithm is presented in Fig. 27.

A decision tree essentially uses 'divide and conquer' technique to break down a complex decision making process into a collection of simpler decisions, thereby providing an easily interpretable solution [29, 30]. The interesting thing of decision tree modelling is that it is transparent and any one can see the tree structure easily to check how the decision is made [29]. It is a predictive modelling technique used

Algorithm: AdaBoost

AdaBoost (data,learner)

1: $N \leftarrow nrow\ (data)$
2: $M \leftarrow ncol\ (data)$
3: $x \leftarrow data[,1: M-1]$
4: $y \leftarrow data[,M]$
5: **for** $i \leftarrow 1$ **to** N: $w_i = \frac{1}{N}$
6: Repeat $i \leftarrow 1, i \leftarrow i+1$:
7: $M_i \leftarrow learner(data, w)$
8: $ms = \{p\ \mathrm{IM}_i(x_p) \neq y_p\}$
9: $\in_i = \frac{\sum \mathrm{i} \in \mathrm{mn}\ \mathrm{w_j}}{\sum_{j=1}^{n} \mathrm{w_j}}$
10: $a_i = \log\ ((1-\in_i)/\in_i)$
11: **for** $j \in ms: w_j = w_j \times e^{\alpha_i}$
12: **for** $i \leftarrow 1$ ***to*** $N: w_i = \frac{\mathrm{w_i}}{\sum_{j=1}^{n} \mathrm{w_j}}$
13: **until** $\in_i >= 0.5\ or\ ms = \emptyset$
14: Return $[M(x) =$ **sign** $(\sum_{j=1}^{T} \alpha_j M_j\ (x))]$

Fig. 27 Pseudocode of AdaBoost algorithm [38]

widely in classification. In a decision tree, the root and each internal node is labelled with a query. At the terminating nodes, we can see the desired classification outcome in the leaf. The pseudo code of the decision tree algorithm is presented in Fig. 28. In this algorithm *E* and *F* indicate the training records and attribute set, respectively. In Step 7, the algorithm works by recursively selecting the best attribute to split the data and expanding the leaf nodes of the tree in Steps 11 and 12 until the stopping criterion is met in Step 1.

In Step 2, the creatNode() function basically extends the decision tree after creating a new node. The Classify() assigns the class level at the leaf of the tree in Step 3. The find_best_split function determines the attribute to fit in the root to generate a decision tree. In general, we use entropy/gain index to select the attribute for the root.

The entropy or expected information based on the partitioning into subsets by A is given by the equation [2]:

$$E(S) = -\sum_{j=1}^{n} f_S(j) log_2 f s(j) \quad (1)$$

where:

- $E(S)$ is the information entropy of the subset S;

Algorithm: Decision Tree

```
TreeGrowth (E, F)
1: if stopping_cond(E,F) = true then
2:   leaf = createNode().
3:   leaf.label = Classify(E).
4:   return leaf.
5: else
6:   root = createNode().
7:   root.test_cond = find_best_split(E, F).
8:   let V = {v|v is a possible outcome of root.test_cond }.
9:   for each v ∈ V do
10:    Ev = {e | root.test_cond(e) = v and e ∈ E}.
11:    child = TreeGrowth(Ev, F).
12:    add child as descendent of root and label the edge (root → child) as v.
13:  end for
14: end if
15: return root.
```

Fig. 28 Pseudocode of the decision tree algorithm [36]

- n is the number of different values of the attribute in S (entropy is computed for one chosen attribute);
- $f_S(j)$ is the frequency (proportion) of the value j in the subset S;
- $\log_2$ is the binary logarithm.

An entropy of 0 identifies a perfectly classified subset while 1 shows a totally random composition.

Entropy is used to determine which node to split next in the algorithm, the higher the entropy, the higher the potential to improve the classification.

The encoding information that would be gained by branching on A is given by:

$$G(S, A) = E(S) - \sum_{i=1}^{m} f_S(A_i) E(S_{A_i}) \tag{2}$$

where:

- $G(S, A)$ is the gain of the subset S after a split over the A attribute
- $E(S)$ is the information entropy of the subset S
- m is the number of different values of the attribute A in S
- $f_S(A_i)$ is the frequency (proportion) of the items possessing A_i as value for A in S
- A_i is ith possible value of A
- S_{A_i} is a subset of S containing all items where the value of A is A_i.

Gain quantifies the entropy improvement by splitting over an attribute: higher is better. The algorithm computes the information gain of each attribute to construct the final decision tree [29].

The stopping_cond() function is used to stop the profuse expansion of the decision tree. However, in the final stage we use tree pruning method to keep the tree size manageable.

5 Experimental Results

It is important to note that for each attack type different set of parameters of the computer/network system may change—we collected the data on what parameters are changing compared with the usual/average usage. We consider 20 attributes to construct the dataset. These are different parameters of CPU, disk, network and memory performances. In practice, all the data points are considered as real values. The total number of instances in our dataset is 536.

In our previous experiment with DoS attack data set, we had chosen Naive Bayes (NB) [18], Multilayer Perceptron (MP) [24], Support Vector Machine (SVM) [27], Decision Tree (C4.5) [28], and PART [12] to classify our data. Naive Bayes is a probability based technique, Multilayer Perceptron and Support Vector Machine are function estimation based techniques, Decision Tree and PART are rules based machine learning techniques. All these techniques have been implemented in WEKA [40], which is a Java based popular machine learning tool. WEKA uses C4.5 . [40] algorithm for decision tree implementation with a WEKA brand name J48.

We previously suggested J48 in cloud system to identify DoS attacks. However, in this research we noticed AdaBoost with J48 (which is WEKA implementation of C4.5) and AdaBoost with PART displayed significantly better accuracies, 95.5224

Table 1 Classification performances of DoS attack data

	Naive Bayes	Multilayer Perceptron	Support Vector Machine	Decision Tree J48	AdaBoost with J48	PART	AdaBoost with PART
Classification accuracy in %	75.00	92.53	82.08	93.47	**95.5224**	93.28	**95.1493**
No. of unclassified instances	0	0	0	0	0	0	0
Model building time in seconds	0.03	4.55	0.67	0.06	0.41	0.11	0.82
Model testing time in seconds	0.01	0.01	0.01	0.00	0.00	0.01	0.01

and 95.1493 %, respectably as shown in Table 1. We also investigated the rest of the algorithms as a base algorithm of AdaBoost but the accuracies have not shown any improvement compared to J48.

From the experimental performance we observed that not a single instance was unclassified. It added the classifiers strength on our problem. Basically in the classifier selection phase user always provide emphasis on two ingredients: accuracy and computational complexity. We have observed the computational complexity in both ways including model building and performance evaluation of the model. In the model building phase, we observed that Naive Bayes was the winner classifier when we did not consider the AdaBoost method. However, after adopting the AdaBoost we found J48 is the faster base classifier than PART. In the model evalution phase we observed that J48 took almost 0 seconds compared with PART. Therefore, we suggest J48 could be a better choice for combating cyber attacks in the cloud domain.

6 Limitations and Scope for Future Work

Notwithstanding the fact that cloud computing is still emerging in the information technology landscape, and so many of the future threats may still be unknown, we can always train a machine with the range of attack patterns learned in the past and, slot that in the new architecture. Further investigations and growth into the following areas and issues would be useful:

- Investigate any outage that happened in Cloud computing and find out the actual cause and, if that has happen due to an attack, prepare full documentation of that attack in order to combat.
- Develop a real-time monitoring system with a large volume of data that will help in proactive attack detection model using modern machine learning techniques and learning tools such as WEKA.
- Conduct experimental analysis to investigate the real world documented attack scenarios, which normally occur in a combination of attack types.
- Investigate and analyse the potential challenges of real life implementation of proactive attack detection model: how Cloud providers can co-operate with their customers in this regard.
- Propose suitable solutions to mitigate these problems with the help of a series of experiments. For instance, identifying combination of attacks in Cloud Computing using supervised learning and new novelty detection model for cyber attacks in Cloud Computing using unsupervised learning, thereby identifying the attacks in an intelligent way.
- Develop pattern analysis graph using database where machine learning will be able to detect any novel attacks.

7 Conclusions

In this chapter, we attempted a resolution of one of the major trust issues between Cloud providers and their customers: "cloud providers are not transparent and try to hide security related data from their customers". We endeavoured to empower the customers with a novel approach on how cloud customers can detect cyber attacks happening in their VM by collecting performance data and using machine learning techniques. The reason for using performance data here instead of traditional security logs and data is that, cloud customers can easily generate performance data of their VM using built-in or third party software without assistance from the cloud provider. We demonstrated through performance charts from hypervisor and performance plots from our data collection to show how accurately these kinds of data can be collected throughout an attack. We also presented the performances of several of the most popular machine learning techniques on attack identification in a cloud environment and, a comparison on performances has been made. We used accuracy measure for the final selection of a learning technique for the task. In this chapter we suggested the use of AdaBoost method with base classifier J48 to protect the Cloud Computing environment from cyber attacks. The major advantage of AdaBoost is that it considers a weighted sample to focus learning on the most difficult attack rather than a random sample of the training data. Moreover, this method incorporates weighted vote instead of combining classifiers with equal vote, the latter is very common in the traditional learning theory.

We admit this study is still in a preliminary stage that needs substantial verification in diverse areas.

References

1. Amazon (2011) Amazon Elastic Compute Cloud (Amazon EC2). Retrieved 27 June 2011, from http://aws.amazon.com/ec2/
2. Archer J, Boehme A, Cullinane D, Kurtz P, Puhlmann N, Reavis J (2010) Top threats to cloud computing, version 1.0. Cloud security alliance. Retrieved 7 May 2011, from http://www.cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf
3. Batishchev AM (2012). LOIC. Retrieved 22 Aug 2012, from http://sourceforge.net/projects/loic/
4. BBC (2010) Anonymous Wikileaks supporters explain web attacks. Retrieved 23 Aug 2012, from http://www.bbc.co.uk/news/technology-11971259
5. Chonka A, Xiang Y, Zhou W, Bonti A (2010) Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks. J Netw Comput Appl.
6. Company H-PD (2012) HP proLiant DL380 G4 server–specifications retrieved 6th Aug 2012, from http://h18000.www1.hp.com/products/servers/proliantdl380/specifications-g4.html
7. Corporation M (2012) Windows 7 Retrieved 6th Aug 2012 from http://windows.microsoft.com/en-au/windows7/products/home
8. Dahbur K, Mohammad B, Tarakji AB (2011) A survey of risks, threats and vulnerabilities in cloud computing. Paper presented at the Proceedings of the international conference on intelligent semantic web-services and applications, ISWSA '11. ACM, New York, USA.

9. Danchev D (2008a) Coordinated Russia vs Georgia cyber attack in progress. Retrieved Oct 25 2008.
10. Danchev D (2008b) The DDoS attack against CNN. com.
11. Danchev D (2011) Dancho Danchev's blog–mind streams of information security knowledge retrieved 31 May 2011, from http://ddanchev.blogspot.com/
12. Frank E, Witten IH (1998) Generating accurate rule sets without global optimization. Paper presented at the fifteenth international conference on machine learning.
13. Freund Y, Schapire R, Abe N (1999) A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence 14(771–780):1612
14. Friedman J, Hastie T, Tibshirani R (2001) The elements of statistical learning, vol 1. Springer Series in Statistics, New York
15. Grid G (2010) Tutorial: how to DoS attack (Ping flooding). Retrieved 23 Aug 2012, from http://ghostgrid.blog.com/2010/12/16/ping-flooding/
16. Grossman J (2011) Jeremiah Grossman. Retrieved 19 June 2011, from http://jeremiahgrossman.blogspot.com/
17. Inc G (2012) NewEraCracker / LOIC. Retrieved 22 Aug 2012, from https://github.com/NewEraCracker/LOIC/
18. John GH, Langley P (1995) Estimating continuous distributions in Bayesian classifiers. Paper presented at the Eleventh Conference on Uncertainty in Artificial Intelligence, San Mateo
19. Khorshed MT, Ali ABMS, Wasimi SA (2011a) Monitoring insiders activities in cloud computing using rule based learning. Paper presented at the Proceedings of IEEE TrustCom-11, Nov 16–18, Changsha, China.
20. Khorshed MT, Ali ABMS, Wasimi SA (2011b) Trust issues that create threats for cyber attacks in cloud computing. Paper presented at the Proceedings of IEEE ICPADS, December 7–9, 2011, Tainan, Taiwan.
21. Khorshed MT, Ali A, Wasimi SA (2012a) Classifying different DoS attacks in cloud computing using rule based learning. Security and Communication Networks, Wiley, New York. doi:10.1002/sec.621
22. Khorshed MT, Ali A, Wasimi SA (2012b) A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing. Future generation computer systems. Elsevier, New York. doi:10.1016/j.future.2012.01.006
23. Kumar A, Sharma AK, Singh A (2012) Performance evaluation of centralized multicasting network over ICMP ping flood for DDoS. Perform Eval 37(10)
24. Lopez R, Onate E (2006) A variational formulation for the multilayer perceptron. Artificial Neural Networks-ICANN 2006:159–168
25. McDowell M (2009) Understanding denial-of-service attacks. Retrieved 10 Jan 2013, from http://www.us-cert.gov/cas/tips/ST04-015.html
26. Nanda R (2008) DDoS Attack/PING flooding: explanation and solution. Retrieved 23 Aug 2012, from http://ramannanda.blogspot.com.au/2009/05/ddos-attackping-flooding-explanation.html
27. Platt JC (1999) Fast training of support vector machines using sequential minimal optimization. Paper presented at the Advances in Kernel Methods–Support Vector Learning.
28. Quinlan JR (1993) C4. 5: programs for machine learning. San Mateo, Morgan Kaufmann, CA.
29. Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106
30. Quinlan JR (1987) Simplifying decision trees. Int J Man-Mach Stud 27(3):221–234
31. Rimal BP, Choi E, Lumb I (2009) A taxonomy and survey of cloud computing systems. Paper presented at the NCM '09. In: Proceedings of the (2009) fifth international joint conference on INC, IMS and IDC (IEEE Computer Society). Washington, DC, USA
32. Ristenpart T, Tromer E, Shacham H, Savage S (2009) Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. Paper presented at the Proceedings of the 16th ACM conference on Computer and communications security Chicago, Illinois, USA.
33. Rocha F, Correia M (2011) Lucy in the Sky without Diamonds: stealing confidential data in the cloud.

34. Rouse M (2006) Ping of death. Retrieved 23 Aug 2012, from http://searchsecurity.techtarget.com/definition/ping-of-death
35. securitytube.net (2012) Ddos attack with Rdos and T3c3i3. Retrieved 22 Aug 2012, from http://www.securitytube.net/video/4719
36. Tan WPN, Steinbach M, Kumar V (2005) General approach to solving a classification problem. Introduction to Data Mining, Pearson Addison-Wesley, Boston
37. VMware (2011) VMware vSphere Hypervisor. Retrieved 16 July 2011, from https://www.vmware.com/tryvmware/?p=esxi&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:en-US:official&client=firefox-a
38. Williams G (2008) DATA MINING Desktop survival guide. dim (survey) 1(32561):15
39. Wilmes G, Kistler U (2007) Engage packet builder–scriptable libnet-based packet builder. Retrieved 24 Aug 2012, from http://www.engagesecurity.com/products/engagepacketbuilder/
40. Witten IH, Frank E, Hall MA (2011) Data mining: practical machine learning tools and techniques, 3rd edn. Morgan Kaufmann, San Francisco

Legal Aspects of Data Protection in Cloud Federations

Attila Kertesz and Szilvia Varadi

1 Introduction

Cloud Computing offers on-demand access to computational, infrastructure and data resources operated from a remote source. Taking advantage of flexible resource provisions enabled by the Cloud technology, many businesses have recently migrated their IT applications and data to the Cloud, allowing them to respond to new demands and requests from customers. The technical motivation for Cloud Computing has been introduced in [4, 48]. Cloud solutions enable businesses with the option to outsource the operation and management of IT infrastructure and services, allowing the business and its employees to concentrate on their core competencies. This new technology enables services to be designed and tailored to the individual requirements of a business, and it also moves functions and responsibilities away from local ownership and management to a service provided by a third-party, and raises several legal issues, such as data protection, which require this service to comply with necessary regulation. As more and more businesses become global, concerns also remain over privacy of widely-distributed data and its processing. Regulations focusing on geographical locations may be a large obstacle to a widespread adoption of Cloud Computing solutions by companies [47].

As a result of the pace of technical and economic progress in this field, it is important to determine the compliance of common Cloud Computing usage patterns with legal constraints and requirements. In this chapter we provide a method for and the results of an evaluation of commonly-observed Cloud federation use cases against the law applying to Cloud Computing. First we derive a general architecture for

A. Kertesz (✉)
MTA SZTAKI Computer and Automation Research Institute, 63, Budapest 1518, Hungary
e-mail: kertesz.attila@sztaki.mta.hu

S. Varadi
Department of International and European Law, University of Szeged,
Rakoczi ter 1, Szeged 6722, Hungary
e-mail: varadiszilvia@juris.u-szeged.hu

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5_15,

Clouds from definitions of international standardization bodies, and use it to identify common Cloud Computing usage patterns. To point out where legal problems may arise, we summarize the national laws related to data protection of major countries, then we assess the revealed use cases against evaluation criteria derived from legislation for the data processing of end-user details and materials, including the roles and responsibilities necessary for legal compliance.

To clarify and exemplify legal compliance in the identified usage patterns, we consider the Data Protection Directive [20] of the European Union, which is a commonly accepted and influential directive in the field of data processing legislation. A paper by Bygrave [5] investigated the possible impact of this directive on the activities of E-commerce operators, and a deliverable of the OPTIMIS European project [40] studied in detail the applicability of this directive for their own Cloud deployment models. In this chapter we take a step forward and examine use cases identified in a generalized architecture compiled from reports of international expert groups, bodies and research projects.

The remainder of this chapter is as follows: Sect. 2 introduces and analyzes several Cloud architectures, and derives a general Cloud federation architecture that encompasses their features. Section 3 reveals common use cases of Cloud federations to show where legal aspects may arise concerning data management. In Sect. 4 we summarize the relevant legislation regarding data protection in Clouds, and Sect. 5 introduces in detail the European law applying to Cloud Computing. Section 6 presents a case study to exemplify how the European law can be applied to the revealed Cloud Computing use cases. Section 7 discusses the recent European reform and future developments in this area. Finally, Sect. 8 summarizes the lessons learned in this chapter, and Sect. 9 provides hints for future research directions.

2 Cloud Federations and the Applied Architectural Models of Clouds

2.1 View of the European Commission

An expert group associated with the European Commission published their view on Cloud Computing in [46] and [45]. These reports categorize Cloud architectures into five groups, as shown in Fig. 1. *Private Clouds* (1) consist of resources managed by an infrastructure provider (IP) that are typically owned or leased by an enterprise from a service provider (SP). Usually, services with "Cloud-enhanced" features are offered in this category, therefore this group includes SaaS (Software as a Service) solutions like eBay [24]. *Public Clouds* (2) offer their services to users outside of the company and may use Cloud functionality from other providers. In this solution enterprises can outsource their services to such Cloud providers mainly for cost reduction. Examples of these providers are Amazon [2] or Google Apps [31]. *Hybrid Clouds* (3) consist of both private and public Cloud infrastructures to achieve a higher level of cost

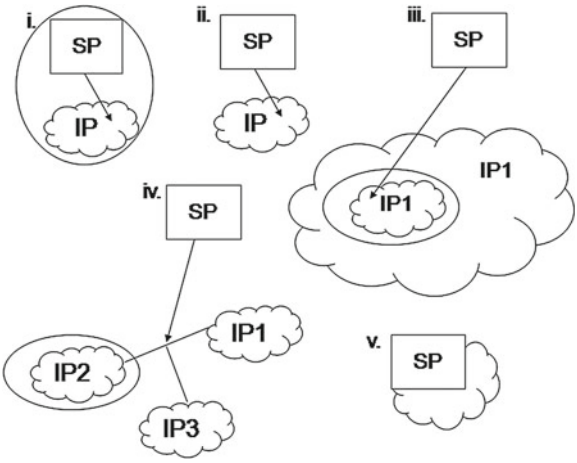


Fig. 1 Cloud architectures derived from the cloud computing expert working group report

reduction through outsourcing by maintaining the desired degree of control (e.g., sensitive data may be handled in private Clouds). The report states that hybrid Clouds are rarely used at the moment. In *Community Clouds* (4) different entities contribute with their (usually small) infrastructure to build up an aggregated private or public Cloud. Smaller enterprises may benefit from such infrastructures, and a solution in this category is provided by Zimory [53]. Finally, *Special Purpose Clouds* (5) provide more specialized functionalities with additional, domain specific methods, such as the distributed document management by Google's App Engine [31]. This group is an extension or a specialization of the previous Cloud categories.

2.2 ENISA Architectures

The European Network and Information Security Agency (ENISA) differentiates between four architectures [6, 7], which are shown in Fig. 2. A *Public Cloud* (1) is a publicly-available infrastructure to which any organization may subscribe and use [also called as service consumers (SC)]. *Private Clouds* (2) offer services built on Cloud Computing principles, but accessible only within a private network. *Partner Clouds* (3) are operated by a provider to a limited and well-defined number of parties. Finally, a *Cloud Federation* (4) may be built up by aggregating two or more Clouds.

2.3 NIST Cloud Architectures

The National Institute of Standards and Technology (NIST) defines four deployment models [34, 36] depicted in Fig. 3. According to their definitions, a *Private Cloud* (1) is an infrastructure operated solely for an organization that may be managed by

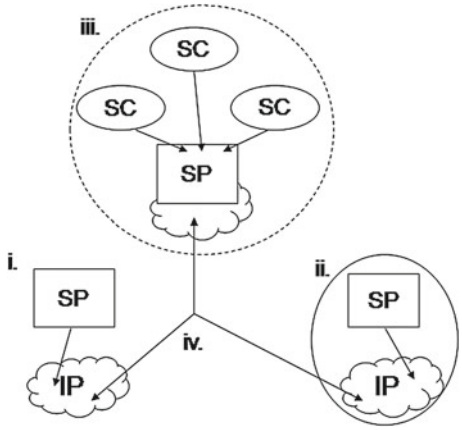


Fig. 2 ENISA cloud architectures

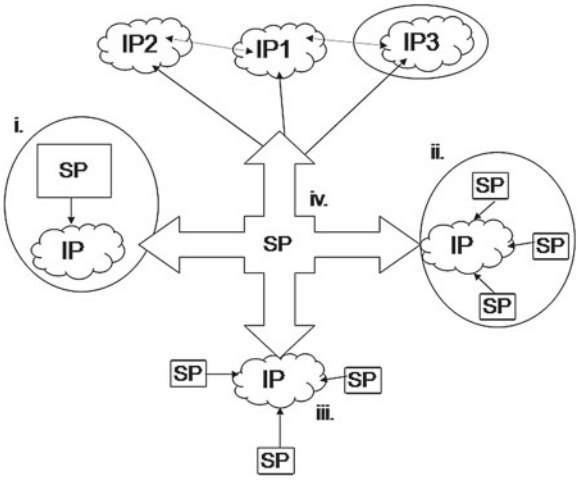


Fig. 3 NIST deployment models

either the organization or a third-party and located locally or remotely. A *Community Cloud* (2) is shared by several organizations, and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by organizations or third parties, and may exist on premises or off premises. A *Public Cloud* infrastructure (3) is made available to the general public or a large industry group, and is owned by an organization selling Cloud services. Finally, a *Hybrid Cloud* (4) is a composition of two or more Clouds (private, community or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., Cloud bursting for load balancing between Clouds).

The Cloud Computing Use Case Discussion Group [8] adopts the NIST models. They extend the view on *Hybrid Clouds* by stating that "multiple Clouds work together, coordinated by a Cloud broker that federates data, applications, user identity,

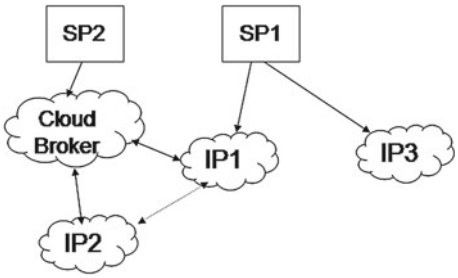


Fig. 4 DMTF deployment models

security and other details". Though a brokering mechanism is needed for federating Clouds, no specific guidelines are given how to achieve this.

The Distributed Management Task Force (DMTF) Open Cloud Standards Incubator view [21] has also adopted the NIST models and defined different scenarios showing how Clouds may interoperate (depicted in Fig. 4). These scenarios explain how datacenters interact with Cloud providers and differentiate three cases:

- If a datacenter, run by Service Provider 1 (SP1) and hosted by Infrastructure Provider 1 (IP1), exceeds the available capacity limits then IP2 provides extra computing capacity for IP1, while SP1 is unaware of this provisioning.
- In a multiple Cloud scenario, SP1 may operate services in both IP1 and IP3 Clouds, therefore a datacenter may request services from both providers since they may support different services or Service-level Agreement (SLA) parameters.
- A provider may act as a Cloud broker to federate resources from other providers (e.g., IP1 and IP2) to make them available to its consumers transparently without using any of its own resources.

2.4 OPTIMIS Project

The architectural views of the OPTIMIS project [28] are shown in Fig. 5. The project has three basic architectural scenarios. In a *Federated Cloud Architecture* (1) a Service Provider (SP) assesses an Infrastructure Provider (IP). IPs can share resources among each other. In a *Multi-Cloud Architecture* (2) different infrastructure providers are used separately by a service provider. Finally in a *Hybrid Cloud Architecture* (3) a Private Cloud (PC) is used by the SP, which can utilize resources of different IPs.

2.5 Reservoir Project

The Reservoir project [43] claims that small and medium Cloud providers cannot enter the Cloud-provisioning market, due to the lack of interoperability between their Clouds. Their approach is exemplified by the electric grid approach: "for one facility to dynamically acquire electricity from a neighboring facility to meet a spike in

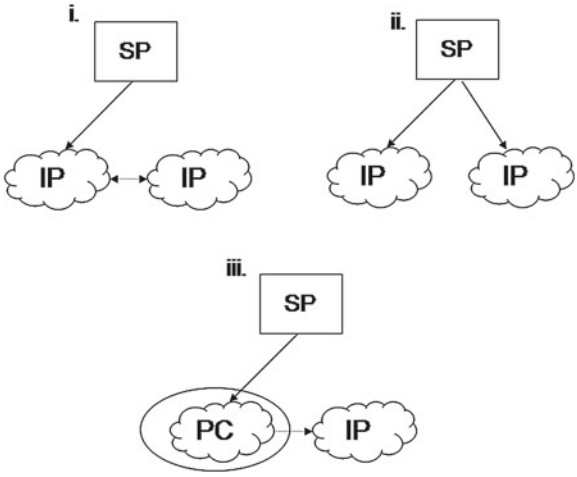


Fig. 5 OPTIMIS Cloud architectures

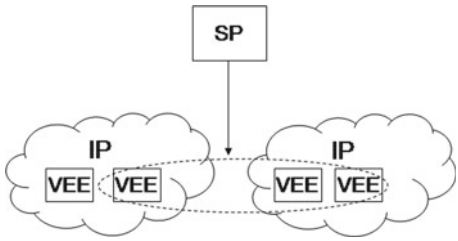


Fig. 6 Reservoir cloud architecture

demand". Disparate datacenters should be federated in order to provide a "seemingly infinite service computing utility". Regarding their architectural view, a Reservoir Cloud consists of different Reservoir Sites (RS) operated by different IPs. Each RS has resources that are partitioned into isolated Virtual Execution Environments (VEE). Service applications may use VEE hosts from different RSs simultaneously. Each application is deployed with a service manifest that formally defines its SLA contract. Virtual Execution Environment Managers (VEEM) interact with VEEs, Service Managers and other VEEMs to enable federations to be formed. A VEEM gathers interacting VEEs into a VEE group that serves a service application. This implies that a Reservoir service stack has to be present on the resources/sites of IPs. Their specialized Cloud architecture is depicted in Fig. 6.

2.6 A Unified View of a Cloud Federation

Figure 7 shows an extended view of *Federated Clouds* incorporating private, public, multi- and hybrid Cloud architectures. Interoperability is achieved by high-level brokering instead of bilateral resource renting. Nevertheless different IPs may share or rent resources, but it is transparent to their management. Such a federation can

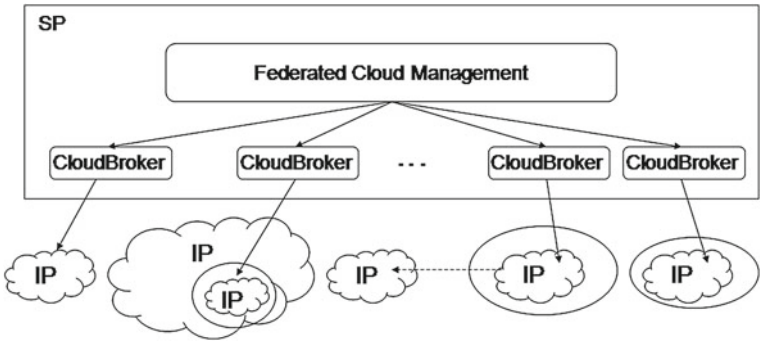


Fig. 7 Federated cloud management architecture

be enabled without applying additional software stack for providing low-level management interfaces [35]. The logic of federated management is moved to higher levels, and there is no need for adapting interoperability standards by IPs (and some industrial providers are reluctant to do so).

In this case, Cloud providers offering PaaS solutions may form "sub-federations" simultaneously to this approach. Specific service applications may be more suitable for such a solution, and projects like Reservoir [43] and 4CaaSt [1] are working towards such a solution. This approach targets IaaS-type providers, e.g., RackSpace, the infrastructure services of Amazon EC2, and providers using Cloud middleware such as OpenNebula or Eucaliptus.

3 Use Cases for Data Management in Cloud Federations

3.1 General Usage Scenarios

The federated Cloud architecture described in Sect. 2.6 is further investigated through a series of use cases to demonstrate where legal issues can arise by using this general organizational structure. Later on these usage scenarios are used to exemplify how service and infrastructure providers should apply the legislation for data protection, and what necessary actions should be taken in order to prevent violations of the applicable regulations. As we will show, the most complicated situations arise when personal data is transferred to multiple jurisdictions.

The most common use case (**C0**), as depicted in Fig. 8, is when a user asks a service provider (SP) to store (and possibly process) their personal data. The SP leases Cloud resources from an infrastructure provider (IP) and the private data of the user is stored in a database (DB) and managed by a virtual machine (VM) located in an establishment (E) of the IP.

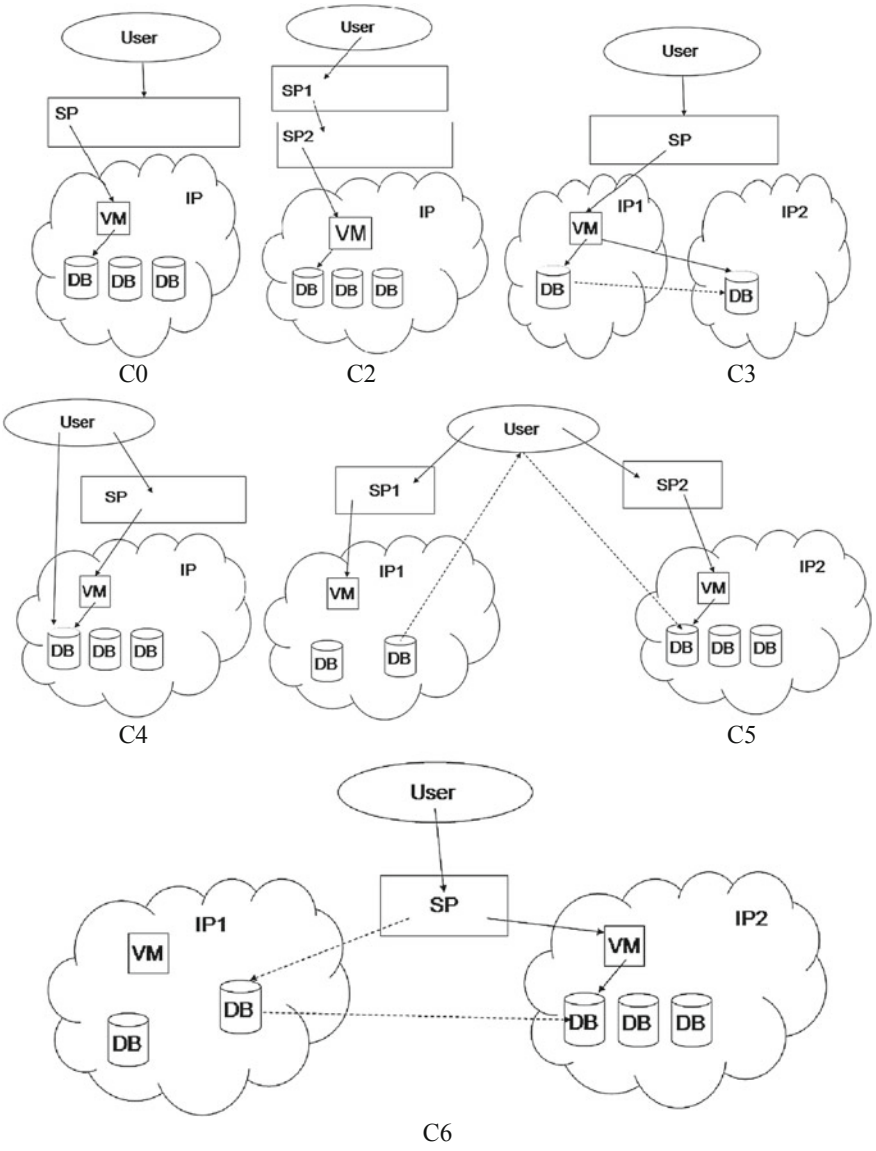


Fig. 8 Common use cases

C1: As the first case, let us consider a situation, when a SP offers its infrastructure to the user through a service front-end directly (so the SP and the IP are the same entities).

C2: The previous case first becomes complicated, when a Cloud service offered by SP1 is actually re-sold to the end-user by another service provider (SP2), who is in contact with IP: in this way a chain of service providers can be utilized.
C3: In this case let us consider a situation, when an SP uses an infrastructure provider (IP1), and IP1 uses resources from a different provider (IP2) to be able to serve higher resource demands of SP. For commercial reasons, this relationship is not necessarily made public, therefore the SP using resources of IP1 may not be aware of that it is also using IP2—as shown in Fig. 8.
C4: In this situation, a user asks an SP to manage his/her private data, and the SP uses an IP for this task which has a public interface, therefore the user may modify its data directly without involving the SP.
C5: In this case a user asks two different service providers (SP1 and SP2) to manage two databases, and the user decides to migrate a part of the data from SP1 to SP2 by themselves.
C6: If an SP utilizes two different IPs (IP1 and IP2) for storing user data, the user may ask the SP to migrate data from IP1 to IP2.

3.2 Location-Related Cases

In the following we discuss use cases where legal issues may arise due to private data processing at different geographical locations, which represent multiple jurisdictions. In these cases let us consider the use case where a service managing private user data is deployed and executed in a datacenter of a Cloud Infrastructure Provider (IP) located in a jurisdiction (J). The service provider utilizes several IPs by aggregating them into a Cloud federation.

The simplest use case (**L0**) is where the SP has one datacenter or establishment (E) located in a single jurisdiction. Here, the law applicable to data processing will generally be the national law applied in the appropriate country (jurisdiction) regardless of the geographical location of the IP. We now introduce more complex cases (also considering the discussions of the Data Protection Working Party [16]) to reveal location-related issues.
L1: In this case an IP has facilities located in different jurisdictions (e.g., J1 and J2). It distributes user data handled by a service on a virtual machine (VM1) using a database (DB1) in an establishment in J1, and another service on a virtual machine (VM2) using a database (DB2) in an establishment in J2.
L2: Similar to the previous case, when an IP (established in J1) has different establishments located in different jurisdictions (e.g., J1 and J2), and it distributes user data between these establishments. In the specific case depicted in Fig. 9a, the processing of the data by VM located in an establishment in J1 involves data located in a different establishment in J2.
L3: Similarly to use cases L1 and L2, when the IP has several establishments (possibly in a third country), and at least one of these is located in the jurisdiction J1.

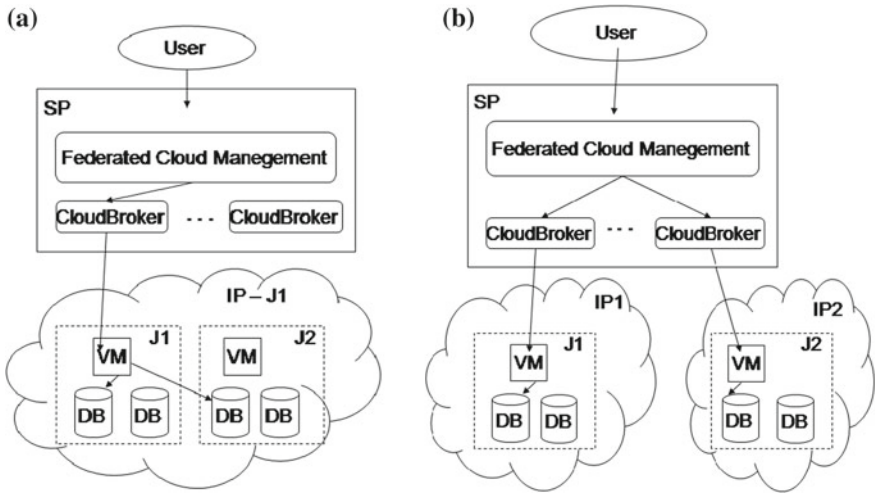


Fig. 9 Data distribution regulations in multiple jurisdictions: **a** with one infrastructure provider (depicting case L2); **b** with different infrastructure providers (concerning case L5)

L4: Similar to case C3, if IP1 in J uses resources from a different provider in J2 (e.g., IP2), the SP using IP1 may be unaware that it is using IP2.
L5: In this situation, an SP provides a federated Cloud management in J1, it utilizes different IPs, and one of these providers (IP2) is located in J2—depicted in Fig. 9b.
L6: Finally, in this case the SP providing a federated Cloud management utilizes different IPs, one of which (IP2) is located at J1.

4 Legislation for Data Protection Applying to the Cloud

According to a recent report [32], the first influential national legislation for data protection was the Swedish law in 1973. Since then, there has been a major technological development in Computer Science, and the emergence of Cloud Computing has contributed to the escalation of national legislations; as a result 89 countries have data protection laws nowadays. These national laws determine separate legal jurisdictions, which are generally the countries themselves. But there are some exceptions: there could be special administrative regions within a country, which belong to different jurisdictions (such as Hong Kong and Macao in China). On the other hand, there are some international organizations which defined guidelines for harmonizing national laws of data protection. For example the OECD's (Organisation for Economic Cooperation and Development) privacy Guidelines of 1981; the APEC's (Asia-Pacific Economic Cooperation) Privacy Framework of 2005; the Data Protection Convention of the Council of Europe of 1981; and finally the European Union's data protection Directive of 1995, which has been the most influential international

instrument. We introduce and analyze in detail the data protection directive of the European Union (EU) in Sect. 5.

Taking a closer look at the national legislation of the most influential countries for Cloud Computing worldwide, in the United States of America (US) there is no comprehensive data protection law, since there are no authorities having such power, and its states can regulate data protection separately. Regarding federated laws, the Privacy Act of 1974 regulates the collection, maintenance, use and dissemination of personal information by federal agencies, but it cannot be applied to non-US citizens, and it contains outdated guidelines that cannot cope with novel technological solutions. Such controversial and uncovered issues could be solved by litigations and case law development [19]. In order to handle cross-border situations, the US uses international agreements. For example the Safe Harbor Framework has been created to enhance US-EU cooperation by providing streamlined means for US organizations to fulfill the adequate level criteria for data protection of the EU [44].

In Canada, there is a unified data protection law called the Personal Information Protection and Electronic Documents Act of 2000 (PIPEDA), which has also been approved by the EU to fulfill its adequacy of data protection. Besides this legal document, there is a privacy commissioner dedicated to data protection issues [39]. This commissioner is an advocate for the privacy rights of Canadians and reports directly to the House of Commons and the Senate. Her powers include: investigating complaints, conducting audits and pursuing court action, reporting on the personal information-handling practices of public and private sector organizations, undertaking research into privacy issues, and promoting public awareness and understanding of privacy issues.

Australia also has a data protection law called Privacy Act of 1988, which was amended in 2012 by Privacy Amendment (Enhancing Privacy Protection) Bill resulted from a privacy law reform process started in 2006. It has introduced new privacy principles for both private and public sectors. It enhances the powers of the Information Commissioner, who is responsible for the freedom of information, privacy and government information policy functions [38].

The Act for Protection of Personal Data (PDPA) was enforced in Japan in 2005. The PDPA outlines basic data protection policies, directs the bureaucracies that protect privacy, regulates businesses processing personal data and imposes sanctions for violations. This act has helped many people in Japan to understand the value of personal data protection, since the Japanese society has not been very sensitive to the protection of privacy, which is probably due to the Japanese cultural and social environment [41]. The Japan Personal Information Protection Act (PIPA) and Law (PIPL) provide further regulations for companies, which require enterprises to manage and protect the rights of Japanese citizens with regard to their personal information, while preserving the usefulness of information technology and personal information for legitimate purposes [37]. They require businesses to communicate why they are collecting and using personal information. They must take reasonable precautions to protect personal information from improper disclosure, unauthorized use or destruction.

China does not have a comprehensive legal framework to regulate the use of personal data [51]. There is no national law for companies defining how to legally collect or process data together with legal remedies for specific violations. The relevant rules are stated in diverse laws, regulations and local ordinances, for example in the Open Government Information (OGI), which requires government organs to proactively disseminate government information and gives natural persons the right to request information from the government [29].

As stated before, Cloud Computing allows the outsourcing of computational power, data storage and other capabilities to a remote third-party. In the supply of any goods and services, the law gives certain rights that protect the consumer and provider, which also applies for Cloud Computing: it is subject to legal requirements and constraints to ensure Cloud services are accurately described and provided to customers with guarantees on quality and fitness-for-purpose. As Sect. 2 of [40] describes, the characteristics of Cloud Computing make it of interest to three main fields of law:

- Intellectual property law, as data and applications (i.e., code) hosted in the Cloud may contain trade secrets or be subject to copyright and/or patent protection;
- Green (i.e., ecological) legislation, since the datacenters hosting the basic Cloud infrastructure (e.g., servers, switches, routers, etc.) require a large amount of energy to operate and indirectly produce carbon dioxide;
- Data protection law.

In this evaluation of data management in Cloud federations against legal requirements we have chosen to perform the investigation exclusively using requirements from data protection law. We do not consider intellectual property law because, as [40] describes, it is often considered on a case-by-case basis making it difficult to derive common requirements to fulfill these obligations for a Cloud architectures. Secondly, green legislation is also not considered here as compliance, because it is an orthogonal concern to the Cloud architecture used; different providers may implement the same architecture with different levels of eco-friendliness.

However, using only one field of law does not restrict the evaluation; data protection covers the dynamic provisioning and processing of data in Cloud environments—intrinsic to the operation of all Clouds—and the field covers the majority of currently available Cloud Computing characteristics and functions, including cases where (Sect. 4 of [40]):

- The infrastructure used to store and process a costumer's data is shared with other customers (i.e., multi-tenancy);
- The Cloud provider's servers are located in several jurisdictions;
- Data is transferred from one location (also called as establishment) to another depending on where resources are available;
- The Cloud service provider decides the location of the data, service and security standards instead of the customer;
- IT resources are not dedicated to a customer but instead are dynamically provisioned.

Thus, data protection legislation is fundamental to Cloud Computing as the consumer looses a degree of control over personal artifacts, when they are submitted to the provider for storage and possible processing. To protect the consumer against the provider misusing their data, data processing legislation has been developed to ensure that the fundamental right to privacy is maintained. However, the distributed nature of Cloud Computing (in that Cloud services are available from anywhere in the world) makes is difficult to analyze every country's data protection laws for common Cloud usage evaluation criteria. We have therefore chosen a common directive that applies as widely as possible and used the European Data Protection Directive (DPD) as a basis for our evaluation. Although it is an EU directive, "countries that wish to engage in data transactions with EU Member States are indirectly required to provide an adequate level of protection" and "the Directive has had a far greater global impact than thus far acknowledged", making it an "effective mechanism to raise the level of data protection worldwide" [3]. In the following section we discuss the EU DPD in detail.

5 Regulation for Data Protection in the European Union

5.1 European Data Protection Directive

EU Data Protection Directive [20] is a directive adopted by the European Union designed to protect the privacy and protection of all personal data collected for or about citizens of the EU, especially as it relates to processing, using, or exchanging such data [49]. All 27 EU Member States are reported to have enacted their own data protection legislation that transposes the directive into internal law. Canada, Australia and Argentina have implemented legislation that complies with the DPD, Switzerland has partially implemented legislation and the USA has voluntary registration to the "Safe Harbor" program to ensure private companies who sign up adhere to the rules set out in the DPD [50].

The DPD has therefore been used in this evaluation as it is a commonly accepted and influential directive in the field of data processing legislation. It was produced in 1995, before Cloud Computing was developed, but can be applied to Cloud Computing as it describes how the protection of the processing of personal data and the free movement of such data should be achieved in a technology-neutral way. The DPD can be summarized as having elements concerned with the responsibilities of two actors involved in data exchanges and restrictions on the free movement of data between them based on their location.

The requirements of the DPD are expressed as two technology-neutral actors or roles that have certain responsibilities that must be carried out in order to fulfill the directive. These roles, the *data controller* and *data processor*, are naturally equivalent to service consumer and service provider roles found distributed computing. According to the Article 2 of DPD [20] a **data controller** is the natural or legal

person which determines the means of the processing of personal data, whilst a **data processor** is a natural or legal person which processes data on behalf of the controller. However, following these definitions, a special case arises: if the processing entity plays a role in determining the purposes or the means of processing, it is a controller rather than a processor. Finally, although not a specific role, third parties are defined in Article 2 of DPD [20] as "any natural or legal person, public authority, agency or any other body other than the data subject, controller, processor and the persons who are authorized to process the data". This definition is further clarified in [15] by stating that such a third party has no specific legitimacy or authorization to process the personal data, therefore it is not involved in the controller-to-subject relationship.

The DPD was designed to allow the free-flow of data between EU Member States. However, this directive also gives the opportunity to third countries to participate in free-flow activities, if deemed to implement "adequate level" of data protection (Article 25 of DPD [20]). This condition means that a third country has to provide at least the same level of protection as the national provisions of the Member States. Once this condition is fulfilled, they can interoperate with other providers within the EU with no barriers.

5.2 Responsibilities Associated to the Roles of the DPD

We have chosen the EU Data Protection Directive as legislation to evaluate current Cloud Computing use cases, since this directive is a widely-used and adopted set of rules governing Cloud Computing fundamentals. The DPD also introduces a set of responsibilities for the roles of data controller and processor. We can use these duties to form evaluation criteria to assess Cloud Computing use cases. The directive is also discussed in much detail with respect to Cloud Computing in [40], and provides a set of criteria that the roles must meet. According to these sources, the *data controller* must:

- Be responsible for compliance with data protection law.
- Comply with the general principles (e.g., legitimate processing) laid down in Article 6 of DPD.
- Be responsible for the choices governing the design and operation of the processing carried out.
- Give consent for processing to be carried out (explicit or implied, orally or in writing).
- Be liable for data protection violations.

The *data processor*, meanwhile, must:

- Process data according to the mandate and the instructions given by the controller.
- Be an agent of the controller.
- Be a separate legal entity to the controller.

These roles are strengthened, if:

- The controller gives detailed instructions to the processor.
- The controller monitors the processor for the status of the processing.
- Relevant expertise can be shown to be present in either party (e.g., the processor is a specialist in it).
- A written contract exists between the controller and processor.
- The controller is able to exercise full and sole control at any time while the data processing takes place.
- The controller is informed of the main elements of the processing structure.

Finally, in the evaluation of specific Cloud-usage scenarios we will assess location-related issues that may arise due to one of the establishments being outside a jurisdiction. In such cases, in general, an adequate level of data protection should be provided according to the EU DPD.

6 A Case Study for Applying the EU DPD to the Identified Use Cases

In this section we discuss how to apply the legislation defined in the DPD for the federated Cloud use cases described in Sect. 3. In these use cases the relevant actors and their roles are identified, from which the necessary actions can be derived in order to prevent violations of the directive. Table 1 summarizes these general cases and depicts the relevant roles (*data controller* (DC) or *data processor* (DP)) the participating actors may play regarding data protection compliance.

The most general use case, **C0**, was when a user asks a service provider to store and process his/her personal data. The SP leases Cloud resources from an infrastructure provider and the private data of the user is stored in a database and managed by a virtual machine located in an establishment of the IP. In this simple case the SP is a *data controller* and IP is the *data processor* instructed by SP. In the case **C1,** when the SP is also an IP, the SP is the *data controller* and there is no *data processor*. If there is a chain of SPs, like in **C2**, the last entity contacting the IP will become the *data controller* and the instructed IP will be the *data processor*. In case **C3**, when

Table 1 Identified roles in common use cases

Use case	User	Service provider	Infrastructure provider
C0	–	DC	DP
C1	–	DC	–
C2	–	DC	DP
C3	–	DC	DC and DP
C4	DC	DC	DP
C5	DC	DC	DP
C6	–	DC	DP

an infrastructure provider (IP1) uses resources from a different provider (IP2), when data moved from IP1 to IP2 for processing, IP1 will become the *controller* instead of SP, and IP2 will be the *processor* instead of IP1. In case **C4**, when the user may modify its data directly without involving the SP, the user becomes the *controller* when accessing the database directly. In **C5**, when the user decides to migrate a part of his/her data from SP1 to SP2 by themselves, the user becomes the *data controller* for the migration. When the user asks the SP to migrate data from IP1 to IP2 like in **C6**, the SP will stay in the role of *data controller* and IP1 will remain the *data processor* since the SP instructed IP1 to do the transfer. If IP1 is not interoperable with IP2, the SP will fetch the data from IP1 and move it to IP2, so the initial roles will not change.

In the following we discuss use cases where legal issues may arise due to private data processing at multiple jurisdictions resulting from utilizing data center establishments at different geographical locations. Considering European Cloud federations, the Article 4 of the DPD [20] states that the location of the data controller's establishment determines the national law applicable for data processing. (The DPD also applies to the EEA and EFTA countries according to [18]). In these cases let us consider the use case, where a service managing private user data is deployed and executed in a datacenter of a Cloud Infrastructure Provider located in an EU Member State (MS). Table 2 summarizes these location-related use cases, depicts the relevant roles (data controller (DC) or data processor (DP)) the participating actors (SP and IP) may play regarding data protection compliance, and specifies the entity, which location determines the national law applicable for data processing.

The service provider utilizes several IPs by aggregating them into a Cloud federation. The simplest use case, **L0**, is where the SP, who is the *data controller*, has one establishment located in an EU Member State. Here, the law applicable to data processing will be the national law of the MS regardless of the location of the IP. In case **L1**, when an IP has facilities located in different Member States (e.g., MS1 and MS2), the IP will become the *data controller*, and at each establishment the corresponding national law has to be applied (denoted by E_x in Table 2). In case **L2**, when an IP, originally established in a MS, has different establishments located in different Member States, and it distributes user data between these establishments, the IP is again the *data controller*. In the specific case depicted in Fig. 9a, the processing of the data by VM located in an establishment in MS1 involves data located in a different

Table 2 Location-related use cases

Use case	SP	IP	National law applicable
L0	DC	-	E of SP
L1	-	DC	E_x of IP
L2	-	DC	E_i of IP
L3	-	DC	E_{MS} of IP
L4	-	DC and DP	E_{MS} of IP
L5	DC	DP	E of SP
L6	DC	DP	E_{MS} of IP

establishment in MS2, the law of MS1 must be applied (denoted by E_i in Table 2). To further complicate this situation, in case **L3**, when the IP has several establishments (possibly in a third country), and at least one of them is located in a MS, the applicability of this MS's law (denoted by E_{MS} in Table 2) will be considered and has to be applied in all establishments. Regarding **L4**, if IP1 in MS uses resources from a different provider in a non-Member State (e.g., IP2), since the SP using IP1 may be unaware that it is using IP2, IP1 is the *data controller* and IP2 is the *processor* and the law of the appropriate MS is applicable. When IP1 is located in a non-MS, and IP2 in a MS, again the law of the appropriate MS is applicable, and IP1 has to provide the adequate level of data protection as defined in the DPD. In case **L5**, an SP utilizes different IPs and one of which (IP2) is located in a non-MS (as depicted in Fig. 9b), the SP is the *data controller* and IPs are *processors*, and the law of the SP's MS has to be applied (according to recital (18) in the preamble of DPD (Directive 95/46/EC, 1995)), and IP2 has to provide at least the same level of protection as the national law of MS. Otherwise, if IP2 cannot ensure an adequate level of protection, the decision making process should rule out IP2 from provider selection. In **L6**, the SP providing a federated Cloud management is not necessarily in a MS, and utilizes different IPs, one of which (IP2) is located at a MS. In this situation the SP is the *data controller* and the IPs are *processors*. Since the establishment (or an equipment) of IP2 is located in a MS, the law of this MS has to be applied, and the establishments located in non-MSs have to provide an adequate level of protection.

In order to conclude this case study, regarding the general use cases, Table 1 shows how the SP is mainly responsible for complying with the data protection regulation. When personal data is transferred to multiple jurisdictions, it is crucial to properly identify the controller since this role may change dynamically in specific actions. We have seen how information on the exact location of the processing establishments is also of great importance in the use cases. Table 2 highlights that even if one datacenter resides in the EU, the law of the appropriate Member State of this datacenter must be applied by the SP.

7 Recent Developments in European Legislation

As we have seen in the previous section, new developments in legislation regulation applying to Cloud Computing are still needed. In the considered use cases we have shown that multi-tenancy is inevitable in Cloud systems, and executing user applications may result in data processing at multiple jurisdictions, for example a user data can be stored in Budapest, processed in Amsterdam and accessed in New York. In the digital age, data is routinely transferred between countries both inside and outside the EU. But not all countries provide the same level of protection for personal data. Binding corporate rules represents one approach that can be used to adequately protect personal data, when it is transferred or processed outside the EU. Businesses can adopt these rules voluntarily and they can be used for transfers of data between companies that are part of the same corporate group. Currently, in order to

be approved, binding corporate rules must be verified by at least three data protection authorities. This situation is identified by Wong [52], who gathered related steps of the Article 29 Working Party to revise the EU directive. This Working Party was set up under the DPD, and it has advisory status and acts independently.

Besides, the European Commission is currently in the process of reforming the European data protection rules, where the main objectives are: to modernise the EU legal system for the protection of personal data, in particular to meet the challenges resulting from globalisation and the use of new technologies; to strengthen users' influence on their personal data, and at the same time to reduce administrative formalities to ensure a free flow of personal data within the EU and beyond; and to improve the clarity and coherence of the EU rules for personal data protection and achieve a consistent and effective implementation and application of the fundamental right to the protection of personal data in all areas of the Union's activities [9].

To achieve these above mentioned goals, the Commission has two legislative proposals according to a press release of the European Commission [25]: a Regulation [11] setting out a general EU framework for data protection and which will replace the currently effective DPD, and a Directive [10] on protecting personal data processed for the purposes of prevention, detection, investigation or prosecution of criminal offences and related judicial activities.

Personal data is increasingly being transferred across borders—both virtual and geographical—and stored on servers in multiple countries both within and outside the EU. The globalised nature of data-flows calls for strengthening the individuals' data-protection rights internationally. This requires strong principles for protecting individuals' data, aimed at easing the flow of personal data across borders while still ensuring a high and consistent level of protection without loopholes or unnecessary complexity. In these legal documents the Commission proposes the following key changes:

- A single set of rules on data protection across the EU to avoid unnecessary administrative requirements.
- It places increased responsibility and accountability for the companies processing personal data (e.g., they must notify the national supervisory authority of serious data breaches within 24 hours).
- It promotes a single national data protection authority in each EU country that people can refer to, even when their data are processed by a company based outside the EU. These authorities will be empowered to fine companies that violate EU data protection rules.
- It strengthens the right to data portability by enabling easier access to users' personal data, and easier data migration among service providers.
- It introduces the 'right to be forgotten' to enable the deletion of user data upon request, when there are no legitimate grounds for retaining it.
- It explicitly states that EU rules must be applied for data processing outside the EU by companies that are active in the EU market.

The Commission's proposals have been sent to the European Parliament and the Council of the European Union (sometimes just called the Council, and sometimes

still referred to as the Council of Ministers) for discussion. The Committee on the Internal Market and Consumer Protection [22] and the Committee on Legal Affairs of the European Parliament [23] proposed some amendments in the text of the regulation. The Council of the European Union has also stated that further examination of the text of the proposed regulation is still needed [14]. The new legislation will take effect two years after they have been adopted by Parliament and the Council.

The European Commission has also initiated a public consultation [26, 42], in the framework of their Digital Agenda for Europe (DAE) launched in May 2010, to find the requirements, barriers and opportunities for the provisioning and use of Cloud Computing which is highlighted in the Agenda [12]. As a result, the European Cloud Computing strategy was published by the Commission [27] in 2012. In this strategy the Commission aims at enabling and facilitating faster adoption of cloud computing throughout all sectors of the economy in order to boost the productivity, growth and jobs. On the basis of an analysis of the overall policy, regulatory and technology landscapes and a wide consultation of stakeholders, undertaken to identify what needs to be done to achieve that goal, the European Commission endorsed a communication on "Unleashing the Potential of Cloud Computing in Europe" [13]. This document sets out the most important and urgent additional actions. It represents a political commitment of the Commission and serves as a call on all stakeholders to participate in the implementation of these actions. The strategy includes three key actions: Standards and Certification, Contract terms and Conditions, and European Cloud Partnership. The necessary standards are planned to be identified by 2013. The development of model contract terms is planned to cover issues such as: data preservation, disclosure and integrity, data location and transfer, ownership of the data, direct and indirect liability change of service by cloud providers and subcontracting. Note, that these are similar issues that have been identified by our investigation of the Cloud federation usage patterns. The European Commission set up a European Cloud Partnership (ECP) program in 2012 to create a common framework for cloud computing across Europe [33], by bringing together industry expects and public sector users to work on common procurement requirements for cloud computing in an open and fully transparent way. The steering board of the new European Cloud Partnership has met on 20 November 2012 in Brussels, kicking off the process to build an EU Digital Single Market for cloud computing.

As we have seen in Sect. 5, the currently effective European DPD is basically appropriate for determining the law applicable for data management in Cloud services, when the data controller and processor roles are well identified. What is more problematic for companies is to apply the identified law at a European scale, because the Member States implemented the DPD rules in different ways. This fact has also been recognized by the European Commission as discussed above. In our opinion, instead of taking sanctions, it decided to perform a reform of the data protection rules using the principle of subsidiarity. Based on this principle the Union can introduce a unified legislation for data protection to be applied by all Member States. This reform will also give the opportunity for the Commission to replace the flexible directive with a strictly applicable regulation.

Concerning our investigation, this proposal for a new regulation mostly clarifies, restates and strengthens the referred rules of the DPD. Only the so-called "right to be forgotten" introduces a new responsibility for Cloud service providers. Some providers claim in the service usage terms and conditions to have the right to retain data, which may be affected by this new regulation. Even though it would definitely be a positive sign for the users and would encourage service utilization, but it would also place further development costs for providers, since the removal of all data replicas may also raise some technical problems.

8 Lessons Learned and Discussion

Cloud solutions enable businesses with the option to outsource the operation and management of IT infrastructure and services, allowing the business and its employees to concentrate on their core competencies. As many businesses are migrating their IT applications and data to Clouds to take advantage of the flexible resource provision, the compliance with the legal issues of data management in these systems become crucial. In this chapter we have derived a general federation architecture for Clouds from definitions of international organizations, and used it to define common Cloud Computing usage patterns. We have summarized the national laws of major countries related to data protection, then we assessed the defined use cases against evaluation criteria derived from legislation for the data processing necessary for legal compliance. To clarify and exemplify legal compliance in the identified usage patterns, we considered the Data Protection Directive of the European Union, which is a commonly accepted and influential directive in the field of data processing legislation. We can conclude that data protection is a far more complex problem in Cloud systems compared to traditional ICT systems.

We have shown that identifying the relevant roles and the national law applicable to common Cloud Computing use cases is not straightforward. We identified dynamic roles changing as actions are initiated among the corresponding providers, which may also affect the national law applicable during service execution. Cross-continental cases may further complicate the situation, e.g., when a US company stores data to a Cloud provider in France the French law will apply, and the exportation of the processed data back to the US will be restricted or prohibited [30]. Nevertheless, according to the discussion in [17], if the European organization complies with the Safe Harbor privacy principles for the protection of personal data, the data is allowed to be transferred from a Member State to the US.

9 Future Research Directions and Conclusions

As we have seen in this chapter, data protection in Clouds is an important issue to consider. While there are national laws regulating data protection all around the world already, not all of them are detailed enough to be applicable to or sufficient

to cover specific Cloud usage scenarios involving multiple jurisdictions. Regarding open research issues and future research directions, on one hand, the Cloud community has to support the revisions of the relevant legislation applying to Clouds by revealing their latest technological solutions and properties, on the other hand, appropriate software frameworks and solutions need to be developed that can enforce the legal compliance of data protection in Cloud federations spanning over several countries, even continents. To achieve this, a similar case study would be needed that details the application of legislation in other regions, particularly in the US.

Acknowledgments The research leading to these results has received funding from the SCI-BUS project of the FP7 Capacities Programme under contract RI-283481.

References

1. 4CaaSt EU FP7 project website (2012). Available at http://4caast.morfeo-project.org
2. Amazon Web Services (2012). Available at http://aws.amazon.com/
3. Birnhack MD (2008) The EU data protection directive: an engine of a global regime. Tel Aviv University Law Faculty Papers, no 95, Tel Aviv University Law School
4. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging it platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst 25(6):599–616
5. Bygrave LA (2000) European data protection. Determining applicable law pursuant to European data protection legislation. Comput Law Secur Rep 16(4):252–257
6. Catteddu D, Hogben G (2009a) Cloud computing risk assessment: benefits, risks and recommendations for information security, ENISA report. Available at http://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-risk-assessment/at_download/fullReport
7. Catteddu D, Hogben G (2009b) An SME perspective on cloud computing. Cloud computing–SME survey, ENISA report. Available at http://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-sme-survey/at_download/fullReport
8. Cloud Computing Use Case Discussion Group (2009). Availble at http://www.scribd.com/doc/179-29394/Cloud-Computing-Use-Cases-Whitepaper
9. COM (2012a) 09 final, Communication from the commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions Safeguarding Privacy in a connected world a European data protection framework for the 21st Century, 25/01/2012
10. COM (2012b) 10 final, Proposal for a directive of the European Parliament and of the council on the protection of individuals with regard to the processing of personal data by competent authorities for the purposes of prevention, investigation, detection or prosecution of criminal offences or the execution of criminal penalties, and the free movement of such data, 25/01/2012
11. COM (2012c) 11 final, Proposal for a regulation of the European Parliament and of the council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (general data protection regulation). Brussels 25(01):2012
12. Communication from the Commission to the European Parliament (2011) The council, the European economic and social committee and the committee of the regions a digital agenda for Europe, COM (2010) 0245 final
13. Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions Unleashing the Potential of Cloud Computing in Europe, COM (2012) 529 final, Brussels, 27.9.2012
14. Council of the European Union (2012) Data Protection package, report on progress achieved under the Cyprus presidency, 16525/1/12 REV 1, 3rd Dec 2012

15. Data Protection Working Party (2010a) Opinion 1/2010 on the concepts of "controller" and "processor". Ref WP 169. Available at http://ec.europa.eu/justice/policies/privacy/docs/wp-docs/2010/wp169_en.pdf
16. Data Protection Working Party (2010b) Opinion 8/2010 on applicable law. Ref WP 179. Available at http://ec.europa.eu/justice/policies/privacy/docs/wp-docs/2010/wp179_en.pdf
17. Decision Commission, no. 2000/520/EC of 26 July, (2000) pursuant to Directive 95/46/EC of the European Parliament and of the Council on the adequacy of the protection provided by the safe harbour privacy principles and related frequently asked questions issued by the US Department of Commerce. Official J L 215:7–47
18. Decision of the EEA Joint Committee (2000) No. 83/1999 of 25 June 1999 amending Protocol 37 and Annex XI (Telecommunication services) to the EEA Agreement, Official J 296:41, Nov 2000
19. U.S. Department of Justice website (2012). Available at http://www.justice.gov/opcl/privacyactoverview2012/1974intro.htm
20. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, Official J L 281:31–50, Nov 1995
21. DMTF white paper no. DSP-IS0101 (2009) Interoperable clouds, a white paper from the open cloud standards incubator 1.0. Available at http://www.dmtf.org/sites/default/files/standards/documents/DSP-IS0101_1.0.0.pdf
22. Draft opinion of the Committee on the Internal Market and Consumer Protection (IMCO) (2012) For the Committee on Civil Liberties, Justice and Home Affairs on the proposal for a regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation) (COM(2012) 0011–C7-0025/2012 - 2012/0011(COD)) Rapporteur: Lara Comi, 25.9.2012
23. Draft opinion of the of the Committee on Legal Affairs (JURI) for the Committee on Civil Liberties, Justice and Home Affairs on the proposal for a regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (COM(2012) 0011–C7-0025/2012 - 2012/0011(COD)) Marielle Gallo, Rapporteure (18.10.2012)
24. eBay Inc website (2012). Available at http://www.ebay.com/
25. EC Press release (2012) Commission proposes a comprehensive reform of data protection rules to increase users' control of their data and to cut costs for businesses, European Commission, IP/12/46, 25/01/2012. available at europa.eu/rapid/pressReleasesAction.do?reference=IP/12/46
26. European Commission (2011) Information society and media directorate-general, converged networks and services, software and service architectures and infrastructures, cloud computing: public consultation report, Brussels, 5th Dec 2011. Available at http://ec.europa.eu/information_society/activities/cloudcomputing/docs/ccconsultationfinalreport.pdf
27. European Commission (2012) European cloud computing strategy (CCS), 2012. Available at http://ec.europa.eu/information_society/activities/cloudcomputing/cloud_strategy/index_en.htm
28. Ferrer AJ et al (2012) OPTIMIS: a holistic approach to cloud service provisioning. Future Gener Comput Syst 28:66–77
29. freedominfo.org (2012) The global network of freedom of information advocates website. Available at http://www.freedominfo.org/regions/east-asia/china/
30. Gellman R (2009) Privacy in the clouds: risks to privacy and confidentiality from Cloud Computing. World Privacy, Forum, 23 Feb 2009
31. Google Apps for Business (2012). Available at http://www.google.com/apps
32. Greenleaf G (2012) Global data privacy laws: 89 Countries, and accelerating, privacy laws and business international report, Issue 115, special supplement, Queen Mary School of Law Legal Studies Research Paper No 98/2012, Feb 2012

33. Kroes N (2012) Setting up the European cloud partnership, World Economic Forum, Davos, Switzerland, 26th Jan 2012
34. Liu F, Tong J, Mao J, Bohn RB, Messina JV, Badger ML, Leaf DM (2011) NIST cloud computing reference architecture, NIST Special Publication 500–292. Available at http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909505
35. Marosi AC, Kecskemeti G, Kertesz A, Kacsuk P (2011) FCM: an architecture for integrating IaaS cloud systems. In: Proceedings of the second international conference on cloud computing, GRIDs, and virtualization (Cloud Computing (2011) IARIA. Rome, Italy, pp 7–12
36. Mell P, Grance T (2011) The NIST definition of cloud computing, NIST special publication 800–145. Available at http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf
37. NextLabs website on Japan data protection (2012). Available at http://www.nextlabs.com/html/?q=japan-data-protection-acts-pipa-pipl
38. Office of the Austrialian Information Commissioner (OAIC) website (2012). Available at http://www.oaic.gov.au/about/what.html
39. Office of the Privacy Commissioner of Canada (OPC) website (2012). Available at http://www.priv.gc.ca
40. OPTIMIS FP7 project deliverable no. D7.2.1.1, Cloud Legal Guidelines (2010). Available at http://www.optimis-project.eu/sites/default/files/D7.2.1.1OPTIMISCloudLegalGuidelines.pdf
41. Orito Y, Murata K (2005) Privacy protection in Japan: cultural influence on the universal value. In: Electronic proceedings of Ethicomp'05
42. Public Consultation on Cloud Computing by the European Commission (2011). Available at http://ec.europa.eu/your-voice/ipm/forms/dispatch?form=cloudcomputing&lang=en
43. Rochwerger B et al. (2009) The reservoir model and architecture for open federated cloud computing. IBM J Res Dev
44. Safe Harbor website of export.gov. (2012). Available at https://safeharbor.export.gov
45. Schubert L, Jeffery K (2012) Advances in clouds–research in future cloud computing, report from the cloud computing expert working group meeting. Cordis (Online), BE: European Commission, 2012. Available at http://cordis.europa.eu/fp7/ict/ssai/docs/future-cc-2may-finalreport-experts.pdf
46. Schubert L, Jeffery K, Neidecker-Lutz B (2010) The future of cloud computing–report from the first cloud computing expert working group meeting. Cordis (Online), BE: European Commission, 2010. Available at http://cordis.europa.eu/fp7/ict/ssai/docs/Cloud-report-final.pdf
47. Svantesson D, Clarke R (2010) Privacy and consumer risks in cloud computing. Comput Law Secur Rev 26:391–397
48. Vaquero LM, Rodero-Merino L, Caceres J, Lindner M (2008) A break in the clouds: towards a cloud definition. SIGCOMM Comput Commun Rev 39(1):50–55
49. What is EU Data Protection Directive 95/46/EC? (2008). Available at http://searchsecurity.techtarget.co.uk/definition/EU-Data-Protection-Directive, http://Whatis.com
50. Whittaker Z (2011) Safe harbor: why EU data needs 'protecting' from US law. Available at http://www.zdnet.com/blog/igeneration/safe-harbor-why-eu-data-needs-protecting-from-us-law/8801
51. Winton A, Zhang A, Innes-Stubb S, Xu L (2012) Data protection and privacy in China, White and Case Technology Newsflash
52. Wong R (February 2011) Data protection: the future of privacy. Comput Law Secur Rev 27(1):53–57
53. Zimory GmbH website (2012). Available at http://www.zimory.com/

Index

S. Nepal and M. Pathan (eds.), *Security, Privacy and Trust in Cloud Systems*,
DOI: 10.1007/978-3-642-38586-5, © Springer-Verlag Berlin Heidelberg 2014