编号：（ ）字 号

本科生毕业设计（论文）

面向云环境基于属性的访问控制

策略设计与实现

任玮 08123493

电子信息科学与技术2012-04

题目：

姓名： 学号：

班级：



二〇一六年六月

中 国 矿 业 大 学

本科生毕业设计

姓 名： **任玮** 学 号**： 08123493**

学 院： **计算机科学与技术**

专 业： **电子信息科学与技术**

设计题目：**面向云环境基于属性的访问控制策略设计与实现**

专 题：

指导教师： **陈岱** 职 称： **副教授**

**2016**年 **6**月 徐州

中国矿业大学毕业设计任务书

学院 计算机科学与技术 专业年级 信科12-4 学生姓名 任玮

**任务下达日期：2016年1 月19日**

**毕业设计日期：2016年1月19日至2016年6月3日**

**毕业设计题目：面向云环境基于属性的访问控制策略设计与实现**

**毕业设计专题题目：**

**毕业设计主要内容和要求：**

毕业设计（论文）可以全面地、综合地展示和检验毕业生掌握所学理论知识的程度和运用所学知识解决实际问题的能力，同时训练学生进行科学研究的基本功，培养学生综合运用所学知识独立地分析问题和解决问题的能力，培养刻苦钻研、勇于探索的精神。

云计算是一种新兴的服务模式，以租赁的方式向用户提供软硬件资源。但是云环境为用户带来便捷的同时，也存在着安全隐患并制约其发展应用。访问控制技术是保障云环境下系统资源安全的核心技术之一。基于属性的访问控制是开放环境下复杂信息系统中较理想的访问控制机制，可扩展性强，细粒度访问、灵活度高等优点。

本毕业设计要求：

1. 调研了解云计算服务的架构和基于属性的访问控制机制。
2. 运用主流的实现模型和方法，设计实现基于属性的访问控制系统。包括基于属性的加解密算法，访问控制策略的设计制定，以实现对敏感数据的加密储存和访问控制
3. 在实验室搭建的数据平台上进行相关技术验证。

4、完成毕业设计论文。

院长签字： 指导教师签字：

中国矿业大学毕业设计指导教师评阅书

指导教师评语（①基础理论及基本技能的掌握；②独立解决实际问题的能力；③研究内容的理论依据和技术方法；④取得的主要成果及创新点；⑤工作态度及工作量；⑥总体评价及建议成绩；⑦存在问题；⑧是否同意答辩等）：

成 绩： 指导教师签字：

年 月 日

中国矿业大学毕业设计评阅教师评阅书

评阅教师评语（①选题的意义；②基础理论及基本技能的掌握；③综合运用所学知识解决实际问题的能力；③工作量的大小；④取得的主要成果及创新点；⑤写作的规范程度；⑥总体评价及建议成绩；⑦存在问题；⑧是否同意答辩等）：

成 绩： 评阅教师签字：

年 月 日

中国矿业大学毕业设计答辩及综合成绩

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 答 辩 情 况 | | | | | |
| 提 出 问 题 | 回 答 问 题 | | | | |
| 正 确 | 基本  正确 | 有一般性错误 | 有原则性错误 | 没有  回答 |
|  |  |  |  |  |  |
| 答辩委员会评语及建议成绩：  答辩委员会主任签字：  年 月 日 | | | | | |
| 学院领导小组综合评定成绩：  学院领导小组负责人：  年 月 日 | | | | | |

摘 要

云计算作为目前计算机领域的研究热点之一，已经受到了广泛的关注和研究，具有多租户、灵活快速和易扩展等特点。但云计算在为用户提供快捷、方便、可扩展的服务同时，其安全问题也日益严重，特别是用户存储在云端数据的安全性急需加强，这给用户隐私保护带来了极大的安全隐患。访问控制技术尤其是基于属性的访问控制技术是解决云存储中数据安全和隐私保护的核心技术。但在复杂的云计算环境中，传统的访问控制模型由于其局限性已不能使用。因此如何设计一套适用于云环境下的访问控制方案，为用户的隐私数据提供安全、动态、细粒度、可扩展的的访问控制显得十分重要。

本文的主要研究内容是云环境下的访问控制技术，特别是基于属性的访问控制技术如XACML、CPABE等框架模型。基于属性的访问控制(ABAC)是通过主体、客体和环境的属性来作为授权决策的基础，即所有的实体都采用属性来进行描述，ABAC具有聚合度高、灵活性高、可扩展性强和支持匿名访问等优点。云计算环境有三类实体：云用户、云服务提供商和数据拥有者。云用户向云服务提供商中的云资源发起访问请求，而服务提供商所提供的资源可能属于不同的数据拥有者和处于不同的逻辑安全域，某一逻辑安全域中的实体可能是恶意用户实体或者恶意云服务实体。因此面对这种特殊的复杂云计算环境构成，现有访问控制模型很难适用。在调研学习相关的理论背景知识后，分析了现有云计算系统中数据存储存在的不足，进而建立了基于属性的云计算访问控制系统的设计思路和整体结构，并完成其原型测试系统，包括了属性输入、策略生成、请求生成、评估系统、加密系统及云存储模块等基本云计算访问控制主功能。

本文提出了适用于云计算环境下的基于属性的安全增强访问控制子系统。通过共用属性集，将基于属性的加密机制(CPABE)与XACML框架相结合，由XACML框架实现数据细粒度的基于属性的访问控制，由CPABE保证敏感数据的机密性。通过设计实验场景、模拟云计算环境进行的仿真实验，测试了访问控制过程的决策结果和流程，并经过对其理论证明、决策时间的分析及决策结果的对比，验证了该方案的高效性、灵活性和安全性。

关键词：云计算；访问控制；基于属性加密；XACML

ABSTRACT

As one of the hotspots in the current Computer Science field, Cloud Computing has been paid attention to and researched widely. Although Cloud Computing provides users with fast, convenient, scalable service, its security issues are increasingly serious, especially the data which cloud users stored in the cloud server need to be strengthen urgently. Because it brings users privacy protection great security risk. Access Control technology, particularly based Attributes, is the key solution to the data security stored in the cloud and protect users privacy. In the complex cloud computing environments, traditional access control models can’t be used again for its limitations. Thus providing a secure, dynamic, fine-grained, extensible, attribute-based access control model for users, and designing a suitable access control model for cloud computing is very important and urgent.

The main research content of this paper.is the Access Control technology in cloud computing environment, especially attribute-based access control model, such as XACML and CPABE frameworks. The attributes of subjects, objects, environments is used as the basis of authorization decision for attribute-based access control(ABAC), that is all entities is described by attributes. ABAC is flexible, scalable, supporting anonymous access and has a high degree of polymerization. There are three kinds of entities for Cloud computing environment: cloud users, cloud service and data owners. Users initiate access requests for the cloud resource in service, but the resource provided by cloud service may belong to different owners and locate at different logical security domain, some logical security domain entity may be a malicious user entity or cloud service entity. Therefore facing this particular and complex cloud computing environment constitution, existing access control model is difficulty to apply. After the research of relevant theory and background, we analysis the deficiencies of prior cloud computing data storage, build a cloud computing access control system’s overall structure and some designing ideas and finally complete its prototype test system, whose main implementation functions are attributes generating, policy creation, request creation, evaluation system, encryption module and cloud computing storage module.

In this paper, we propose a new enhanced security access control system based on attributes in cloud computing. By using common sets of attributes, we combine the attribute-based encryption mechanisms(CPABE) with XACML framework in our access control model. XACML framework ensures the fine-grained, efficient, flexible access control based attributes for data., while the confidentiality of the sensitive date in cloud is guaranteed by CPABE. Experiments show that this control access system is high-performance, flexible and secure.

**Keywords**：Cloud Computing; Access Control ; Attribute-based encryption ; XACML

目 录

[第一章 绪论 1](#_Toc453068446)

[1.1研究背景 1](#_Toc453068447)

[1.1.1云计算及其面临的安全问题 1](#_Toc453068448)

[1.1.2云计算环境下的访问控制 2](#_Toc453068449)

[1.2国内外研究现状 2](#_Toc453068450)

[1.2.1传统访问控制技术研究现状 2](#_Toc453068451)

[1.2.2基于属性的访问控制研究现状 3](#_Toc453068452)

[1.3论文选题的意义 4](#_Toc453068453)

[1.4主要工作思路与组织结构 5](#_Toc453068454)

[第二章 相关基础知识介绍 6](#_Toc453068455)

[2.1相关技术调研与选型 6](#_Toc453068456)

[2.2 XACML 8](#_Toc453068457)

[2.2.1 XACML访问框架与流程 8](#_Toc453068458)

[2.2.2 XACML策略语言模型 9](#_Toc453068459)

[2.2.3 XACML开源框架调研分析与选型 10](#_Toc453068460)

[2.3基于属性的加解密(ABE) 11](#_Toc453068461)

[2.3.1属性加密基础知识 11](#_Toc453068462)

[2.3.2密文策略的属性加密方案 13](#_Toc453068463)

[2.3.3 CPABE访问控制系统架构 13](#_Toc453068464)

[2.4 OpenStack云平台架构 14](#_Toc453068465)

[2.5 web开发技术介绍 15](#_Toc453068466)

[2.6本章小结 16](#_Toc453068467)

[第三章 基于属性的访问控制系统设计方案 17](#_Toc453068468)

[3.1系统总体设计思路 17](#_Toc453068469)

[3.2系统关键方案设计 18](#_Toc453068470)

[3.2.1构建通用属性集 18](#_Toc453068471)

[3.2.2 DO申请数据托管服务与CSP加密数据服务 19](#_Toc453068472)

[3.2.3 CSP构建访问控制策略 19](#_Toc453068473)

[3.2.4 User申请访问数据 20](#_Toc453068474)

[3.3系统整体设计流程图 20](#_Toc453068475)

[3.4本章小结 22](#_Toc453068476)

[第四章 基于属性的访问控制系统实现方案 23](#_Toc453068477)

[4.1系统部署 23](#_Toc453068478)

[4.1.1系统软硬件环境 23](#_Toc453068479)

[4.1.2系统部署实体机拓扑图 23](#_Toc453068480)

[4.2 OpenStack平台设计与实现 23](#_Toc453068481)

[4.3系统实现功能 24](#_Toc453068482)

[4.4访问控制系统主要模块设计与实现 25](#_Toc453068483)

[4.4.1数据库设计与实现 25](#_Toc453068484)

[4.4.2用户登录与注册模块设计与实现 26](#_Toc453068485)

[4.4.3文件上传、浏览模块设计与实现 27](#_Toc453068486)

[4.4.4访问策略生成、请求生成模块设计与实现 29](#_Toc453068487)

[4.4.5 PDP、PEP模块设计与实现 30](#_Toc453068488)

[4.4.6 CPABE数据加解密模块设计与实现 32](#_Toc453068489)

[4.5本章总结 33](#_Toc453068490)

[第五章 系统功能测试与安全性分析 34](#_Toc453068491)

[1.系统功能测试 34](#_Toc453068492)

[2.系统安全性分析 39](#_Toc453068493)

[第六章 总结与展望 40](#_Toc453068494)

[6.1总结 40](#_Toc453068495)

[6.2展望 40](#_Toc453068496)

[参考文献 42](#_Toc453068497)

[附录：核心代码 44](#_Toc453068498)

[附录一 文件浏览代码 44](#_Toc453068499)

[附录二 策略生成核心代码 44](#_Toc453068500)

[附录三 请求结果生成核心代码 45](#_Toc453068501)

[附录四 CPABE加解密算法核心代码 45](#_Toc453068502)

[附录五 notePolicy.xml文件 47](#_Toc453068503)

[翻译 49](#_Toc453068504)

[英文原文 49](#_Toc453068505)

[中文译文 57](#_Toc453068506)

[致 谢 64](#_Toc453068507)

第一章 绪论

1.1研究背景

1.1.1云计算及其面临的安全问题

云计算(Cloud Computing)是一种以互联网技术为基础，为云用户或其他设备按需供给所需资源数据的服务模式，是分布式技术、计算机并行计算、网络安全存储、设备虚拟化等传统计算机互联网技术不断发展融合创新的结果。云计算在由大规模计算机构成的资源池中合理分配相关运算任务。云用户按需获取其云平台提供的CPU、存储空间等云计算服务，计费标准是按照用户资源使用量。作为当前信息领域技术的前沿热点，其已经在学术界、社会和产业界引起了广泛的关注研究，其中以苹果、IBM、微软和阿里巴巴等为代表的科技企业已经在云计算领域取得了非常大的发展和研究成果。

以当前云计算相关领域技术的发展来看，云计算具有很多优势,如超大规模、设备虚拟化、可靠性极高、通用性强、良好的可扩展性、用户按需计费、成本低等。

云计算的体系结构一般分为三层，分别为：物理层、虚拟层、服务层，见图1.1。通常意义上的云计算服务——基础设施即服务（IaaS），平台即服务（PaaS）和软件即服务（SaaS）三层构成[1]构成了云计算中的最上层。IaaS(Infrastructure-as-a-Service)：基础设施即服务。用户通过云计算获得所需计算机硬件基础服务；例如：硬件服务器租用。PaaS(Platform-as-a- Service)：平台即服务。PaaS一般意义上是指将软件开发平台作为一种服务，在云中为用户提供开发所需的各种服务。SaaS(Software-as-a- Service)：软件即服务。它通过云平台向用户提供软件服务。

图1.1 云计算服务架构

由于云计算技术不断进步，多云平台的相继开放和市场容量爆炸增长，用户对云计算的各种需求日益提高。由于云计算特有的数据外包、资源虚拟化、多租户共享和跨域共享的特点，这就对其安全设计提出了非常大的挑战[2]，如今云计算的安全问题已经成为了制约其进一步发展壮大的拦路虎。最近几年很多著名的云计算服务供应商如IBM、 Google等都出现了数据资源和用户隐私信息泄露等安全问题，导致了非常严重社会后果和经济损失；2015 年Verizon云出现的问题再次让公众对云计算安全感到担忧。经权威调查显示，绝大部分的用户对于在云环境下存储敏感数据存在不信任感。

云计算安全联盟CSA在2015年的报告强调了云计算领域出现急待解决的安全问题并为其解决和发展提出了建议。云安全联盟总结的九大云安全威胁有：Data breaches; Data loss; Account or service traffic hijacking (账号劫持或服务流量劫持);Insecure interfaces and APIs (不安全接口/应用程序接口); Denial of service (拒绝服务攻击);Malicious insiders;Abuse of cloud services(滥用云服务); Insufficient due diligence;Shared technology vulnerabilities (共享技术中的漏洞)。

在这份报告中，如何防止云端数据被破坏或者丢失和非法获取占到了很重的篇幅，是此次报告关注的焦点之一。而访问控制技术对于确保云平台中的资源共享和权限管理发挥着举足轻重的作用。所以怎样为云计算设计一个支持细粒度访问、安全、高效的访问控制框架就变得非常关键[3]。

1.1.2云计算环境下的访问控制

访问控制(Access Control)是根据访问者（主体）身份信息或者其他属性来限制其对数据资源（客体）信息安全技术[4]。其主要有三个作用，分别是防止非法用户访问受保护资源、授权合法用户访问保护资源和禁止合法用户对保护资源的非授权操作。访问控制技术是保障计算机系统可用性、安全性和完整性的重要基础，也是网络安全和敏感资源保护的关键技术。访问控制基本过程是访问主体依据访问策略或定义的权限对来访问客体进行权限控制，一般包括以下基本要素：主体、客体、行为、策略；两个基本任务是：1)认证访问用户的身份，2)决定该用户是否有权限访问对某种数据资源和进行某些操作。经过很多年的研究发展，访问控制技术得到了很大的改进，对信息安全作出了重要的贡献。但是随着云计算技术的不断发展和广泛应用，之前的访问控制技术存在着或多或少的缺陷，新的需求变化也对传统的访问控制信任模型机制提出了新的更高的要求[5]。

1.2国内外研究现状

1.2.1传统访问控制技术研究现状

访问控制技术是为解决用户访问计算机中存储的数据资源的问题和管理访问权限而被提出的。在早期的访问控制中，主要技术有：自主访问控制类型[6]（Discretionary Access Control，DAC）、强制访问控制类型[7]（Mandatory Access Control，MAC）和基于角色的访问控制类型[8]（Role Based Access Control）。

(1)自主访问控制(DAC)

自主访问控制是指授权主体即DO(数据拥有者)拥有对其所拥有资源即客体的分配或者撤销其他用户的访问权限。在采用自主访问控制的系统内，一般是指定的某些特权用户（一般指管理员）或用户组修改管理该策略列表。一般情况下采用访问控制列表ACL（Access Control List）管理不同用户对数据的拥有的访问权限，一张表中有着不同用户对这个数据的操作权限。同时DAC存在以下缺点：客体复杂问题，即不能对较为复杂的数据资源实现有效保护；访问权限的连续委托问题，即权限不能连续分配；抵御木马病毒攻击。

因此在云环境中，数据的访问请求规模非常大，数据资源种类多，常常有连续委托的需求，网络安全问题也很严峻。这样会使系统资源开销增大，效率降低，维护更加困难，灵活性不好，因此DAC不适合当今大规模网络环境，如云计算环境等。

(2)强制访问控制(MAC)

强制访问控制模型（MAC Model）是一种通过安全级别完成数据资源访问控制的技术。MAC要求超级管理员或者系统对主体（访问者）和客体（数据）设置相关安全级别标签。MAC决定是否对访问请求授权是通过系统对主、客体的安全级别进行比较来实现的。安全等级设置的标签具有偏序关系，以 SC 表示安全类型是一个偏序关系。TS 对应绝密级别，S对应秘密级别，信息安全级别上绝密要高于秘密，用偏序关系表示为SC(s)≥SC(o)。一般情况下DAC用来保护系统资源，MAC用来作为补强的的安全措施，保证了对数据的安全访问控制。因此强制访问控制模型的适用场景一般是将数据分密级和类进行管理的对安全性要求很高的领域，如：金融银行和军事安全[9]等。

但是，MAC强大的靠安全级别实现的安全特性造成了由其构成的访问控制灵活性和可伸缩性表现一般。在如今云计算分布式情况下，访问请求规模大，需要动态访问，MAC难以保证。而且在分布式大规模系统中，数据访问用户通常不处于同一个逻辑层次（即不再同一个逻辑域中），域中每个用户设置的访问权限级别不确定。因此，MAC 的应用场景较少。

(3)基于角色的访问控制(RBAC)

RBAC是上世纪70年代被提出并很快得到了广泛的关注。1992年Ferraiolo在AC模型中引入了角色概念，第一次提出了于角色的访问控制（RBAC）。RBAC核心思想是了用户和角色相关，角色和权限相关，将用户和权限相分离，用户通过拥有不同角色来实现拥有不同权限。在所有访问控制模型中，RBAC由于其很多优秀特性而被广泛使用和研究，较为成熟。后来Sandhu 等人在 Ferraiolo 和Kuhn 的研究基础上提出了 RBAC96，RBAC99等模型。为了规范RBAC使用标准，NIST起草并公布了RBAC建议，其中规定了RABC的基本模型和功能规范。国内也有很多关于 RBAC 模型的研究，例如基于多亲树的角色可视RBAC等。

但在云环境下,一般情况下访问数据的用户很有可能不是处在同一个安全域内，所以对应基于角色的访问控制模型，跨多安全域进行角色的细粒度划分和访问策略的安全管非常困难，安全性难以得到保障。因此云计算对于访问控制提出的新的需求，基于角色访问控制模型由于其固有缺陷,很难满足复杂网络环境下于授权的细粒度、安全性和动态性要求等。但是2005年提出的基于属性的访问控制模型可以很好地解决以上问题。

1.2.2基于属性的访问控制研究现状

传统的访问控制模型，如DAC、RBAC、MAC等，其设计思想一般是针对同一个安全域，用户的访问标识在安全域是唯一固定的，在云环境中应用这些AC模型会有动态扩展性和灵活性差的问题。但是在云计算环境中，客户请求或者服务器可能不处于同一个信任域中，不同的客户端和云服务提供商CSP频繁交互，而它们一般处于不同的安全域内，不能知道对方的全部信息，这样传统的访问控制模型对于于云计算平台不再适用[10]。而基于属性的访问控制（Attribute Based Access Control，ABAC）为实现云计算环境下的数据的访问控制提供了一种合适的新方法[11]。其数据访问控制策略的制定由数据拥有者制定，具有很大的灵活性，按照匹配主体、环境、资源、动作等属性值来决定是否决定为行为授权。这样就解决了云环境下诸如用户数据的动态扩展和细粒度访问权限问题。与传统访问控制模型将数据保存在可信服务器，访问权限全部由服务器控制相比，基于属性的访问控制可以保证即使将用户数据存储在并不完全可信的服务器上，但由于其经过了加密只有满足策略属性要求的用户才能进行解密数据并对其访问。

为了解决云计算中数据的隐私保护和有效共享问题，近年来己经有很多基于属性的加密技术已经被提出来。最早是Sahai等人于2005年提出基于属性的加密方案[12]，由于其良好特性，后来出现了很多基于其的ABE加密访问控制控制研究[13-15]。在基于身份加密方案中，身份概念看作是一个或多个属性组成的身份属性集合,并并将数据访问控制结构与定义的属性集合相结合,这样就实现了对密文或者秘钥的基于属性访问控制。由于基本ABE具有无法支持灵活的访问控制策略定义的不足,后来Goyal[16]等人在2006年提出了基于密钥策略的属性加密方案(KP-ABE),KPABE将DO制定的访问策略与用户解密秘钥相关联，用户属性集嵌入到密文中。为了使密钥策略能够描述更加复杂的情景，2007年Ostrovsky和Waters等人提出非单调性的(non-monotonic)基于属性的ABAC解决方案[17]。而为了解决KP-ABE中的存在的问题,Bethencourt[18]等人于2007年提出了密文策略的基于属性的加密方案(CP-ABE),用户的秘钥与用户通过认证的属性集相关，密文与DO制定的访问策略相关联，用户属性满足访问策略就能进行解密。由于云计算环境中数据资源的高度集中,对动态扩展和开发性要求高等特点，CP-ABE比KP-ABE更适用于云计算中的访问控制系统,同时对于密钥管理也变得方便有效快捷。同时在国内，许多学者对云计算安全问题、ABE加解密与访问控制都非常关注，展开了很多研究工作。例如冯登国和张敏等人对云安全做了综述性研究[5]，全面介绍了当前云安全遇到的问题和可能的解决方案。ABE方面，广州大学李进等[19, 20]提出了一种对于基于属性密码体制中的密钥滥用问题的解决方案，和ABE外包技术[21]。上海交通大学梁晓辉等人[22, 23]提出了一种基于属性的代理重加密定义。

1.3论文选题的意义

作为当今信息科学的研究热点之一，云计算通过虚拟化租赁方式为用户提供软硬件服务，给人们带来了前所未有的方便和极大提升生产效率的同时，也存在着安全问题并制约其进一步发展。而访问控制技术能够有效对用户权限管理，保证了存储数据的安全性，是整个系统安全保障的核心技术。而传统的访问控制技术很难适用于分布式环境，特别是不能保证多域环境下对数据资源的交互操作与安全共享安全。

但是近几年出现的基于属性的访问控制（ABAC）模型由于其具有的灵活性高、可扩展，支持细粒度访问等非常优秀的特性，比较适用于复杂的分布式云环境。ABAC的基本原理是通过主体、环境、资源、动作的属性与访问策略是否匹配来决定是否对请求进行授权，访问策略由数据拥有者制定，就就解决了跨域访问控制中用户动态扩展和灵活访问等传统访问控制模型较难解决的问题。

本毕设的意义是考虑到目前较为成熟的访问控制系统不能很好地适用于云环境，需要设计一种新的AC系统来满足云计算对于访问控制所要求的多域访问、大规模用户扩展和动态细粒度访问等需求，而最近提出ABAC模型由于其面向属性的特质非常适用于云计算环境，因此设计实现面向云数据下基于属性的访问控制系统非常具有研究价值和实现意义。

1.4主要工作思路与组织结构

本毕设论文的主要工作思路和研究内容都是以面向云计算环境下基于属性的访问控制设计与实现为中心所展开的。主要研究学习比较了基于属性的访问控制模型，基于属性的加解密算法，XACML访问控制框架，Web技术开发，Openstack云平台搭建及使用方法。

主要思路是：在传统的访问控制系统中，信息都以明文的形式存储与服务器，用户通过认证后可以获得自身对应权限的数据信息。但是这个方案存在以下几个问题：

(1)数据直接以明文形式全部存在于不可信服务器是不安全的；同时数据全部加密是没有必要的，也会对资源产生浪费，降低效率。

(2)数据的权限管理实现现较为粗糙，不能实现对数据的细粒度访问控制。

(3)数据访问策略较为简单，很多情况下需要更为复杂的策略描述。

因此针对以上问题，本毕设根据云环境对于访问控制的要求，参考传统的访问控制模型和，提出了一种基于XACML授权框架和CPABE加解密相结合的基于属性的云环境访问控制系统解决方案。

本毕设论文文共分为六部分，各部分内容如下：

第一章首先介绍了云计算的背景、内涵、优势和面临的安全问题，接着简单说明了传统访问控制模型与基于属性的访问控制模型的主要原理、发展历史及国内外研究现状，并分析了各自在云环境中的优缺点，最后指出了课题的研究背景、目的和意义。

第二章研究介绍了访问控制相关选型和实例，说明了XACML框架和授权流程，比较了几种不同的开源框架并最终选择sun\_xacml，还有关于基于属性的加解密机制的基础知识、加密过程及一般框架介绍，最后是简要补充了本系统的部署平台OpenStack和Web开发技术。

第三章详细介绍了本文所设计的访问控制系统的总体设计思路，用到的关键方案设计，如构建通用属性集、上传数据、访问数据等的原理和设计方案，最后仔细说明了系统整个设计流程图，解释了整个系统设计的步骤。本系统使用的是sun\_xacml构建访问控制授权框架，jpbc实现cpabe加密。

第四章具体说明了访问控制系统的实现设计方法，包括系统部署与搭建情况，功能介绍和主要模块的详细构建过程等内容。

第五章介绍了对构建的基于属性的访问控制系统的相关测试与分析过程，包括安全性测试和系统功能测试等。

第六章对本文所设计实现的基于属性的访问控制系统进行了总结，并对下一步的工作进行了展望。

第二章 相关基础知识介绍

为了对后文说讲解内容有更深的理解和掌握，本章主要介绍了访问控制子系统主要用到的关键技术。比较详细地讲解了XACML，ABE，ABAC，OpenStack的背景、理论和模型等相关知识。

2.1相关技术调研与选型

访问控制是为了控制用户对数据访问权限借以保证数据的安全性的关键技术。在强调用户隐私和安全的当今社会，访问控制技术已经被广泛使用。出现的各种访问控制框架和模型，各有优缺点，被应用在不同适合的场景，具体有自主访问控制类型、强制访问控制和基于属性的访问控制模型。如今比较成熟的是将其综合应用起来构建一整套访问控制系统。而基于角色的访问控制一般应用在较为封闭固定的管理系统中，而云计算这种对动态可扩展性要求较高的较为不合适，现结合公司实例简单对云计算访问控制相关模型介绍如下。

1．百度云访问控制框架

百度云访问控制服务主要采用两种方式来对存储在云端的数据进行访问控制。一是采用URL签名来对访问者身份进行识别并验证；二是采用ACL访问控制列表来规定资源和资源桶的访问权限。其中URL签名格式如下Sign = Flag:AccessKey:signature,其中signature由用户通过认证中心申请的公私钥与访问内容所决定，云服务器根据Flag计算得到用户签名并与请求相匹配，相同则通过验证。之后用户通过ACL来管理其资源的访问权限，通过设置Policy来对权限进行再分配。鉴权流程图如图2.1所示。



图2.1 鉴权流程图

2.阿里云访问控制框架

通过对阿里云ODPS进行分析学习，了解其访问控制的基本流程，发现其核心权限控制功能是通过ACL与Capability相结合来实现的

阿里云的服务框架如图2.2所示，主要由四层组成。第一层客户端层主要功能是面向用户的功能，如API、SDK等；第二层接入层主要是完成对用户的认证过程；第三层逻辑层完成类似于ACL的资源访问控制功能和capability票据传递功能；第四层存储计算层完成对数据和资源的存储功能。

具体控制过程如下：首先解析来自客户端的query，得到请求中的主客体，并对主体进行身份验证，检查客体是否存在；接着应用ACL进行权限检查，查看主体是否对客体具有相关权限；然后生成执行计划；最后将上层ACL控制与下层Capability控制结合，生成权限票据，并具有过期时间。

图2.2 阿里云服务框架

通过对实际生活中访问控制模型的调研发现，企业一般都使用较为成熟稳定的ACL自主访问控制模型。但是面对发展越来越迅猛的云计算环境，考虑到其对安全性、高扩展性，和跨域访问的要求，传统的这些访问控制模型不能很好地应用到其中。但是XACML等基于属性的访问控制可以很好地解决这些问题，所以本系统决定采用将XACML和ABE相结合起来进行访问控制系统设计。

2.2 XACML

XACML 是在 2003 年 2 月由 OASIS （organization for the advancement of structured information standards）制定的一种基于 XML (extensible markuplanguage)用于决定请求/响应的通用访问控制策略描述语言和执行授权策略的框架[24]。在基于Web的分布式应用环境的安全策略广泛使用XACML语言进行表示。

2.2.1 XACML访问框架与流程

XACML访问控制主要框架和流程如图2.3所示[24]，现简单介绍如下：

策略执行点(Policy Enforcement Point，PEP)：PEP是在一个具体的应用环境下执行访问控制的实体。具体功能是将各种应用环境下不同访问请求转换为符合XACML规范的请求和根据PDP判决的请求结果执行相应的判决动作等。

策略决策点(Policy Decision Point，PDP)：PDP是访问控制系统中授权决策的实体。它通过PAP和PIP对访问请求做出访决策结果。

策略管理点(Policy Administration Point，PAP)：PAP是在访问控制系统中生成和维护数据访问策略的实体。

策略信息点(Policy Information Point，PIP): POP是可以获取主体、客体和环境的属性信息的实体，并为其他实体提供属性信息

上下文处理器组成:主要功能是完成相关匹配、转换工作。

图2.3 XACML访问控制框架

2.2.2 XACML策略语言模型

XACML策略语言模型如图2.4所示。它使用XML标签标示各个子元素模块，包括策略名、规则名、属性名、规则组合名称等，种类多。属性具体包括主体、客体、行为和环境四种，因此XACML语言表达能力强，能很好地描述现实场景对访问控制的要求

XACML策略语法包括：一组规则（Rules）、规则组合算法的标识符（Policy Combing Algorithm）、一组义务（Obligations）和一个目标（Target）[24]。目标（Target）作用是确定该策略是否与请求有关。规则（Rule）由目标、条件和结果构成，一个策略可以有多条规则。条件是关于匹配属性的描述；结果是请求评估正确后得到的预期结果；目标与策略中的目标作用类似。3.规则组合算法的作用就是解决多条规则间出现的结果冲突。义务（Obligations）是PEP在得到请求结果后，必须与授权实施同时执行的一组工作。

图2.4 XACML策略语言模型

2.2.3 XACML开源框架调研**分析与选型**

在选定了XACML作为访问控制主体框架后，我们寻找并简单实现了几种不同支持XACML访问控制开源框架，主要有Sun\_xacml、facpl、enterprise\_xacml（Google implement）、ndg\_xacml、xEngine和Wsoa\_banana几种，现简单介绍如下：

FACPL：它作为一种形式化定义用来具体说明、分析和执行的基于属性访问控制语言，但它并没有按照XACML标准进行设计。优势是语法简单易懂，容易入手，采用java编写，作为Eclipse的插件形式实现，容易实现。但是实用范围小，帮助资料不多。在测试过程中较难与其他框架整合，最后放弃。

Enterprise\_xacml:作为最新的的xacml开源框架，由Google公司开发，支持XACML2.0标准。它克服了很多Sun\_xacml开源框架的不足，增加了很多新的机制，如缓冲机制，改进了策略搜寻算法等，提升了框架效率，具有很多优势。但是由于开发较新，在简单测试其功能后，发现学习资料较少，学习成本高，考虑到时间因素最后放弃采用这一框架，但今后应该时刻关注其发展。

Ndg\_xacml：支持XACML2.0标准，但是由于采用Python编写实现，考虑到系统主要采用java编写，在简单测试了其功能后放弃了这一框架。

Wsoa\_banana：相比于其他框架，它支持XACML3.0标准，学习资料较多，而且具有应用到实际场景的具体商业例子，本身也是基于Sun\_xacml实现，学习成本较低。但是在前期测试过程中，没有成功实现其基本功能，最后放弃。

Sun\_xacml:作为开发最早、应用最广、学习资料最多的XACML实现框架，本身是基于XACML2.0标准实现，较为成熟。很多XACML开源实现样例和资料都以其为实现，学习成本低。在测试过程中，发现其虽然不能支持XACML全部特性，交流较低，扩张机制不足，但综合考虑，最终采用其作为系统实现框架。

XACML作为一种策略描述语言和授权框架，为访问控制提供了统一的语法，较大地提升了Web环境下不同结构组织之间协同访问工作的效率。其优点有适应多种应用环境、支持广泛的数据类型、具有规则组合算法、策略表达描述能力强等。总之可以用来描述各种复杂访问场景，满足对云环境下的细粒度的访问和安全需求。但是同时也具有表达复杂、策略必须可信、效率不高等缺点。XACML作为一种主流的访问控制模型框架，已经成功应用到了很多商业环境，取得了很好的效果。

总之，XACML 作为一种功能强大的标准访问控制描述语言，因其通用性好、可扩展性强大而被广泛关注和使用。但是其缺点也很明显，语法复杂，较难学习，对委托授权支持不够，性能不高，与现有引擎集成较为困难等。

考虑XACML是基于属性的访问控制模型,可以实现对资源细粒度动态可扩展访问控制，能够根据在云计算安全系统环境中主、客体和环境属性灵活动态地依据相关策略评估访问请求,并做出授权决定，因此比较适用于云环境。但是XACML只是负责对数据进行一层访问控制，保存在云服务器的数据的安全性不能得到保障。一种的解决方案是对其中的数据进行加密，云服务器只保存经过加密后的数据，这样就保证了用户敏感数据的的机密性。而如今基于属性的加密是云环境下访问控制的热点之一，其基于属性的特点决定了它可以与XACML有效结合，通过共用属性集的性质或者策略相互转换的方法，即可以提高系统效率，又保证了系统安全性，同时访问控制有着细粒度动态等特性。后文将详细介绍有关基于属性的访问控制相关知识。

2.3基于属性的加解密(ABE)

基于属性的加解密是如今云计算环境下访问控制的研究热点，自从2005年首次被提出以来很多学者都在其基础上进行了很多的改进，如今主要研究方向有CPABE、KPABE、分层ABE、多授权机构ABE。考虑到云计算对于访问控制的要求、时间和本人的能力，决定采用基本的CPABE来进行对数据的访问控制。

2.3.1属性加密基础知识

1.双线性对

定义1（双线性对）

定义G1，G2两个 p阶循环群，p是随机素数，设g是一个p阶群G1的生成元。定义映射e：G1G2G2是双线性对[25]在满足下列条件：

(1)双线性：对所有u，vG1，a，b，则



(2)非退化性：;

(3)可计算性：定义g1，g2G1，则存在一个有效的算法可以计算。 

定义2（双线性Diffie-Hellman假设BDH）

该假设定义攻击者可以在中随机选择三个参数a,b,c，则在多项式时间内不可能有不可忽视的优势区分元组(A=，，B=，,C=，，Z=)和（A=，，B=，,C=，Z=）。

定义3（公钥加密方案）

一般情况下，公钥加密方案主要是由以下四部分组成：

●系统初始化过程。输出公开系统参数PK、系统主密钥MK。

● 密钥抽取过程。输入公开系统参数、 系统主密钥，输出私钥SK。

● 加密过程。输入公开参数、 明文M，输出密文CT。

● 解密过程。输入私钥、 密文，输出明文M。

定义4（门限）

门限被定义一个逻辑运算单位。输入一般可以接受 N个，并且具有确定的阀值K，规定 K≤N，输入要求只能有{0，1}两个逻辑状态， 输出规定只有一个状态。当输入状态为 1 的个数大于或者等于阀值 K 时，输出状态为 1，否则为 0。示例如图 2.5，图 2.6，图 2.7。

 图2.5普通门限 图2.6或门 图2.7与门

定义5（访问树）

访问树表示一个对数据访问结构[26]。访问树中的每一个非叶节点都是一个门限，这个门限由其子节点和门限值构成；叶子节点都则与一个具体的属性想对应，其门限值为1。访问控制树实现中一般都定义了几个函数：parent(x)函数表示节点x的父节点；attr(x)函数表示节点x关联的属性。访问树中所有的子节点编号用函数index(x)表示，从1到N进行编号，树中节点的编号都是唯一确定的,示例如图 2.8[27]。



图2.8 访问树结构

定义6（秘密共享方案）

秘密共享方案是关于(*t*,*n*)并以拉格朗日插值定理为基础由Shamir构造。其基本原理如下：假设n个用户要共享某个秘密sGF(q)(q 为素数, 且q>n), 那么这个秘密的发送者需要随机构造多项式P(x)（它是GF(q)上*t*−1次的多项式，而且要求*P*(0)=s）。然后, 秘密发送者发送第i个接收者(i,P(i))。这样，恢复期共享秘密s的唯一条件是必须只有有任意大于等于t个接收者将他们的密钥集合起来。这里假设访问结构δ是一个集合，其包含t个接收者的秘钥，设任意t个参与者拥有的子秘密记为P(x1),P(x2)…P(),定义



那么t-1次的多项式P（x）可描述为



显然有



2.3.2密文策略的属性加密方案

基于密文策略的属性基密方案(CPABE)的一般原理是将用户私钥与其属性相关联，数据密文与数据拥有者(DO)与制定访问策略及对应访问结构树相关联。用户能够解密密文的唯一条件是其私钥（拥有属性）满足对应访问结构树。 因此CPABE较KPABE更符合现实条件和云计算环境，也应用到了很多真实解决方案中，本系统也用此作为数据加密方案，因此将详细介绍此方案的构建过程[28]。

构建方案：CPABE构建包含四个基本过程：系统初始化算法（Setup），加密算法(Encrypt)，秘钥生成算法(KeyGeneration)，解密算法(Decrypt)，具体描述如下[27]

⚫系统初始化 Setup()算法。输入一个系统默认的安全参数；输出PK公共参数和MK主密钥。  
⚫加密算法 Encrypt（PK,M,A）算法。输入参数包括公共参数PK（上一步生成）、明文M，以及根据数据拥有者所指定策略A；输出数据密文CT。其中访问策略（结构）与密文CT相结合。  
⚫密钥生成 KeyGeneration（MK，S）算法。输入参数有主密钥 MK 和经过可信第三方认证的用户属性S；输出该用户私钥SK，一般情况下将其存储到用户。  
⚫解密算法 Decrypt（PK，CT，SK）算法。输入参数有公共参数PK，密文 CT。访问策略A与密文CT相关，用户通过第三方可信机构得到自身属性集S对应的秘钥SK，如果属性集合S与策略A相匹配则得到明文M。

2.3.3 CPABE访问控制系统架构



图2.9 访问控制系统架构

如图2.9所示为一般基于属性的访问控制系统示意图，数据拥有者Owner从AA（权威中心）申请公钥PK,根据策略m将文件m进行加密得到密文CT并保存到云端Server，云用户Users根据自己属性向AA申请私钥SK并从Server得到密文，如果拥有属性满足对应访问策略则得到明文m。

2.4 OpenStack云平台架构

OpenStack是一个开源的云计算管理搭建平台。由NASA和Rackspace共同研并且由Apache授权许可。它是云计算中IaaS(基础设施即服务)的组件，实验者可以通过它建立云计算服务器环境并为用户提供云计算服务。OpenStack目前已经受到很多研究者和企业的广泛应用。

本文所设计的访问控制系统最后部署并在其上进行测试，它主要功能是是由其提供云计算环境并为开发提供相关服务，如用户认证等。OpenStack的主要优势是构建过程简单，可扩展性强。其核心组件有Nova和Swift，分布用来进行云计算服务器配置、相关计算和用于分布式存储[29]，典型框架图2.10。Neutron组件主要用来管理平台网络，Keystone组件则是管理用户目录，Cinder组件作用是管理计算实例所使用的块级存储模块，Glance作用是为VM镜像提供支持。

图2.10 OpenStack框架

2.5 web开发技术介绍

由于本系统是基于Web端开发C/S架构，所以在系统构建过程中，学习研究应用了Web开发需要使用到的技术，主要有JSP、Servlet、JS(Jquery)，HTML/CSS等，现简单介绍如下：

(1)JSP

JSP（全称Java Server Pages）由Sun太阳公司首先提出，许多公司和开发者共同创建的，它的作用是使开发者能够响应Web客户端请求，动态生成HTML、XML或其他格式文档的Web网页的技术标准，其本质是一个Servlet。本系统中使用JSP技术创建相关交互页面，实现简单的前后端交互。。

(2)Servlet

Servlet是sun公司开发，是一门用java开发的动态web资源的网页服务端技术。为了实现交互式浏览、修改数据，动态生成Web内容。它为Web应用提供了基于组件和平台的方式。

本访问控制系统使用Servlet来处理客户端前端页面与云服务器后台的数据处理和相互，并动态生成相关的web页面。

(3)JS(JQuery)

JavaScript(JS)是一种网络脚本语言，一般用来改进设计、验证表单、创建cookies、实现简单页面逻辑功能等。它的优势是动态、弱类型、解释性语言，功能强非常大。

JQuery是一个轻量级JavaScript库，它的作用是极大简化JS编程，提高前端开发者工作效率。Jquery优势是可以很方便的处理html事件、实现动画效果，提供AJAX交互。

本系统使用JavaScript与Jquery实现系统前端web页面的表单验证、动画效果、数据传输等功能。

(4)HTMl(CSS)

HTML/CSS作为Web前端开发的基础，实现了浏览器网页的一般组件和样式设计，是如今万维网网页设计的基础，具有简单、可扩展、平台无关等特点。

本系统使用这两种技术实现网页基本组件和样式。

(5)Tomcat

Tomcat是一个免费开源的轻量级Web应用服务器，是Apache等公司及开发者共同开发。Tomcat可以作为Servlet和JSP运行的容器，优势是技术先进、性能稳定。

本系统使用其作为开发和调试访问控制子系统的web服务器，初始的版本也部署其上，用于测试。

2.6本章小结

本章主要介绍了访问控制选型和本文使用的XACML访问控制框架，基于属性的加密（ABE）相关知识和本文所要使用的CPABE机制，本系统所要部署的OpenStack云平台和在WEB开发中所使用到的技术，为后面章节所介绍的云环境下基于属性的访问控制系统设计和实现提供了理论基础与技术支持。

第三章 基于属性的访问控制系统设计方案

本系统将基于属性的XACML框架和CPABE属性加密机制相结合合起来，它们共用主客体、环境属性集，或者共用访问策略。这样，在保证云计算环境下用户数据和隐私的安全情况下，还为访问控制系统提供了细粒度访问、高效、动态、可扩展的特性。

系统设计的总体思路是在云环境下（开发测试环境是Web），利用XACML实现对用户存储在云端数据的访问控制；对用户某些敏感或者机密数据，利用CPABE算法对其进行进行加密操作，在云端服务器只保存经过加密后的密文数据，这样就保证了用户保存在云端数据的安全性。由于当前CPABE实现算法效率较低，其中较为可行的方法是将其与对称加密AES算法相结合，利用AES对数据加密，利用CPABE加密AES加密的秘钥来改善整个系统的效率[30]。

3.1系统总体设计思路

一般情况云环境下访问控制系统总体设计方案如图3-1所示，由认证子系统、授权子系统、审计子系统组成。其中认证可以采用用户令牌方式；审计则为系统提供了违规操作警告和日志记录功能；授权采用本系统设计的基于属性的云访问控制，由XACML授权机制保证云平台中对安全性要求不高的非敏感数据，CPABE与XACML授权机制相结合共同保证对安全有较高要的机密数据[31]。

由于本人能力和精力有限，设计实现的时间不长，本毕设所设计的访问控制系统只是实现了总体方案中的XACML授权部分和CPABE加密机制，也就是说是实现了基于属性的云存储访问控制子系统。

图3.1 云环境下访问控制总体方案

考虑到对于云服务提供商并不能完全信任，即存储在云端的明文数据可能受到攻击，因此为了提高云数据的机密性，特别是存储在其中的敏感数据，本系统采用了ABE加密措施使云端保存的是经过加密的数据。这样即使云端务器被攻破或者非法获取，攻击者也不能获得有效数据。但是考虑到CPABE目前算法效率较低，系统中使用可能导致整体性能和效率下降，本系统采用了以下方法来缓解系统性能的下降：

(1)CPABE加密与XACML框架使用存储在数据库中提前构建好属性集。即在用户注册和上传数据时将主体属性和客体属性保存在数据库中，CPABE与XACML使用该通用属性集，从而部分降低了属性的管理开销，从而提高了系统性能

(2)对于体量较大的数据，本系统在可以先采用AES对称加密算法来加密数据，再使用CPABE加密AES秘钥，这样就提高了整个系统加密加密的效率。

(3)文件的加解密操作都在云服务器端执行，降低客户端系统消耗，保证用户数据加密效率。

3.2系统关键方案设计

3.2.1构建通用属性集

由于XACML设计和CPABE机制都是基于属性的AC解决方案，如何高效管理存储相关属性集就是提升AC系统效率，改性AC系统设计关键点。

本AC系统使用数据库(MySQL)来作为属性权威(AA,attribute authority)来创建和管理主体、客体和环境等XACML和CPABE用到的属性[32]，他们都从AA中得到所需信息，这样双方不用单独维护管理属性集，降低了成本，减少了属性异构产生的可能性，提升了系统效率，。其中，主体属性有id, 姓名、性别、年龄、学校、专业、职务、password等；客体属性主要有客体名称、所属主题、Owner、数据大小、创建日期等；行为属性包括打开，更新，删除；环境属性包括域名、IP地址、日期、网络状况和行为发生时间等。

在用户进行访问策略构建的过程中，数据拥有者从数据库中提取主、客、环境属性字段并指定对应内容。生成资源策略文件后需要将其存储到策略仓库文件中，如图3.2所示，方便请求匹配查找。

在用户向某一数据资源提出访问请求，云服务器对这一请求进行访问决策这个过程XACML需要从属性仓库中提取相关属性生成请求XML文件，接着PDP从属性仓库中获取对应属性以进行决策。



图3.2利用通用属性集构建访问控制策略

3.2.2 DO申请数据托管服务与CSP加密数据服务

数据拥有者在数据上传后需要制定XACML访问控制策略和决定是否进行加密。云计算服务器在需要加密的情况下使用如图所示的加解密模块对数据进行加密操作，或者将其直接存储在云端。以下为具体过程：

(1)用户在页面指定CPABE加密策略T（门限结构）。

(2)云服务器根据资源访问策略T，数据M和之前得到的供参数PK，调用CPABE加密算法Encrypt(PK,M,T)加密文件F得到。具体原理如下：定义函数ind(x)是节点x的索引编号，par(x)是节点x的父节点，att(y)是叶子节点y对应的属性。假设访问树中节点x门限值为，则随机多项式是节点x生成的(-1)次表达式，代表了节点x的秘密。随机从中选择数s，另（R为根节点）。其他节点x，让



另Y为叶子节点的集合，则密文如下表示：



(3)将生成的密文存入到云服务器数据仓库中（可能会区分是否是加密数据仓库）。

3.2.3 CSP构建访问控制策略

本系统在访问控制策略的创建时，XACML和CPABE的策略是分开、由数据上传者自己制定的，需要DO了解XACML和CPABE策略的指定方式。系统将XACML策略文件存储到策略文件仓库(PAP)中,方便PDP调用，搜索请求对应策略，安全管理策略文件。

CPABE访问控制策略是门限结构，支持如AND，OR，n of m的形式。如：(CUMT AND STUDENT) OR (2 OF (CUMT, TEACHER, MONITOR))。

XACML访问控制策略的最后表现形式是以XML文件存在的，详细介绍见第一章XACML相关知识，构造过程间第四章XACML访问控制策略生成。

3.2.4 User申请访问数据

当用户申请对保存在云服务中的数据资源进行不同访问操作请求时，需要进行以下处理：

(1)验证云用户的身份是否合法。

(2)通过属性仓库中提取的主客体属性和用户当前环境的属性信息生成XACML请求文件。

(3)XACML访问控制框架对请求文件和策略仓库中的策略进行匹配决策，若匹配成功则接受访问否则拒绝该请求。

(4)在XACML结果为允许的情况下，若请求访问的文件为加密文件，利用用户的属性集A生成的私钥SK（其中 为随机参数，为属性）



执行解密算法Decrypt(CT,SK)得到文件数据明文，解密过程如下：定义DecryptNode(CT,SK,x)为递归函数，另i=att(y),计算每个叶子节点x：

当时，DecryptNode(CT,SK,x)返回空；

当时，



DecryptNode(CT,SK,x) =

若节点z为非叶子节点，则利用个叶节点的DecryptNode(CT,SK,x)作为拉格朗日插值定力的插值点得到。

访问控制树T的根节点为R，另



=

则明文为



3.3系统整体设计流程图

本毕设设计的云环境下基于属性的访问控制子系统系统流程如图3.3所示，具体流程如下：

1. 云用户进入不同服务页面后首先要进行身份认证，在认证成功的情况下才能访问不同页面（用户登录后首先进入欢迎页面）；若用户没有注册，可以进入用户注册页面进行注册，在这里会填入用户的属性，用来为XACML和CPABE提供其对应的属性集；在用户登录时还会为其生成属性对应的私钥并存放在本地。

2. 用户在进入到欢迎界面后，可以进行对数据的操作，如：上传数据、浏览已经上传到云服务器存储的数据、删除数据、修改数据等。也可以对自身用户信息，如：属性等进行修改。由于本系统关注点在于对数据进行访问控制，所以只对访问控制相关内容进行研究。

3. 当用户选择上传数据操作时，首先选择本地对应文件进行上传，后台自动将文件（客体）的属性，如:上传文件文件名、文件数据大小、文件类型和所有者等信息写入数据库中；下一步也是最重要的和关键的一步就是生成该文件对应的XACML格式访问策略文件，其中包括策略ID、目标、规则等信息；接下来用户选择是否对数据进行加密操作，不需加密直接返回欢迎页面，文件上传成功，如果需要对上传数据进行加密操作，则进入CPABE加密策略输入界面，当数据上传者指定了对应策略后服务器端将数据利用CPABE加密算法进行加密并将加密文件存储到云端。

4. 当用户选择浏览数据页面时，进入数据内容页面，显示云端已经存储的数据，云服务用户选择感兴趣的文件并选择对应的操作，并从数据库中提取主客体信息和环境信息生成XACML请求文件；在PDP组件中输入用户提交的请求xml和策略库文件，对用户请求进行判断，若请求结果为deny表示用户对该数据没有对应权限，返回数据操作页面；在请求结果为允许的条件下，判断数据是否为加密数据，若为非加密数据，直接返回结果；若为加密数据，则使用用户对应的私钥对加密数据进行解密，若成功返回提示信息，否则返回对应错误信息。

图3.3 系统基本流程图

3.4本章小结

本章主要介绍了本毕设所设计的面向云环境下基于属性访问控制子系统的方案设计。包括了本系统的主要模块功能、数学构造、设计思路和程序流程。这一章为后续章节系统的具体构造提供了详细的开发思路。

第四章 基于属性的访问控制系统实现方案

本系统是在Windows系统下使用Eclipse进行开发的JavaWeb开发。XACML框架利用SUN公式的开源库sun\_xacml[33]实现，CPABE利用JPBC为其提供底层实现，参考Linux下的CPABE-Toolkit[34]实现，最后部署在OpenStack云服务器端

4.1系统部署

4.1.1系统软硬件环境

操作系统：Win7 6.1（64位）

CPU：Intel® Core™ i5-3317U CPU @ 1.70GHz

内存：4.00GB DDR3 1600MHz

硬盘：日立HTS45050A7E380(500GB / 5400转/分)

开发环境：Eclipse Hbuilder Sublime

第三方库：SUN\_XACML JPBC cpabe-0.11 json-lib mysql-connector等

4.1.2系统部署实体机拓扑图



图4.1 系统部署实体机拓扑图

如图4.1所示，设计的基于属性访问控制系统云服务器实体机拓扑图主要由三块组成，分别有存储节点、代理服务器和认证服务器。其中认证服务器作用是验证系统访问者即用户身份；代理服务器作用是实现基于XACML的访问控制和；存储节点作用是物理上存储用户上传到云端的各种数据资料。

4.2 OpenStack平台设计与实现

由于系统开发是基于本机上的web测试，测试后集成到实验室OpenStack云平台上实现对数据的访问控制功能，其搭建和介绍并不是本文的关注点，下面简单介绍其swift存储服务和keystone认证服务的交互的访问过程。



图4.2 OpenStack平台设计图

如图4.2所示，在OpenStack平台流程中用户端首先向认证服务器keystone发送登录请求，若通过其获得Token；接着用户端使用Token将请求发送到swift-proxy；swift-proxy提取其中的Token并向keystone验证用户身份；若验证通过，swift-proxy获得访问者身份信息，用户登录成功，否则返回错误信息；接着根据验证信息和所访问的数据的访问策略决定是否授权访问，成功则将请求的资源返回给用户端，失败则返回错误信息。

4.3系统实现功能

本文所描述的云环境下基于属性的访问控制子系统主要有四个子系统主要组成：

一、数据库相关子系统。该系统主要作用是存储云用户上传的相关数据；存储主体、客体相关的属性；提供与后台交互的接口。

二、XACML访问控制子系统。该系统简单实现了XACML中的PDP、PAP、PEP、上下文环境等组件，提供了制定策略、请求转换、策略请求匹配等基本功能。

三、CPABE加解密功能子系统。该系统主要实现了数据方面基于属性的加解密，包括公共参数、主私钥、用户秘钥的生成；加密、解密算法的实现；加密数据、相关公私钥的存储等功能。

四、Web端的前端子系统。该系统主要是前台与用户交互的实现，利用JSP、HTML/CSS、Jquery等前端语言为后台提供数据、显示结果等。

如图4.3所示为访问控制系统基本方案设计图，由四部分组成。分别是底层的数据存储层，上层的前端Web显示层和后端java逻辑层（XACML授权框架和CPABE加解密）。以下具体介绍相关子模块的设计实现。



图4.3 访问控制系统基本方案框架图

4.4访问控制系统主要模块设计与实现

为了实现对保存在云服务器中数据资源的细粒度安全访问控制功能，本系统主要分为以下几个关键模块：数据库设计与实现模块、用户登录与注册模块、文件上传模块、文件浏览与XACML请求生成模块、访问控制策略生成模块、PDP模块、CPABE加解密模块等，下面进行详细说明，系统欢迎/主页面如图4.4。

图4.4 欢迎页面

4.4.1数据库设计与实现

测试系统的用于保存主客体属性信息的数据库设计模块由Mysql数据库实现，利用JDBC API编程实现后台对数据库表相关交互操作，利用JavaBean-DAO设计模式实现java和数据库间的交互。以下为操作步骤：

1）利用MySQL Workbench建立AccessControlXACML数据库，定义用户名、密码、地址端口号。在其中建立resource\_tbl表与user\_tbl表，分别用于存储客体、主体属性信息，具体字段信息如下图4.5.

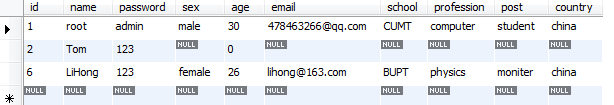


图4.5 user\_tbl表

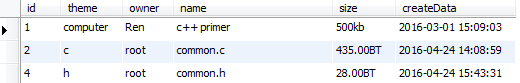


图4.5 resource\_tbl表

2）利用JDBC API在Eclipse IDE中编写配置数据库配置信息dbconfig.properties，指定连接数据库的URL、用户名、密码等信息；新建数据库连接类，读取之前新建的配置信息，返回一个数据库连接，方便之后数据库操作。

3）实现相关Entity类，如User、Resource，与数据库中的表相对应，提供set()和get()方法；实现实体类对应的DAO接口，它的功能是定义所有用户对数据库的操作，如添加记录、删除记录、修改记录等；定义DAO实现类，实现以上接口的所有方法，并连接数据库。

4）除属性信息保存在数据库中，其他在云端存储的数据以文件的形式存储在OpenStack云平台中。

4.4.2用户登录与注册模块设计与实现

用户登录与注册模块保证了系统用户的合法性，实现了用户属性信息的注册、修改等功能。利用JSP、Servlet实现相关逻辑部分。利用Servlet中的Filter过滤技术保证了在前后端数据传输过程中编码的一致性。前端页面利用Bootstrap和Jquery设计实现，同时保证了web页面的美观性和功能性，后台逻辑实现利用Servlet中doPost()方法进行处理。由于实现较为简单，这里只给出相关程序流程图，用户登录程序流程见图4.6。注册程序主要是利用Servlet获得前端页面输入的数据后直接写到对应数据库表中。



图4.6 用户登录示意图

4.4.3文件上传、浏览模块设计与实现

该模块的主要功能是实现用户将数据上传到云服务器端，并自动将用户上传数据的属性信息添加到数据库对应的resource\_tbl表中；将已经上传的数据信息提供给云用户，并为用户提供相关操作。

对于文件上传流程，一般情况下是客户端浏览器将文件转换成流的形式传输并提交给服务器。本文利用Apache的开源工具common-fileupload实现文件上传功能，具体用到了common-fileupload和两个common-io包。在文件上传的实现中，要注意一下问题：1）为保证云环境中相关服务器的安全，用户上传的文件应该放在其他人不能访问的目录下，在测试端Eclipse中如WEB-INF的目录就可以。2）为防止文件覆盖的发生，保证同一个文件可以上传到服务器多次，每个用户上传文件在服务器中都应该有一个唯一的文件标示名。3）为防止目录下出现的太多文件，利用了hash算法打散存储。4）限制上传文件的类型、最大值等。程序流程图4.7如下.

图4.7 文件上传程序流程图

对于文件浏览操作界面，由于本系统只是涉及到对数据的访问权限控制，并没有实现具体对数据的操作。在页面中显示了所有上传到云服务端的各个文件，并为用户提供了对其可以进行的三个操作：阅读、更新和删除。其Web页面实现利用了bootstrap库进行开发，后端利用Servlet后台获取文件列表信息，具体原理是利用递归获取上传文件目录下的所有文件名，并将其保存在map结构中。最后利用在JSP页面嵌入c标签来显示文件目录信息，遍历指定目录下所有文件的流程图见图4.8所示。服务器文件浏览部分的核心代码如附录一。



图4.8 得到文件目录程序流程图

4.4.4访问策略生成、请求生成模块设计与实现

本模块作为基于属性访问控制系统的核心功能之一，是对数据进行访问控制所需的基本组件，因此是XACML框架的核心组成部分（PDP和PAP）。在云用户将文件上传到云服务器后，DO需要为上传的文件制定XACML访问控制策略，规定该文件对于数据访问者的权限。前端页面利用JQuery和JSP实现。考虑到前后台有大量的数据需要交互，利用AJAX将前端数据打包成JSON格式，后台进行相关解析来实现交互工作,策略编辑设计的整体思路见图4.9。



图4.9 策略编辑设计思路

对于本文所设计的基于属性的访问控制授权系统，XACML访问控制策略是其核心组成部分，是权限控制的关键要素。考虑到其功能强大，支持类型多，较为复杂，为了便于用户理解和使用，本系统设计的访问控制策略生成页面没有支持XACML策略的全部组件生成。其中Target组件中<Resource>的默认由之前上传文件的信息自动生成，<Subject>和<Action>信息则被忽略；本系统不支持<Obligation>、<Advice>、<PolicySet>组件。其他Policy核心组件，如策略ID、策略组合算法、策略描述、规则在本系统中都支持。另外，在<Rule>中，不支持<advice>。本策略构造可添加多个<Rule>,在<Rule>中支持<Subject>，<Resource>，<Action>，<Environment>相关属性指定。

在规则生成时，前端页面目前支持多种变化类型。在一条规则生成时，可以支持非常细粒度的策略生成。规则中主、客体属性由数据库中对应表字段名指定，动态生成。比如在对用户属性进行限定时，需要具体指定is/isn’t、具体属性(如姓名、年龄、学校等)、匹配方式(如equal、less than and greater than等)。

策略编辑页面利用Jquery库实现相关动画动作和数据AJAX封装，CSS实现页面美化。后台策略文件生成利用了Sun\_xacml开源库策略生成的相关API。首先解析从前台传来的JSON数据，利用迭代器递归方式从中提取生成策略所需的信息，如策略名、规则信息和主客体环境属性等；然后通过初始化函数生成不同的策略组件，并最终将生成的策略XML文件安全存储到云端对应的服务器中。核心代码见附录二，生成的策略xml文件见附录五。

4.4.5 PDP、PEP模块设计与实现

这两个模块设计所使用的是开源框架Sun\_xacml提供的API。PDP模块主要实现的功能有：生成XACML核心组件PDP(包括配置策略文件位置；添加主体、客体、环境属性；自定义函数等功能)；关联PDP配置文件；针对策略评估请求并生成对应response.xml结果。PDP创建和请求评估过程如下：

(1)定位策略文件所在位置、请求文件参数并配置FilePolicyModule。

(2)创建PolicyFinderModule并利用FilePolicyModule对象初始化。

(3)设置PDP策略发现程序。

(4)初始化之前自定义的PDP类，创建本次匹配所需要的PDP。其中利用sun\_xacml的FilePolicyModule类来对PDP构造函数中所需的PDPConfig进行配置。

(5)调用PDP中的evaluate方法，对来自之前经过PEP转换符合XACML标准的请求进行评估，最后返回授权结果并输出到对于文件。

XACML中PEP模块主要功能是根据访问者本身的属性信息、请求资源的属性信息、用户所要进行的操作和其他所需的环境信息创建符合XACML标准要求的请求文件。本系统使用较为简单的方式生成请求。通过用户选择需要操作的文件得到客体信息，从数据库中查找对应属性保存到List<>；通过Session中的注册的user信息得到主体信息，从数据库中查找对应属性保存到List<>；通过用户选择得到行为信息；通过系统运行环境得到环境信息。最后根据这些请求信息，利用sun\_xacml生成请求的API指定每个请求信息的url、AttributeId、DataType最后生成对应的request.xml，其程序流程图见图4-10，核心代码见附录三，生成的request.xml见图5.5。



图4.10 请求生成流程图

4.4.6 CPABE数据加解密模块设计与实现

CPABE作为本文所设计的基于属性访问控制系统的核心模块，保证了保存在运费钱中数据的安全性，是基于属性访问控制的核心功能。由于本系统利用java实现，不能利用c语言设计的cpabe toolkit。故借鉴其思想，利用JPBC加密库实现CPABE底层的双线性对；利用网上开源代码实现基本AES加密，再其上封装一层基于密文策略加密的实现，即CPABE。由于目前处于测试实验阶段，本系统设计的CPABE加密算法访问控制策略目前支持门限结构，因此策略描述能力较弱，但之前设计的XACML框架与其相结合后可以很好地弥补这一缺陷。

CPABE加解密过程中，数据拥有者需要对上传的数据制定访问控制策略，以对其进行权限控制。数据上传成功并指定XACML访问策略后用户需要指定该文件需要进行加密操作；在确认加密的情况下需要DO以文本框形式输入CPABE访问策略。

具体模块流程如下：

1．服务器创建时（测试系统默认为用户登录时），检测服务器中是否已经创建PKFile和MKFile。如果没有，执行CPABE初始化算法setup(),生成加解密过程中所需要的公钥PKMile和主私钥MKFile并保存在服务器中。具体实现过程是：检查公共参数和主公钥是否为空，然后通过JPBC API读取相关配置信息生成双线性对，最终生成PK与MK,

2.用户验证成功后，在信任中心检测是否存在该用户对应私钥。若不存在，调用CPABE用户秘钥生成算法KeyGen(attrs, PK, MK, SKFile)，从服务器数据库中提取用户对应属性，和保存在服务器端的PKFile、MKFile，生成用于用户解密的私钥并存储到AA（信任中心），这样就保证了私钥的安全性。

算法具体过程是首先检查用户属性集是否为空，MK、PK是否存在，然后调用SerializeUtils类的unserialize()方法实例化秘钥生成所需的PK和MK文件，最后通过JPBC生成的双线性对pair和对象Element与用户属性集来构造秘钥并最后以文件形式输出。

3.在DO加密文件时，根据DO指定的CPABE访问控制策略（即在上图中输入的策略）调用CPABE加密算法Enc(file, policy, PK, cipherFile)，加密用户上传到服务器的文件并最终只在云端保存数据密文。

算法具体过程如图4.11所示，主要包括先对输入信息进行合法性验证和检查相关文件是否存在，然后将字符串形式策略policy进行解析转换，最后调用底层AES实现好的相关函数进行加密操作。



图4.11 CPABE加密算法流程图

4.在用户通过验证后请求文件相关访问权限时，需要先进行XACML机制进行权限验证。如果请求结果为允许，并且请求数据为加密数据时，需要从AA申请云用户对应私钥，调用CPABE解密算法Dec(ciperFile, SK, FK)对数据文件进行解密。如果云用户属性满足对应数据访问策略，则请求结果为允许，返回数据原文。算法流程和核心代码见附录四。

4.5本章总结

本章详细介绍了本访问控制系统的部署环境、平台运行过程、系统主要功能和各个模块的具体实现方法，如XACML策略编辑和CPABE加解密过程，并展示了系统主要界面框架，为读者展示了本系统的设计实现过程。

第五章 系统功能测试与安全性分析

在初步完成系统后，对其进行了进一步的功能测试，保证系统的完整性和可靠性。测试环境是开发使用的ASUS K46C笔记本，性能：Intel® Core™ i5-3317U CPU 1.70GHz，4.00GB 1600MHz内存；操作系统Win7 6.1（64位）。服务器为Tomcat 7.0.55。IDE为Eclipse Luna Service Release 1 (4.4.1)，览器为Chrome 49.0.2623.87。

1.系统功能测试

在访问控制系统搭建成功后，我们进行了简单的系统功能测试以验证对数据的细粒度访问控制和加密功能，现介绍步骤如下：

1.首先进入登录界面，见图5.1，如用户未注册需要先进行用户注册。填写信息见图5.2，填写完成后生成对应数据库表见图5.3。

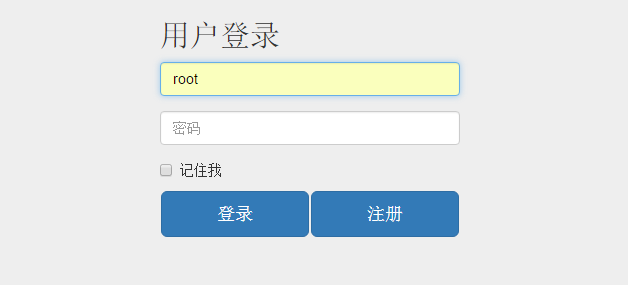


图5.1 用户登录界面

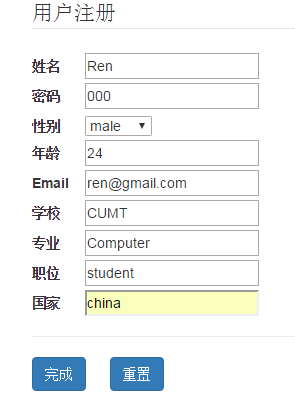


图5.2 用户注册信息

C:\Users\renwei\AppData\Local\Temp\ScreenClip.png

图5.3 数据库中用户属性信息

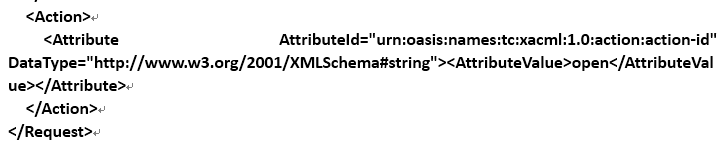
2.用Ren用户信息登录系统后，进行数据上传操作上传数据名为note.txt,文件上传成功后如下图5.4；并为其制定XACML访问控制策略，策略为“只有用户学校为CUMT才可以对其进行delete操作”，运行成功后的生成对应文件的授权策略文件见附录五（该测试先不进行ABE加密）。



图5.4 文件上传成功示意图

3.进行访问测试。首先用Ren用户登录，进行数据删除操作，由于Ren用户school属性为CUMT，可以进行该操作生成的请求文件request.xml如图5.5所示，操作结果如图5.6所示；用LiHong用户登录，由于其school属性为BUPT,不能对其文件进delete操作，操作结果如图5.7所示。

图5.5 请求文件



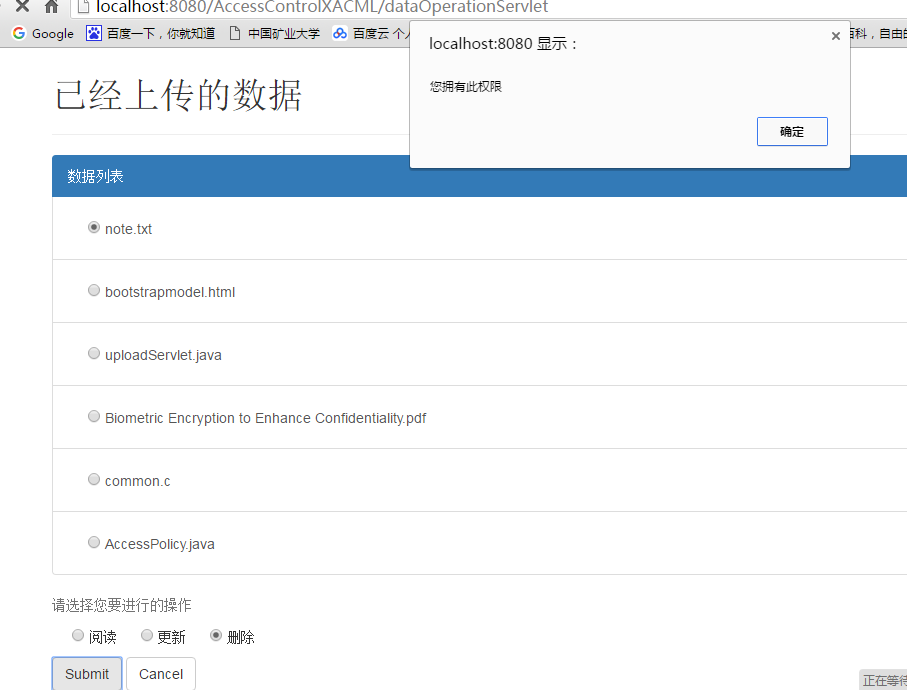


图5.6 Ren用户操作结果



图5.7 LiHong用户操作结果

4.进行数据加密操作，按以上步骤进行数据上传，文件名为picture.jpg，XACML访问策略为“country属性为China的用户都可以对其进行read操作”，这里不再进行赘述。在制定完XACML访问控制策略后，选择进行“数据加密操作”，在弹出的策略编辑器中（见图5.8）输入以下策略，生成的加密文件（后缀为.cpabe）如图5.9所示。

“(1 of (teacher, student, instructor)) AND (1 of (CUMT, BUPT))”

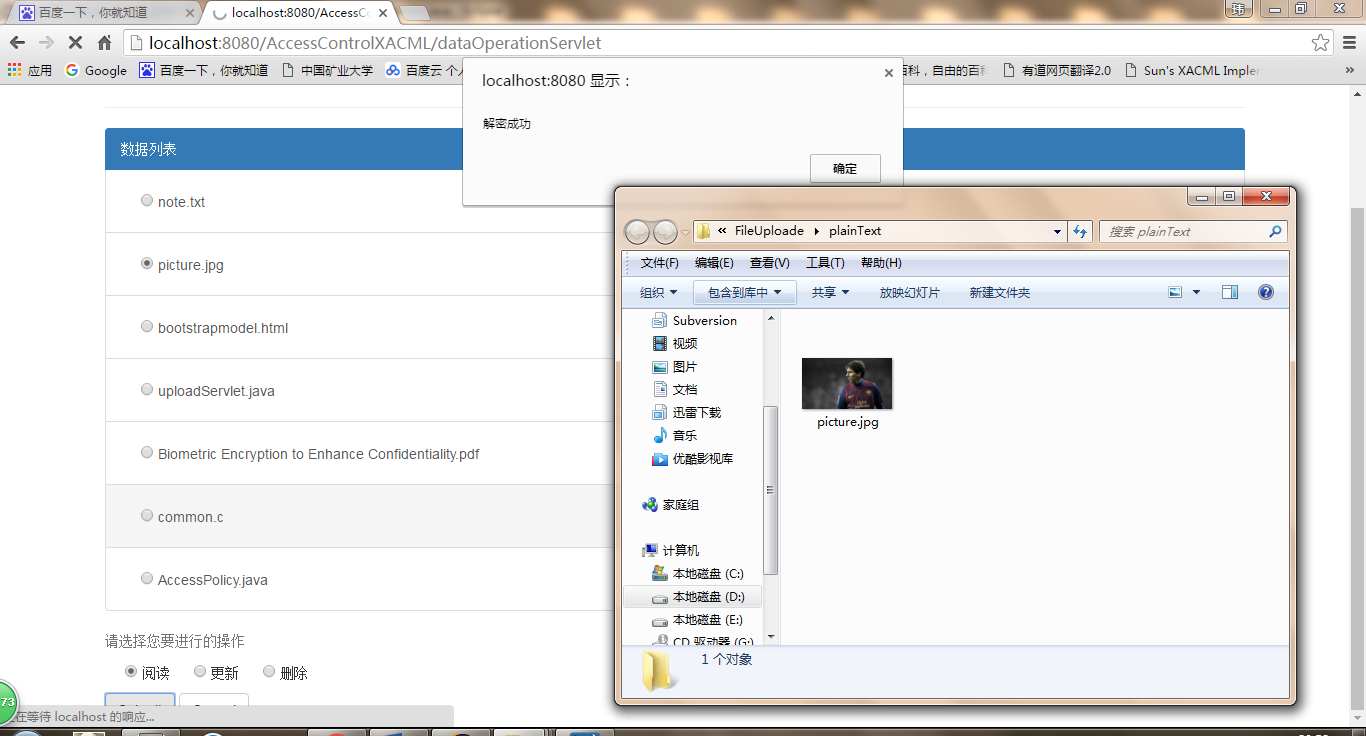


图5.8 CPAPE策略编辑页面



图5.9 加密文件

5.进行用户操作验证，由于本系统中所有用户country属性为China，因此通过XACML访问控制框架。下一步进行CPABE验证，Ren用户满足其策略，结果如图5.10，弹出解密后的文件；LiHong用户不满足其策略，结果如图5.11，不能进行解密操作。



5.10 Ren用户操作结果



图5.11 LiHong用户操作信息

2.系统安全性分析

(1)数据安全性分析

为保证存储在云服务器中数据的安全性，首先对于敏感数据，用户通过CPABE算法加密，对于CPABE加密后得到的密文采用XACML授权机制进行访问控制。由于CPAGE已经被Waters证明其安全性是基于DBDH困难假设之上的，从而可以证明密文数据的安全[35]，因此整个访问控制系统对应云端的数据的安全性和隐私是可以得到保证的。而XACML框架则保证了合法用户可以在授权范围内对数据进行细粒度动态访问。

(2)访问控制策略安全性分析

在本文所设计的方案中，所需的访问策略是由数据拥有者制定的。策略中相关的主体、客体和环境属性被安全存储在数据库中，非法用户不能获取这些信息，这就保证了访问控制策略的机密性。同时访问控制策略中细粒度的属性，如环境属性保证了整个系统的高效性和安全性。

(3)抗合谋攻击

假设云服务器不能完全信任，假设多个进行合谋攻击的用户属性机会满足对应数据访问控制结构树。是攻击者对应属性产生的私钥，其属性不满足对应策略就不能访问相关数据；就算攻击者进行合谋攻击，也不能恢复，对应解密的秘钥文件也无法得到。并且对于合谋攻击Goyal等人已经在文献中证明CPABE是安全的[36]，而本文是基于CPABE实现的，其抗合谋攻击的安全性也得到了保证。

第六章 总结与展望

6.1总结

云计算作为近年来学术界、产业界的新兴发展趋势之一，取得了极大的关注和飞速发展。但是在云计算发展过程中逐步凸显出来的问题，首当其冲的就是安全和隐私问题，已然成为其发展路程上的拦路虎，严重制约着它的进一步发展。考虑到云计算环境具有分布式、海量数据、扩展性强的特点，我们提出了一种适用于云环境的基于属性的访问控制系统方案，在保证安全性的同时，它支持细粒度访问，可扩展性和灵活性也得到了较大幅度的提高。本文的主要研究成果是成功实现XACML和CPABE相结合的基于属性的访问控制系统，并最终将测试版本部署在OpenStack平台上。

本系统通过共用属性集和访问策略转化来使XACML访问控制框架和CPABE加密机制相结合。使系统具有了XACML授权机制的实时、可扩展、细粒度访问、动态的优点，又通过CPABE属性加密算法保证了保存在云服务敏感数据的机密性。考虑到加密算法效率不高的特点，本系统将加解密实现移交在服务器端实现，减轻了云服务用户的负载负担。

在sun\_xacml、jpbc和cpabe-toolkit等技术的支持下开发了整个系统的框架流程，包括设计原理、软硬件平台、部署拓扑结构、重要子模块的详细具体设计等，在整个系统构建中设计了两个最重要的模块：策略生成和加解密子系统。最后通过C/S客户端进行对系统的详细功能测试，并深入分析了测试结果，成功取得了预期的效果。

该系统有如下功能/特点：

(1)支持用户新用户注册，主体属性动态增加。

(2)XACML和CPABE共用主、客体属性，提高系统效率，同时集成了XACML授权机制和CPABE的优点。

(3)在web端设计实现了基于属性的访问控制系统.其中XACML访问控制策略制定（PAP）的网页版实现，简化了XACML中较为复杂的机制，便于用户理解，同时保留了其特有的强大语法表达能力。

(4)在基于属性的加密设计中，设计了相关UI页面，便于用户理解。

6.2展望

由于时间原因和本人能力有限，本文实现的云环境下基于属性的访问控制子系统只是一个初步的设计方案，还存在着不少问题和待优化的地方。其中最主要的问题是如何更好地将XACML访问控制机制和CPABE加密机制更加高效结合起来，目前只是共用主、客体属性集，在访问控制策略方面还只是各自输入，很不方便；另外的问题是XACML框架本身，由于其语法的复杂和开源框架的局限性，还存在着诸如系统效率不高、无法适用委托授权等问题；还有在系统优化方面，缺乏缓冲机制，导致策略评价效率低，策略搜寻算法低效等问题。在云平台应用方面，目前只是简单部署在OpenStack云服务器上，没有充分利用其自身的访问控制和身份验证子模块。今天的研究设计任务是如何更高效的实现用户权限撤销，数据策略更新等问题。

云环境下基于属性的访问控制是目前云安全研究热点之一，它具有很多传统访问控制方案中所不具备的优势，虽然有一些问题存在，但是瑕不掩瑜，特别是其细粒度性、可扩展性和动态性，非常适合云环境下对访问控制的要求，特别是其中基于属性的加密ABE(Attribute based Encrypt)更是现阶段访问控制和数据加密领域的一个极具挑战性的研究课题，更是极具前景的发展方向。

参考文献

[1] RITTINGHOUSE J, RANSOME J. Cloud Computing: Implementation, Management, and Security [M]. CRC Press, Inc., 2009.

[2] CACHIN C, KEIDAR I, SHRAER A. Trusting the cloud [J]. Acm Sigact News, 2009, 40(2): 81-6.

[3] SHAIKH F B, HAIDER S. Security threats in cloud computing; proceedings of the Internet Technology and Secured Transactions, F, 2011 [C].

[4] SANDHU R S. Access Control: Principles and Practice [J]. IEEE Communications Magazine, 1999, 32(9): 40-8.

[5] 冯登国, 张敏, 张妍, et al. 云计算安全研究 [J]. 软件学报, 2011, 01): 71-83.

[6] CONWAY R W, MAXWELL W L, MORGAN H L. On the Implementation of Security Measures in Information Systems [J]. Communications of the Acm Cacm Homepage, 1972, 15(4): 211-20.

[7] DOROTHY E. A lattice model of secure information flow [J]. Communications of the Acm, 1976, 19(5): 236-43.

[8] BHARADWAJ V G, BARAS J S. Towards automated negotiation of access control policies; proceedings of the IEEE International Workshop on Policies for Distributed Systems and Networks, 2003 Proceedings Policy, F, 2003 [C].

[9] 邵佩英, 孙淑玲. 基于传统DBMS的强制访问控制安全功能的设计与实现 [J]. 计算机工程与应用, 1999, 8): 58-60.

[10] BERTOLINO A, BUSCH M, DAOUDAGH S, et al. A Toolchain for Designing and Testing XACML Policies; proceedings of the IEEE Sixth International Conference on Software Testing, Verification and Validation, F, 2013 [C].

[11] 沈海波, 洪帆. 基于属性的授权和访问控制研究 [J]. 计算机应用, 2007, 01): 114-7.

[12] SAHAI A, WATERS B. Fuzzy Identity-Based Encryption [M]//CRAMER R. Advances in Cryptology – EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005 Proceedings. Berlin, Heidelberg; Springer Berlin Heidelberg. 2005: 457-73.

[13] YUAN E, TONG J. Attributed based access control (ABAC) for Web services; proceedings of the IEEE International Conference on Web Services, 2005 ICWS 2005 Proceedings, F, 2005 [C].

[14] LING C, COOLEY J A, KHAZAN R I, et al. Collusion-Resistant Group Key Management Using Attribute-Based Encryption [J]. 2007, 2007(

[15] YU S, WANG C, REN K, et al. Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing; proceedings of the Conference on Information Communications, F, 2010 [C].

[16] GOYAL V, PANDEY O, SAHAI A, et al. Attribute-based encryption for fine-grained access control of encrypted data [J]. Proc of Acmccs’, 2010, 89-98(89-98.

[17] OSTROVSKY R, SAHAI A, WATERS B. Attribute-Based Encryption with Non-Monotonic Access Structures; proceedings of the Ccs 07 Acm Conference on Computer & Communications Security, F, 2007 [C].

[18] BETHENCOURT J, SAHAI A, WATERS B. Ciphertext-Policy Attribute-Based Encryption [J]. 2007, 2008(4): 321-34.

[19] LI J, REN K, ZHU B, et al. Privacy-Aware Attribute-Based Encryption with User Accountability; proceedings of the International Conference on Information Security, F, 2009 [C].

[20] LI J, HUANG Q, CHEN X, et al. Multi-authority ciphertext-policy attribute-based encryption with accountability; proceedings of the ACM Symposium on Information, Computer and Communications Security, F, 2011 [C].

[21] LI J, JIA C, LI J, et al. Outsourcing Encryption of Attribute-Based Encryption with MapReduce; proceedings of the International Conference on Information and Communications Security, F, 2012 [C].

[22] LIANG X, CAO Z, LIN H, et al. Attribute Based Proxy Re-encryption with Delegating Capabilities [J]. Aisaccs Pages–, 2009, 276-86.

[23] LUO S, HU J, CHEN Z. Ciphertext Policy Attribute-Based Proxy Re-encryption [M]. 2010.

[24] OASIS. eXtensible Access Control Markup Language (XACML) Version3.0 [EB/OL].

.2014-4-17, http: //docs.oasis-open.org /xacml/3.0/xacml-3.0-core-spec-os-en.pdf

[25] 苏金树, 曹丹, 王小峰, et al. 属性基加密机制 [J]. 软件学报, 2011, 22(6): 1299-315.

[26] 张玉凤. 基于CPABE的云存储访问控制研究 [硕士学位论文]; 华中科技大学, 2013.

[27] 令狐雄展. 云计算中基于属性的访问控制技术研究 [硕士学位论文]; 西安电子科技大学, 2014.

[28] 冯黎晓. 云计算环境下基于属性的访问控制方法研究 [硕士学位论文]; 内蒙古科技大学, 2014.

[29] 王蕾. 基于属性的云存储访问控制系统--安全云存储子系统 [硕士学位论文]; 西安电子科技大学, 2014.

[30] 牛德华, 马建峰, 马卓, et al. 基于属性的安全增强云存储访问控制方案 [J]. 通信学报, 2013, S1): 276-84.

[31] 李辰楠. 基于属性的云存储访问控制系统 [硕士学位论文]; 西安电子科技大学, 2014.

[32] 刘晓建, 王力生, 廖新考. 基于CP-ABE和XACML多权限安全云存储访问控制方案 [J]. 计算机科学, 2016, 3):

[33] XACML open source project SUNXACML[EB/OL].<http://sunxacml.sourceforge.net>

[34] Advanced crypto software collection[EB/OL].<http://acsc.cs.utexas.edu/cpabe>

[35] BETHENCOURT J, SAHAI A, WATERS B. Ciphertext-Policy Attribute-Based Encryption [M]. Proceedings of the 2007 IEEE Symposium on Security and Privacy. IEEE Computer Society. 2007: 321-34.

[36] GOYAL V, PANDEY O, SAHAI A, et al. Attribute-based encryption for fine-grained access control of encrypted data [M]. Proceedings of the 13th ACM conference on Computer and communications security. Alexandria, Virginia, USA; ACM. 2006: 89-98.

附录：核心代码

附录一 文件浏览代码

listfile(File file, Map<String, String> map){

if (!file.isFile()) {

File files[] = file.listFiles();

for (File f : files) {

listfile(f, map);

}

} else {

String realName = file.getName().substring(

file.getName().indexOf("\_") + 1);

map.put(file.getName(), realName1);

}

}

附录二 策略生成核心代码

String policyName = (String) policyDefine.get("policyName");

…

while (iterator.hasNext()) {

String key = iterator.next().toString();

JSONObject ruleItem = PolicyRule.getJSONObject(key);

RuleMy rule = new RuleMy();

rule.setName((String) ruleItem.get("ruleName"));

rule.setEffect((String) ruleItem.get("ruleEffect"));

,,,}

// create the target for the policy

ptf = new PolicyTargetFactory(UploadServlet.nameofFile);

…

try {

URI ruleId = new URI(ruleName);

int effect = ruleEffect.equals("permit") ? Result.DECISION\_PERMIT : Result.DECISION\_DENY;

ruleTargey = rtf.createRuleTarget();

Apply condition = null;

if (rc.getConditionValue() != null) {

condition = rc.createRuleCondition();

}

Rule rule = new Rule(ruleId, effect, null, ruleTargey, condition);

ruleList = new ArrayList<>();

ruleList.add(rule);

…

Rule defaultRule = new Rule(new URI("FinalRule"), Result.DECISION\_DENY,

null, null, null);

ruleList.add(defaultRule);

policyId = new URI(policyName);

combiningAlg = caf.getRuleAlg(combiningRuleAlgorithm);

policyTarget = ptf.createPolicyTarget();

Policy policy = new Policy(policyId, combiningAlg, description, policyTarget, ruleList);

String outPath = "D:\\Ecilpse\\workspace\\AccessControlXACML\\policy\\" + policyName + "\_policy.xml";

policy.encode(new FileOutputStream(outPath), new Indenter());

附录三 请求结果生成核心代码

String operation = request.getParameter("operation");

String data = request.getParameter("data");

String user = (String) request.getSession().getAttribute("user");

List<String> resAttributes = gri.getResAttributes();

List<String> userAttributes = gui.getUserAttributes();

…

cr = new createRequest(userAttributes, resAttributes, operation);

cr.requestBuilder(requsetGeneratePath, false);

…

附录四 CPABE加解密算法核心代码

public void setup(String PKFileName, String MKFileName) {

PKFileName = isEmptyString(PKFileName) ? Default\_PKFileName

: PKFileName;

MKFileName = isEmptyString(MKFileName) ? Default\_MKFileName

: MKFileName;

try {

CPABEImpl.setup(PKFileName, MKFileName);

} catch (Exception e) {

e.printStackTrace();

}

}

public void keygen(String[] attrs, String PKFileName, String MKFileName, String SKFileName){

SKFileName = isEmptyString(SKFileName) ? Default\_SKFileName : SKFileName;

PublicKey PK = SerializeUtils.unserialize(PublicKey.class, new File(PKFileName));

if(PK == null){

err(Error\_PK\_Missing);

return;

}

MasterKey MK = SerializeUtils.unserialize(MasterKey.class, new File(MKFileName));

CPABEImpl.keygen(attrs, PK, MK, SKFileName);

}

public void enc(String encFileName, String policy, String outputFileName, String PKFileName) {

…

File encFile = new File(encFileName);

if (!encFile.exists()) {

err(Error\_EncFile\_Missing);

return;

}

if (encFile.isDirectory()) {

err(Error\_Enc\_Directory);

return;

}

try {

outputFileName = isEmptyString(outputFileName) ? encFile

.getCanonicalPath() + Ciphertext\_Suffix : outputFileName;

} …

CPABEImpl.enc(encFile, p, PK, outputFileName);

encFile.delete();

}

public void dec(String ciphertextFileName, String PKFileName, String SKFileName){

…

try {

dis = new DataInputStream(new FileInputStream(new File(ciphertextFileName)));

} catch (FileNotFoundException e) {

e.printStackTrace();

}

…

Ciphertext ciphertext = SerializeUtils.\_unserialize(Ciphertext.class, dis);

PublicKey PK = SerializeUtils.unserialize(PublicKey.class, new File(PKFileName));

String output = null;

if(ciphertextFileName.endsWith(".cpabe")){

int end = ciphertextFileName.indexOf(".cpabe");

int begin = ciphertextFileName.lastIndexOf("\_") + 1;

//output = ciphertextFileName.substring(0, end);

output= "D:\\Ecilpse\\workspace\\AccessControlXACML\\FileUploade\\plainText\\"

+ ciphertextFileName.substring(begin, end);

}

else{

//output = ciphertextFileName + ".out";

int begin = ciphertextFileName.lastIndexOf("\_") + 1;

output = "D:\\Ecilpse\\workspace\\AccessControlXACML\\FileUploade\\plainText\\"

+ ciphertextFileName.substring(begin, ciphertextFileName.length()-1) + ".out";

}

File outputFile = CPABEImpl.createNewFile(output);

OutputStream os = null;

Element m = CPABEImpl.dec(ciphertext, SK, PK);

AES.crypto(Cipher.DECRYPT\_MODE, dis, os, m);

}

附录五 notePolicy.xml文件

<Policy PolicyId="notePolicy" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable">

<Description>any whose school is CUMT can delete it</Description>

<Target>

<Subjects>

<AnySubject/>

</Subjects>

<Resources>

<Resource>

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

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

<ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" DataType="http://www.w3.org/2001/XMLSchema#string"/>

</ResourceMatch>

</Resource>

</Resources>

<Actions>

<AnyAction/>

</Actions>

</Target>

<Rule RuleId="ruleNote" Effect="Permit">

<Target>

<Subjects>

<AnySubject/>

</Subjects>

<Resources>

<AnyResource/>

</Resources>

<Actions>

<Action>

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

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

<ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string"/>

</ActionMatch>

</Action>

</Actions>

</Target>

<Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">

<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">

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

</Apply>

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

</Condition>

</Rule>

<Rule RuleId="FinalRule" Effect="Deny"/>

</Policy>

翻译

英文原文

**Biometric Encryption to Enhance Confidentiality  
in Cloud Computing**

Abstract—Virtualization technology is the base technology used in Cloud computing. Therefore, virtualization enables Cloud computing to provide hardware and software services to the users on demand. Actually, many companies migrates to the Cloud computing for many reasons such as capabilities of processor, bus speed, size of storage, memory and managed to reduce the cost of dedicated servers. However, virtualization and Cloud computing contain many security weaknesses that affects the biometric data confidentiality in the Cloud computing. Those security issues are VM ware escape, hopping, mobility, diversity monitoring and etc. Furthermore, the privacy of a particular user is an issue in biometric data i.e. the face reorganization data for a famous and important people. Therefore, this paper proposed biometric encryption to improve the confidentiality in Cloud computing for biometric data. Also, this paper discussed virtualization for Cloud computing, as well as biometrics encryption. Indeed, this paper overviewed the security weaknesses of Cloud computing and how biometric encryption can improve the confidentiality in Cloud computing environment. Apart from this, confidentiality is enhanced in Cloud computing by using biometric encryption for biometric data. The novel approach of biometric encryption is to enhance the biometric data confidentiality in Cloud computing.

Keywords -Biometric Encryption, Virtualization, Cloud computing.

Ⅰ.INTRODUCTION

Cloud computing is not a new technology; rather it is anew way of delivering computing resources and services. The development of Cloud computing has brought new changes and opportunities to IT industry. Its general purpose is to dynamically allocate scalable resources to multiple users. Users of the cloud can acquire on-demand basis cloud services and access it globally. Cloud computing also provides measured services, that is to say customers only pay for what they use [1]. Furthermore, Cloud computing offers an effective way to reduce IT expenses, Capital Expenditure (CapEx), and Operational Expenditure (OpEx), [1, 2], and thus it offers economic benefits to users and organizations. It utilizes many technologies that make the cloud environment today. Among these technologies which Cloud computing utilizes is Virtualization. Virtualization is a technology commonly defined as abstraction or execution environment which hides the complexity of hardware layer and allows multiple operating systems to run without the need of real hardware [3, 4]. Virtualization has also been used to offer dynamic resource allocation and service provisioning, particularly in IaaS Cloud environment [1]. After all, virtualization plays a big role in making Cloud computing environment. Cloud computing environment solely depends on virtualization technology to deliver its business services SaaS, PaaS, and IaaS. It is very important to understand virtualization technology at all levels and not only focusing in CPU, Memory, rather virtualization now involves also application system storage and networking.

II. VIRTUALIZATION IN CLOUD COMPUTING

Virtualization technology is one of the base technologies used in Cloud computing. Back in 1960s when applications were multiplexed on very high cost mainframes there was a need for a technology that allow for as much as possible of resource utilization, and hence virtualization technology emerged. Virtualization uses virtual machine monitor VMM in hardware resource multiplexing, server consolidation and to support simultaneous execution of multiple instances of Oss (typically called Guest OS) [3, 5], therefore, VMM can take control over the executing flow of the guest OS. VMM is a thin layer software layer which conventionally runs on a machine's hardware. It also might run on top of a host OS on the host system. VMM is responsible of managing the VMs and allows no direct interaction with the host hardware. Guest OSs running on VMs interacts with the host systems resources only via the VMMs. The VMM typically runs in the most privileged level and considered trusted component while the guest OS is considered not trusted and hence runs on user mode. Later on, as technology advanced which causes advancing in the capabilities of processor, bus speed, size of storage, memory and managed to reduce the cost of dedicated servers, there was almost no need to utilize such a technology as virtualization [4]. Despite all aforementioned aspects of new technology, new challenges arose. This new advanced cheap technology led to increase of underutilized machines that brought upon it a significant space and management overhead. Therefore, organizations realized the necessity of using VM [4], as they could not afford keeping track of every server’s application versions, patches. In addition, securing these servers became a huge burden for the organizations. Therefore, to overcome these challenges organizations moved back to VMs, merged those VMs onto few physical servers and efficiently managed those VMs via VM monitor VMM; sometime refer to as the hypervisor.

III. TYPES OF VIRTUALIZATION

There are two types of virtualization environments in which VMM is deployed. Type I is known as Full virtualization where VMM is interfacing directly with the system hardware this type of architecture is also called as native architecture [6], Figure 1. Type II is not interfacing directly with the hardware of the system, rather it runs as an application alongside with the host OS. Type II is called Para-virtualization Figure 2. We would like to discuss these two types and how these two architectures impact the security of virtualization environment such as virtualized cloud environment.

A-Full virtualization

Full virtualization is considered when the hypervisor is implemented directly on top of physical hardware or embedded in the host OS kernel. It is also called hardware virtualization because shared resources such as device drivers and hardware layer resources are virtualized by the VMM for guest OS. An example of this type is the Xen hypervisor, KVM hypervisor.

B- Para virtualization

Para virtualization is commonly deployed on those machines that don’t support Full virtualization such as Intel “x86” architecture. It runs as software and it is enabled by the host OS which provides I\O drivers and bootstrapping code[4, 7, 8]. This type is known to be less secure because no matter how secure the VMM is, it is effected by the security of the host OS itself. An example of this virtualization architecture is the implementation of virtual environments using VMware, Sun VirtualBox, and Microsoft Virtual PC. In both the above architectures ,VMM’s main goal is to support three key attributes in virtual environment [4, 9]:

- Isolation

- Interposition  
- Inspection  
Isolation  
 VMM or the hypervisor provides isolation through virtual memory abstraction and will not let the VMs to share physical memory. Thus, It lets the VMs to think that each of them is possesses its own address space and has full control over it.

Interposition  
 This key requires that VMM has the capability to manage all privileged operations on a physical hardware. This means that VMM is always interfacing with the physical hardware and mediate all the requests from the guest OS to the underlying hardware. This provides some level of security. It ensures that no direct interaction with   
lower hardware layer.

Inspection  
 Inspection refers to VMM have full access to all VMs states. This includes CPU, memory, and device states. However, Virtualization has now primarily contributed as solution for security, reliability, and administration in that along with its associated techniques help to solve number of security issues. According to IBM researcher in [10], a single system could implement a multiple-level secure system by dividing it into multiple single- level virtual systems and securely separating them.

IV. VIRTUALIZATION ADVANTAGES

1. Resource

Pool This characteristic provides a uniform abstraction of resources [4]. This means that a physical machine is not viewed as limited entity with particular fixed capabilities, in fact, this feature shows how virtualization consolidates a collection of VMs onto a single machine. Thus, it lowers resource costs and space requirements.

1. Flexibility

This feature allows the user to run multiple instances of an operating system one a single computer. VMs can be migrated easily to another physical machine. It is also possible to change the specifications of virtual computers while they are running; Adding RAM, HDD [11].

1. Availability

Availability ensures that VM image can continuously run in the need of shutting down the physical node [11]. This means if an upgrade or maintenance is to be done to that physical machine; it can be done without affecting the availability of the VM instance. Therefore, a VM hosted in the particular physical machine needs to be migrated to another physical machine and can be restored back after maintenance is complete.

1. Scalability

This feature shows the virtualization technology strong involvement in making up the cloud. Scalability is considered one of the basic features of virtualization. It assures the process of adding or removing VM instances when the demand for capacity or new VMs increases over time [11].

1. Cost

Virtualization offers effective cost-reducing due to resource utilization. This can be achieved by consolidating number of servers into one physical server, hence reducing the capital expenditure (CapEx)[11].  
f) Security

Though many argue that virtualization is facing security challenges, it offers isolation between the VM instances. Isolation provides encapsulation, i.e. no VM is allowed to communicate directly with the other. However, isolation, if not deployed properly, it pauses threat by itself. This emphasizes that virtualization offers some level of security. For example, a web server VM hosted in a machine alongside with database VM and email VM and If the attacker compromised that web server VM the other VM instances are unaffected.

V. BIOMETRICS ENCRYPTION

Authentication is a mechanism used to identify people and to provide the authorization for them. Authentication and identification have been used in many fields to provide proper security mechanisms. Authentication and identification are used in access control, verification, security audit, etc. Moreover, identification can be achieved by using the behavior features or physiological features. Apart from this, the science of using these features is called biometric identification.

Biometric identification includes iris, voice, fingerprint, face reorganization and etc. Biometric identification is more flexible authentication method than the secret key, because it is extremely difficult to be forgotten or even lost. However, the secret key is considered more secure approach, because some Biometric identification has been hacked using fake Biometric information attack, such as fake fingerprints attacks[12]. In public security and banks, biometric identification has been applied as a strong security mechanism and strong solution [13].  
 Biometric encryption (BE): it is a solution that has been used to protect biometric identification. Indeed, cryptography was the solution to overcome the threats on biometric identification. Therefore, encryption was proposed as a patent by Bodo[14]. After that, the first version of biometric encryption was completely proposed by other researchers [15]. Biometric encryption also was proposed for face reorganization to enhance the privacy [16]. Also, biometric encryption has been used in Telehealthcare systems [17]. Furthermore, biometric encryption was implemented in a standalone system using fingerprint fuzzy fault schema[18] . In addition, a new framework of biometric encryption was proposed with filter-bank based fingerprint feature [19]. Besides, the security of Mobile-ad hoc network was enhanced through unimodal biometric encryption key[20].Moreover, a cell phone was developed with biometric encryption based on user’s behavior to authenticate the cell phone user[21]. Actually, biometric technology uses three models: key release, key binding and key generation. Recently, the feasibility of deploying biometric encryption has been conducted for mobile Cloud computing [22].  
Biometric encryption differs from normal password encryption, because it merges the biometric image with random generated key using BE binding algorithm to generate Biometrically-encrypted key. In the decryption process, the Biometrically-encrypted key is merged with the biometric image using BE retrieval algorithm to get the key retrieved [23].  
Cloud Computing Security Issues and BE Effects

Despite all those charming features and strong support that virtualization technology provides to the computing environment that utilizes it. It has a huge impact on the development of these relying environments. Although, as mentioned before, virtualization is not new technology as Cloud, it has several security issues. These security issues have migrated into Cloud computing environment. Most of these security issues are based on the virtualization level.  
 However, Biometric encryption provides extra level security for privacy against these security issues. It is important to mention that BE does not solve the security issues in Cloud computing. But, it secures Biometric data against most of these security issues. BE provides extra level of security for the attacker in the Cloud which is the encryption mechanism.  
A. VM Hopping  
 This kind of threat is directly affecting the customers/users of the Cloud. It is always due to resource sharing, a feature which virtualization offers. Resource sharing allows the physical machine and its resources such as CPU, memory networking and storage to be shared among the users reside in that particular physical machine, these users/tenants are called residents and thus the machine is going to provide co-residency/multi-tenancy [24]. Although multi-tenancy is one of the most crucial goals of virtualization, it is considered as the weakest point in a chain. It exposes virtualized environments to huge risk and hence, it is huge threat to Cloud computing. However, this attack cannot directly occur from outside. This means an attacker needs to be placed in that particular physical machine as where his target resides [25]. Although, the attacker should follow some steps vary from allocating his victim to execute arbitrary commands, this should allow the attacker to be able to break the isolation control and attack his victim [26].

BE effects: when VM hopping occurs, the attacker can get access to neighbor’s VM ware; however, the biometric data is encrypted and it is not readable for the attacker. However, if the attacker could control the whole VM ware and get access to the biometric device, the biometric information will be readable for the attacker.  
B. VM Mobility  
 VM mobility addresses a crucial security issue [24-26]. VM mobility offers some flexibility in migrating VM instance from storage to another over the network. This security issue is usually due to weak configuration of the cloud network environment. Although this flexibility provides robust support to physical security to protect against stealing the storage, yet it leads to other security issues such as, it allows the attacker to get a copy of the VM which is transmitted from a machine to another. This type of breach violates the confidentiality and integrity of users’ data and thus shows that users’ data is at risk when it is stored in the cloud. However, mitigating the risk caused by VM Mobility is not a duty of the users [26]. Cloud providers also share some responsibilities in reducing such risk. Therefore, security management of users data stored in the cloud should be written in a Service Level Agreements (SLA) that clearly states the obligations of both Cloud providers and Cloud users.

BE effects: even though the attacker can get a copy of  
the VM, the attacker gets encrypted biometrics data.  
C. VM Diversity  
Virtualization technology offers an ease of use in which it allows the users to efficiently create many VMs. This freedom shows that securing and managing these VMs is a huge burden due to various OSs that can be deployed in seconds [26]. This kind of diversity makes VM security management a challenge. However, a proper SLA could help address this issue. Looking at Cloud computing delivery service models, we can say that VM security management in this context is not the responsibility of the cloud service provider (CSP) alone, in fact, it is also the responsibility of the cloud user. For example, in IaaS, the CSP must ensure security and robustness of the underlying infrastructure such as the hypervisor, whereas the user must properly ensure that his VM and offered service is configured properly and secured, and this includes keeping the guest OS patched and up-to-date. However, this is not limited to IaaS. PaaS service model also requires some security and maintenance but PaaS is somewhat robust against VM diversity.

BE effects: different VMs may contain different types of operating systems. These operating systems contain different weaknesses, so it is the duty of the VMs owners to keep their VMs updated as well as BE. This approach can protect their data in case of intrusion to their VMs.

D. Denial of service  
 Denial of service is shutting down an available service; also it is blocking the performance and the functionality of the source. Virtualization environment is a shared resources environment where VMs share the same CPU, memory, bandwidth and disk. Furthermore, the provider always has limited capacity of those resources. Apart from this, one VM can exhaust these shared physical resources which will cause denial of services. Moreover, if the VMs in the same physical machine which use the shared resources at the same time, denial of service will occur. The provider should be aware of the maximum usage of its resources which the VMs can use. Also the provider should configure its resources in a proper manner to prevent denial of services. Denial of service is a serious matter that should be treated with well configuration and monitoring [24, 26-29]

BE effects: when the service is down, there are no effects for biometrics encryption. Biometric encryption targets the confidentiality of biometric information only and it is not targeting the availability of the data.

E. VM Escape

VMM is designed to allow VMs to share system resources in controlled approach. Therefore, VMM must enforce isolation between VMs and system resources by preventing any direct interaction with the lower hardware layer. Exploiting a compromised VM in a way that allows an attacker to take control over the hypervisor is known as VM escape. VM escape [24], the program running in a virtual machine is able to completely bypass the virtual layer (hypervisor layer), and get access to the host machine. Thereby, it escalates to root privileges, basically escape from the virtual machine privileges. This vulnerability will allow the attacker to likely have control over all guest Oss resulting in a complete breakdown of the security framework of the environment [11, 28]. VM escape is the worst case when the isolation between the VMs and host is compromised.

BE effects: when VM escape occurs, the attacker will control the main host machine and get the root privileges. Therefore, the attacker can have access to all the resource as well as biometric devices, so the attacker can read the biometric information. When VM escape occurs BE does not help to protect biometric information.

F. VM Monitoring

The primary characteristic in virtualization technology is isolation VMs from each other inside a single host. Apart from this, without a proper configuration for the host machine, one VM can get an access and monitor other VMs. This means the provider lost the confidentiality of the system. One VM can lunch a cross-VM side channel attack to extract the memory information about the victim. Moreover, ARP poisoning can be lunched to turn the traffic from the victim VM to the attacker VM, which allows the attacker to monitor all the traffic of the victim VM. Monitoring VMs is a critical issue which the provider must prevent to keep the confidentiality of its system [24].

BE effects: if the attacker could sniff the data of another VMs, the data of BE will be encrypted which will provide other security layer and extra challenges for the attacker.

G. The communication between virtual machines A clipboard in virtual machine technology allows the communication between the hosts VMs; this is very convenient method to do the communication between hosts. However, in the same time, malicious can be transferred easily as well [29]. Another example, some of the providers do not apply full isolation for the VMs by allowing the VMs to access the host virtual machine and use a common application. This action is very risky and proper isolation should be applied [30]. In virtual machine environment, encryption is a good practice to keep the transferred data secured and confidential.

BE effects: malicious codes can transmit the desired data to the attacker or allow the attacker to apply a successful intrusion to other VMs. BE keeps the biometric data encrypted and not readable for the attacker.

H. Host Control virtual machines

The host is the controller of all the virtual machines; the host is the manager, the inspector and the protector of the virtual machines on it. So the host can monitor all the traffic of those virtual machines, and control them as well. Different studies on the virtualization technology showed that the host can affect the VMs [29]. So the host security is very important to secure the virtual machines, also a proper configuration should be applied and access control restriction to the host to keep all the virtual machines secured as well.

BE effects: it is a good practice to keep the data of the VMs, encrypted in the host; therefore, even if the host is controlled by an attacker, the attacker cannot read the encrypted biometrics data. However, if the attacker could gain the root privileges, biometric information will be readable to the attacker.

VI. CONCLUSION

Virtualization has some security issues. These security issues migrated to the cloud environments which is affecting the confidentiality of biometric data. Biometric encryption is a solution which is proposed in this paper. Actually, Biometric encryption provides confidentiality to the biometrics data in the cloud. Therefore, Biometric encryption is highly recommended to be implemented in the Cloud computing. Indeed, this approach provides extra security level to the Cloud computing. Moreover, it overcomes many security weaknesses in Cloud computing related to biometric data confidentiality. Indeed, confidentiality is enhanced in Cloud computing by using biometric encryption for biometric data. However, Biometric encryption does not solve the security issues in Cloud computing, rather it secures Biometric data against most of those security issues. Until this moment, there is no related research studied Biometric encryption in Cloud computing. However, biometric encryption in Cloud computing will be implemented in the future. Therefore, biometric encryption for biometric data in Cloud computing will be deployed and evaluated as well.

中文译文

**生物加密加强云计算保密性**

摘 要

在云计算中虚拟化技术是基本的使用技术。因此，虚拟化可以使云计算按照用户的需求为其提供硬件和软件服务。事实上，很多公司将其业务迁移到云计算的原因很多，比如处理器能力，总线速度，存储器大小，内存而且可以有效减少专用服务器的成本。但是，虚拟化和云计算中存在着很多安全缺陷会影响云计算中生物数据的保密性。这些安全问题有虚拟机设备的逃离、跳跃、流动性多样性监测等等。进一步说，一个特定用户的隐私在生物数据中是一个问题，比如，对于一个著名并且重要的人的面部识别数据。因此，本文提出了生物加密来提高云计算环境下生物数据的保密性。并且，这篇文章还讨论了云计算、虚拟化，和生物加密。这篇文章综述了云计算的安全缺陷和生物加密如何改善云计算环境下的保密性。除此之外，通过对生物数据使用生物加密的方法云计算环境中保密性得以加强。这种新的生物加密方法加强了云环境下的生物数据保密性。

关键词：加密，虚拟化，云计算

I. 引言

云计算并不是一种新兴技术，反而这是一种提供计算资源和服务的新方法。云计算的发展已经非IT 工业带来了新的改变和机会。它一般的目标是为众多用户动态分配可扩展的资源。云用户可以获得按需求的基础云服务并且可以全球进行访问。云计算也提供了计量方法，这样消费者仅仅为其所使用的付费。此外，云计算提供了一种有效的方法来降低IT 花费，资本支出和运营支出，这样它给用户和机构提供了经济上的好处。它利用了很多技术使云环境变成了今天这样。云计算利用的技术之一就是虚拟化。虚拟化通常如下定义：作为隐藏硬件层和允许多操作系统运行而不需真实的硬件的抽象或者可执行环境。虚拟化已经被使用来提供动态的资源分配和服务提供，特别是在云环境下的Iaas 层。毕竟，虚拟化在云计算机环境中扮演着一个很重要的角色。云计算环境仅依靠虚拟化技术就可以传递其商业服务给SaaS，Paas和Iaas。理解虚拟化技术在所有层都是非常重要的，不仅仅要关注CPU ，内存，虚拟化现在已经涉及到应用系统存储和网络当中

II. 云计算中的虚拟化

虚拟化技术在云计算中是基础技术之一。早在20世纪60年的当应用程序的复用成本非常高，大型机必须要有一种技术来允许尽可能多得利用资源，虚拟化技术应运而生。虚拟化在资源复用，服务器整合和支持操作系统多实例同时执行中使用虚拟机监测技术VMM。因此VMM 可以客户机操作系统的执行流。VMM 是一个通常运行在机器硬件上的软件层。它也可能运行在主机系统上一个主系统的顶层。VMM 负责管理虚拟机，与主机硬件的非直接互动。运行VM 上的客户机操作系统有主机系统资源交互仅仅通过VMM。VMM 通常运行在最高特权级别而且认为是可信的成分，而客户操作系统认为不被信任，因而对用户模式运行。后来上，随着技术的进步而导致在处理器能力，总线速度，存储大小，内存和有效降低专用服务器成本能力的进步，通常不再需要理由这样的技术作为虚拟化。尽管有新技术的以上方面，新的挑战也出现了。这种新的高级而低廉的技术导致了未充分利用机器的增加，从而带来了显著的空间和管理开销。因此，组织意识到了使用VM 的必要性，因为他们无法负担跟踪每一个服务器应用程序的版本，补丁。此外，这些服务器的安全对组织者也成为了一个巨大的负担。因此为了克服这些挑战，组织回到了虚拟机，合并这些虚拟机到几个物理服务器并且通过虚拟机监视器VMM 高效管理这些虚拟机，有时也被称为管理程序。

Ⅲ. 虚拟化类型

有两种类型的VMM 部署虚拟化环境。类型一被称为全虚拟化，VMM 与系统硬件直接交互，这种框架也被称为本地框架，如图1.类型二不与系统的硬件直接进行交互，而是作为一个与主机操作系统一起运行的应用程序。类型二被称为准虚拟化，如图2.我们讨论了这两种类型和这两种架构如何影响虚拟化环境的安全性，如虚拟化云环境。

1. 全虚拟化

全虚拟化是当虚拟机管理程序是被认为是直接在物理硬件之上实现或嵌入主机OS 内核。它也被称为硬件虚拟化，因为共享资源，例如设备驱动程序和硬件层资源在客户操作系统被VMM 虚拟化。这种类型的一个例子是在Xen 虚拟机管理程序，KVM 管理程序。

1. 半虚拟化

虚拟化通常部署在那些不支持全虚拟化，例如英特尔的“x86”架构机器。它作为一个软件运行并且它是由主机操作系统提供启用I \ O 驱动程序和引导代码[4，7,8]。这种类型是交不安全的，因为不管VMM 怎么安全，它是由主机OS 本身的安全性实现的。这种虚拟化架构的一个例子是虚拟化环境的实现如使Vmware ，太阳的VirtualBox 和微软的Virtual PC。 在上述两种结构中，VMM 的主要目标是在虚拟环境中支持三个关键属性：

-隔离

-介入

-检验

隔离

VMM 或虚拟机管理程序通过隔离虚拟内存的抽象，不会让MVs 共享物理内存来提供隔离性。因此，它可以让虚拟机认为他们每个人拥有自己的地址空间并有对它的完全控制。

介入

这关键需要VMM 有能力管理上的物理硬件的所有特权操作。这意味着VMM 始终可以物理硬件交互并且能够调解所有从客户端操作系统到底层硬件的请求。这提供了一定程度的安全性。它确保与较低的硬件层没有直接的互动。

检验

检验是指VMM 可以完全访问所有虚拟机状态。这包括CPU ，存储器和设备的状态。但是，虚拟化现在主要的贡献是针对安全性，可靠性和管理性作为一个解决方案，来与其它相关技术一起解决安全问题。根据IBM 的研究[10]，单一系统可以实现一个多级安全系统，通过将其分为多个单级的虚拟系统，安全地将它们分离的方法。

Ⅳ. 虚拟化优势

1. 资源池

这一特性提供了一个均匀的抽象资源[4]。这意味着，一个物理机器不被视为特别固定的功能实体的限制，其实这个功能显示虚拟化技术如何整合虚拟机到一台机器。因此，可以资源成本和空间需求。

1. 灵活性

此特性允许用户可以在单台计算机上运行一个操作系统的多个实例。虚拟机可以轻松迁移到另一台物理机器。 也可以在它们运行期间改变的虚拟计算机的配置，如添RAM ，硬盘[11]。

1. 可用性

可用性确保虚拟机映像在需要关闭物理节点[11] 可以连续的运行。这意味着，如果需要对物理机做升级或维护工作；它可以在不影响进行虚拟机实例的可用性的情况下实现。因此，当托管在特定的物理机器的虚拟机需要迁移到另一台物理机，当维护完成可以恢复过来。

1. 可扩展性

这项特征显示虚拟化技术参与组成的云的强大。可扩展性是虚拟化考虑的的基本特征之一。 它确保当容量的需求或新虚拟机的增加超过时间[11] 时添加或删VM 实例的过程。

1. 成本

虚拟化提供了通过资源利用有效降低了成本。这可以通过合并多个服务器的到一个物理服务器上，从而减少可实现成本支出（CAPEX）[11]。

1. 安全性

尽管许多人认为，虚拟化正面临着安全挑战，但它提供了虚拟机之间的实例隔离。隔离提供封装，比如说即没有VM 是允许直接与其他VM 通信。然而，隔离如果不能正确使用，它本身就会造成威胁。这强调了虚拟化提供了一定程度的安全性。例如，web 服务器虚拟机与数据库虚拟机和电子邮件的虚拟机在主机上一起使用。如果攻击者攻破Web 服务器虚拟机但其他VM 实例是不受影响。

Ⅴ. 生物加密

认证是用来识别人并为他们提供的授权的机制。认证和识别在很多领域被用于提供适当的安全机制。认证和识别在访问控制，验证，安全审计等中使用。此外，标识可通过使用行为特征或生理来实现。除此之外，使用这些特征的科学被称为生物特征识别。

生物特征识别包括虹膜，语音，指纹，人脸重组等。生物特征识别比秘钥更加灵活，因为它是非常困难的被遗忘甚至丧失。但是，密钥是更安全的方法，因为有些生物识别标识已经使用假生物入侵信息被攻击，如假指纹的攻击[12]。 在公安，银行，生物识别已经作为一个强大的安全机制和解决方法[13]被应用。

生物加密是一个被用来保护生物特征信息的解决方案。事实上，加密被用来作为解决生物特性识别微信的解决方法。因此，Bodo 将加密提出来并作为一个专利[14]。之后， 其他研究人员提出了第一个完全的生物加密版本[15]。生物加密也被用来加强人脸识别的私密性[16]。并且，生物加密已经在Telehealthcare 系统[17]中使用。此外，生物加密在一个独立的系统已经被实现，使用指纹模糊故障策略[18]。而且，移动自组织网路的安全性通过单峰生物加密得到加强[20]。另外，由于生物加密基于用户行为来认证手机使用者的手机发展迅速[21]。实际上，生物识别技术采用三种模式：秘钥发放，秘钥绑定和密钥生成。近日，在移动云计算[22] 部署生物识别加密的可行性已经在进行。

生物特征加密与正常密码都不同，加密，因为它将融合了生物图像与使用具有约束力算法来生成生物特征加密密钥的随机产生密钥。在解密过程中，生物特征加密密钥被合并到生物识别图像，使用BE 检索算法来获得秘钥[23]。

云计算安全问题与BE影响

尽管虚拟化给利用它的云计算环境提供了这些迷人的特征和强大的支持。它对这些依赖环境的发展有着巨大的影响。尽管如前面所提到，作为云虚拟化并不是一种新的技术，但它有几个安全问题。这些安全问题已经迁移到云计算环境中。大多数这些安全问题是基于虚拟化的水平。

然而，生物加密提供了额外程度的安全对于这些问题的隐私。云提到计算中BE 不能解决安全问题是非常重要的。但是它解决了生物数据的了大多数的安全问题。BE 提供了额外程度的安全保证对于云中的攻击者，这被称为加密机制。

1. VM 跳跃

这种威胁是直接影响到云中的客户/ 用户。它常常因为资源共享，虚拟化提供的的功能造成。资源共享允许在物理机和它的资源，如CPU， 内存网络和存储共享驻留在特定物理机用的户，这些用户/ 租户被称为居民，因此，该机制提供共同居住/ 多租户[24]。虽然多租户是虚拟化最关键的一个目标，它也被认为是一个链中最薄弱的点。它暴露了虚拟化环境巨大风险。因此，它是云计算的巨大威胁。 然而，这种攻击不能直接从外面发生。意即攻击者需要在他的目标所在的特定物理机器被中[25]。 虽然攻击者应该遵循一些步骤，从分配他的受害人到执行任意命令，这应该允许攻击者能够打破隔离控制和攻击他的受害者[26]。

BE 影响：

BE 作用：当VM 跳频时，攻击者可以可以访问邻接VM ；然而，生物数据因被加密并且对于攻击者它是不可读的。然而，如果攻击者可以控制整个虚拟机，获取的生物信息测定装置，对于攻击者生物信息将可读。

1. VM 流动性

虚拟机移动性解决了一个关键的安全问题[24-26]。通过网络将VM 实例从存储地移动到另一个，VM 移动性提供了灵活性。这个安全问题通常是由于云计算网络环境的弱配置。虽然这种灵活性提供强大的物理安全来防止窃取存储，但它会导致其他安全问题，比如说它允许攻击者获得从一台机器传送到另一台的VM 副本。这种缺陷违反了用户数据的机密性和完整性，当数据被存储到云端时会面临风险。但是，降低有VM 移动性导致的风险不是用户的义务[26]。云服务提供商业承担着减少这些风险的的责任。因此，存储在云出用户数据的安全管理应该写入到服务级别协议(SLA)中，这样会明确规定云服务提供商和云用户的义务。

BE 影响：

即使攻击者得到了VM 的副本，攻击者也可以获得加密的生物数据。

1. VM 多样性

虚拟化技术为用户提供了易用性，在它允许用户高效创建多个VM。这种自由暗示着，管理和确保VM 安全是一个巨大的负担因为其多种多样在几秒中部署的系统[26]。这种多样性让VM 安全管理成为一个挑战。然而，一个适当的SLA 可以帮助解决这个问题。然而，一个适当的SLA可以帮助解决这个问题。看着云计算交付的服务模式，我们可以说，虚拟机在这种情况下的安全管理是不仅仅云服务提供商（CSP）的责任，事实上，这也是云用户的责任。例如，在IaaS层，CSP必须确保底层基础架构的安全性和的稳健性，如管理程序，而用户必须正确确保他的虚拟机和提供的服务已正确配置和确保安全，这包括了保持客户机系统打补丁和更新。然而，这并不限于IaaS。 PaaS的服务模式也需要一些安全和维护，但PaaS的是对VM多样性更加具有鲁棒性

BE 影响：

不同的虚拟机可包含不同类型的操作系统。这些操作系统包括不同的弱点，所以它是虚拟机所有者的责任，保持他们的虚拟机以及BE 更新。这种方法可以在有人侵入他们的虚拟机的情况下保护他们的数据。

1. 拒接服务

拒绝服务可以关停可用的服务;也可以阻塞来源的的性能和功能。虚拟化环境是一个共享的资源环境，虚拟机共享相同的CPU，存储器，带宽和磁盘。另外，提供者资源有限，除了这些，一个虚拟机可能会耗尽这些共享物理资源而这将导致拒绝服务。还有，如果虚拟机同时在同一物理机使用共享资源，拒绝服务也会发生。提供者应该清楚VM 可以使用的最大资源。同时也应该配置它的资源用一种合适的方式来避免拒绝服务。拒绝服务时一个严重的事情，它应该具有良好的配置和监测。

BE 影响：

当服务出现故障，对生物识别加密没有影响。生物特征加密仅仅是针对生物识别信息的保密性并且它不是针对数据的可用性。

1. VM 逃离

VMM 被设计用来让VM 共享系统资源用控制的方式。因此，VMM 必须通过防止与低级硬件层的任何直接交互来执行虚拟机和系统资源之间的隔离。在某种程度上一个被攻破的VM 允许攻击者来控制管理程序被称为VM 逃离[24]。运行在一个虚拟机上的程序可以完全绕过虚拟层(管理程序层)，并且访问主机。以此，它升级为root 权限，从虚拟机特权层基本逃离。这个弱点运行攻击者可能控制所有的客户操作系统，从而导致该环境安全框架的完全损坏[11, 28]。VM 逃离是最糟的情况当在VM 与主机间的隔离被完全攻破。

BE影响：

当VM 逃离发生时，攻击者必须控制主机并获得root权限。因此，攻击者可以对所有的资源和生物设备进行访问，所以攻击者可以读到生物信息。当VM 逃离发生时，BE 对保护生物信息没有任何帮助。

1. VM 监测

虚拟化技术的主要特点是在单个主机内彼此隔离的虚拟机。除此之外，主机没有一个合适的配置，VM 可能会访问和监测其他的VM。这意味着提供者失去了系统的保密性。一个VM 可以实施跨VM 边信道攻击来提取受害者的内存信息，ARP 病毒可以打开受害者的VM 到攻击者VM 的通信，这会允许攻击者监测所有受害者VM 的流量。监测VM 是一个很重要的问题，服务提供者必须阻止来保存其系统的保密性。

BE影响：

如果一个攻击者可以发掘到另一个VM的数据，BE 的数据将被加密，这将给攻击者带来其他的安全层和挑战。

1. VM 间的通信

在虚拟机技术中的剪贴板允许主机虚拟机间的通信。这是非常方便的方法来实现主机间的通信。但是，同时病毒也会被容易传输[29]。另外一个例子是，一些服务提供者没有完全应用VM 的隔离，通过允许VM 访问主虚拟机，并且合适的隔离机制应该被应用[30]。在虚拟机环境中，加密是一个保持传输数据安全可信的好方法。

BE影响：

恶意代码可能为攻击者传输所希望的数据，并且允许攻击者成功侵袭其他VM。对于那些攻击者BE 保持了生物数据的加密和不可读。

1. 主机控制VM

主机是所有虚拟机的控制器；主机是管理者，督察者和运行在其上虚拟机的保护者。因此，主机可以监控所有虚拟机的流量，并控制它们。在虚拟化技术上不同的研究表明主机会影响VM[20]。所以主机安全对于保护虚拟机是非常重要的，一个合适的配置也应该被采用，访问控制限制所有主机来保持所有虚拟机的安全性。

BE影响：

将主机中VM 的数据加密是一个很好的做法，因此，即使主机被攻击者控制，他也不能读到加密的数据。但是，如果攻击者获得了root 权限，攻击者将能读到生物信息。

Ⅵ. 结论

虚拟化有一些安全问题，这些问题转移到了云环境中并影响了生物数据的机密性。本文提出了一种生物加密的解决方案。事实上，生物加密为在云环境中的生物数据挺了机密性。因此，云计算中生物加密是被强烈建议推荐的。的确，这种方法为云计算提供了更高层级的安全性。

此外，它克服了云计算中与生物数据机密性相关的安全缺陷。通过为生物数据使用生物加密云计算中的机密性得到了加强。但是，生物加密不能解决云计算中所有安全问题，而是它保护了生物数据免受于大多数的安全问题。直到此时，依然没有云计算中关于生物加密的研究。但是，将来云计算中生物加密将被实现。因此，云计算中生物数据的加密也将被部署和评价。

致 谢

随着毕业设计的完成，我在中国矿业大学大学生活也接近了尾声。在这四年中，我度过了很多难忘的时光，我收获了知识，增长了见识，锻炼了才干，认识了很多优秀的老师和同学。至今仍然清晰记得大一刚来时的兴奋与憧憬，到如今即将毕业更多的则是不舍，在矿大度过的本科四年将是我一生中最为宝贵的回忆。

感谢我的毕业设计指导老师陈岱副教授。陈老师是一位非常负责、知识渊博、和蔼可亲的老师，从我毕设开始选题到系统开发再到最后论文的修改和定稿，都给了我非常大的帮助和具体细则的指导。当我毕设遇到难以解决的问题时，梁老师总是耐心地为我分析问题、解决问题，在这个过程中我获益良多。陈老师在计算机嵌入式领域研究颇多，它渊博的知识和丰富经验让我逐渐了解这一领域，在此我再次对陈老表示最诚挚的谢意！

感谢我的校外毕业设计指导老师孙岩教授。孙老师是一位科研能力强、待人友善、认真负责的老师。孙老师在整个毕业设计期间，为我指明了研究设计的方向，提出了很多建设性的建议，给我指出了论文和系统中存在的问题和可能的创新点，帮我解决了很多困惑，交给我了很多如何在科研上发现、解决问题的思路和方法，让我对云计算这一领域有了更加深入的学习。

感谢我的母校中国矿业大学为我提供了最好的学习实验和生活条件，我在这里学到了丰富的知识，积累了宝贵的经验，在这里我留下了最美好的回忆。同时感谢我毕设所在的学校北京邮电大学，在这里为我提供了一流是实验条件，有实验室优秀的师兄师姐为我答疑解惑。

感谢信科12-4班的全体同学，感谢四年中都有你们的陪伴，感谢你们带来的欢声笑语，感谢大家的互帮互助。

最后，感谢各位老师评阅我的论文，并提出宝贵的意见和建议，在此我向各位老师表示诚挚的感谢！