

浙江工业大学硕士学位论文

云计算数据存储安全的研究

作者姓名：高煜红

指导教师：方路平教授 陈清华教授

浙江工业大学信息工程学院

2014 年 4 月

**Dissertation Submitted to Zhejiang University of Technology  
for the Degree of Master**



**THE RESEARCH OF DATA STORAGE  
SECURITY IN CLOUD COMPUTING**

**Candidate: Yuhong Gao**

**Advisor: Luping Fang   Qinghua Chen**

**College of Information Engineering  
Zhejiang University of Technology  
April 2014**

# 浙江工业大学

## 学位论文原创性声明

本人郑重声明：所提交的学位论文是本人在导师的指导下，独立进行研究工作所取得的研究成果。除文中已经加以标注引用的内容外，本论文不包含其他个人或集体已经发表或撰写过的研究成果，也不含为获得浙江工业大学或其它教育机构的学位证书而使用过的材料。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人承担本声明的法律责任。

作者签名：

高煜红

日期：2014年 5月 26日

## 学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权浙江工业大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于

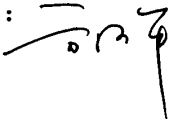
1、保密□，在\_\_\_\_\_年解密后适用本授权书。

2、不保密☒。

（请在以上相应方框内打“√”）

作者签名：高煜红

日期：2014年 5月 26日

导师签名：

日期：2014年 5月 26日

# 云计算数据存储安全的研究

## 摘 要

随着大数据时代的到来和网络存储技术的不断发展,人们越来越多地将自己的数据存放在网络中并进行数据共享。云计算和云存储技术的发展为实现海量数据的存储提供了有效的途径。但是由于技术的不成熟和制度的不完善,云计算数据安全问题一直困扰着人们,制约了云计算的发展。如何设计一个良好的云端存储平台和安全的数据存储与共享模式,如何高效、安全地存储和管理云端的数据成为了企业和各组织、专家们研究的热点。

当前主要流行的云存储平台设计有 Google 的 GFS 和开源 Hadoop 中的 HDFS,这些平台为处理海量数据提供了巨大的帮助,但是在这些平台中还存在一些不完善的安全机制制约了其发展。同时,云存储中最基本的功能之一是数据共享,个人用户或者企业之间想要更安全、更高效地进行数据共享变得越来越频繁,从而找到有效的密钥管理方案显得更为重要。因此,根据实际需求,本文作了相关研究和分析,设计了不同的方案,改进了 HDFS 的性能,以达到行之有效的数据存储安全目标,其主要工作内容和创新点如下:

1. 研究了云计算数据存储安全的关键技术,包括身份认证、数据加密等技术,这些为后面设计的方案提供了理论基础。

2. 利用当前开源云平台 Hadoop,对其分布式文件系统 HDFS 进行分析和优化,完成安全分布式云存储平台的设计和实现。文章设计了一个基于 Hadoop 的安全存储平台(Hadoop-Based Secure Storage Platform, HBSSP),并在整个系统中加入了安全资源池,保证整个系统资源的可扩展性和可靠性。当整个系统设计好后,文章进行了测试和性能分析,表明本文设计的安全云存储平台在实际生活中具有一定的应用价值。

3. 进一步研究了密钥管理机制,提出 KAE 加密算法,该算法通过对密钥的聚合处理,从而有效达到数据间安全灵活地共享和处理。最后通过与其他同类算法比较,证明本文引入的算法有一定的优势。

**关键词:** 云计算, 存储安全, HBSSP, 数据加密, KAE 算法

# THE RESEARCH OF DATA STORAGE SECURITY IN CLOUD COMPUTING

## ABSTRACT

With the development of big data era and network storage technology, people will put more and more data on the online storage and share with others. However, due to the immaturity of technology and imperfect of systems, data security issues have been plaguing people in cloud computing and it restricts the development of cloud computing. Experts focus on how to design a good cloud storage platform and a secure data storage and sharing model how to efficient and secure storage and manage the cloud data in enterprises and organizations.

Currently, the major popular cloud storage platforms are Google's GFS and HDFS, these provide a huge help for us to handle massive data. However, these platforms still have some imperfections security mechanism. Meanwhile, one of the basic cloud storage functions is to provide data sharing. It is becoming increasingly frequent between individual users who want a more secure or more efficient data. Therefore, in order to achieve the effective data storage security objectives, this paper analysis and designs a variety of scheme and finally improves the performance of HDFS. The main contents and innovations are as follows:

1. This paper studies the key technologies of cloud computing data storage security, including authentication, data encryption and other technologies and these technologies provide a theoretical scheme basis for the paper.
2. This paper uses the current open source cloud platform Hadoop, designs a Hadoop-Based secure storage platform, and also the system is added to a secure resource pool ensure that the entire system resource scalability and reliability. Then the paper analyzes and evaluates the performance and integrity of the entire system, it is proved that the whole system has a certain value in real life.
3. Furthermore, the paper introduced KAE encryption algorithms to ensure the security of shared data. It is compared with other similar algorithms and proved that this algorithm has some advantages.

**Key Words:** cloud computing, storage security, HBSSP, data encryption, KAE algorithm

## 目 录

摘要.....	i
第 1 章 绪论.....	1
1.1 课题研究的背景及意义.....	1
1.2 国内外研究现状.....	2
1.2.1 亚马逊 S3.....	2
1.2.2 Google 云存储.....	3
1.2.3 Dropbox.....	3
1.2.4 IBM 云存储.....	3
1.2.5 国内三大运营商.....	4
1.3 本文研究的主要内容.....	5
1.4 论文的结构.....	5
第 2 章 云计算数据存储安全关键技术的研究.....	7
2.1 身份认证.....	7
2.1.1 口令核对法.....	7
2.1.2 基于智能 IC 卡的身份认证.....	8
2.1.3 Kerberos 身份认证.....	8
2.1.4 PKI 身份认证.....	9
2.2 数据加密.....	9
2.2.1 对称加密.....	10
2.2.2 非对称加密.....	12
2.3 数据备份和完整性.....	14
2.4 密钥管理.....	15
2.5 本章小结.....	16
第 3 章 基于 Hadoop 的安全存储系统的研究和设计.....	17
3.1 Hadoop 云存储平台.....	17
3.1.1 Hadoop 框架结构.....	17
3.1.2 Hadoop 安全性分析.....	18
3.2 HBSSP 的分析与设计.....	20
3.2.1 需求分析与系统总体框架.....	20
3.2.2 基于 PKI 身份认证技术的 HDFS.....	21

3.2.3 HBSSP 中文件加密设计 .....	24
3.3 HBSSP 的资源池化 .....	28
3.3.1 安全云服务资源池化 .....	28
3.3.3 ZFS 与 HDFS 集成设计 .....	28
3.4 本章小结 .....	31
<b>第 4 章 云存储平台实验和性能分析.....</b>	<b>32</b>
4.1 测试环境 .....	32
4.2 测试结果与性能分析.....	32
4.3 本章小结 .....	36
<b>第 5 章 基于 KAE 算法的密钥管理研究.....</b>	<b>37</b>
5.1 同类算法分析 .....	37
5.1.1 分层加解密算法 .....	37
5.1.2 基于对称密钥的紧凑密钥算法.....	39
5.1.3 基于身份的紧凑密钥算法.....	39
5.2 KAE 算法设计 .....	40
5.2.1 问题提出 .....	40
5.2.2 算法设计及应用 .....	41
5.3 KAE 算法分析 .....	46
5.4 本章小结 .....	49
<b>第 6 章 全文总结与展望.....</b>	<b>50</b>
6.1 全文总结 .....	50
6.2 研究展望 .....	50
<b>参考文献.....</b>	<b>52</b>
<b>致谢.....</b>	<b>55</b>
<b>攻读学位期间参加的科研项目和成果.....</b>	<b>56</b>

# 第1章 绪 论

## 1.1 课题研究的背景及意义

随着因特网的使用和数据传输的日益发展,数据开始呈现爆炸式增长,人们对数据的依赖性也越来越强。大数据时代的到来和现阶段网络存储技术的不断发展,人们会越来越多地将自己的数据放在网上存储和共享,数据作为企业和个人资产的重要组成部分,其重要性和核心性不断凸显,在人们的生活中扮演着举足轻重的作用。云计算的出现给信息技术带来了巨大的转变。它为无数企业级用户提供高性能的同时,也正在其低成本、快速部署、灵活调整规模等方面发挥着重大优势<sup>[1]</sup>。云计算与大数据的爆发式增长夯实了云存储的基础性地位。IDC 中国企业级系统研究部周震刚认为,预计 2013 年,存储占 IT 云服务比例将由 2009 年的 9% 上升为 14%,到 2014 年,销售出的磁盘中将有 50% 的存储空间出现在公有云中,存储即服务成为云服务的首要应用<sup>[2]</sup>。

尽管数据在不断地增长,但是如何有效地保存数据变为一个企业快速发展的首要问题,目前,很多企业将数据存储作为单独的一个项目来进行管理,企业通过云计算提供的云存储服务,实现网络中整个计算资源的整合和共享。与云计算类似,云存储是指通过集群应用、网格技术或分布式文件系统等功能,将网络中大量不同类型的存储设备通过虚拟化软件集合起来协同工作,共同对外提供数据存储和业务访问功能<sup>[3]</sup>,是对虚拟化存储资源的管理和使用。

与传统的数据存储技术相比,云存储有着多方面的优点。首先,它拥有强大的计算能力和存储能力,在庞大的计算机集群中根据自己的需求请求计算能力和存储容量,如同日常生活中的煤电水一样,这是传统的方式难以做到的;其次,在云计算服务模式,分布式存储系统在体系结构、系统规模、性能以及可用性等方面经历了较大变化,满足更稳定、更可靠的需求<sup>[4]</sup>;最后,通过云计算服务,越来越多的业务会向云端迁移,可以降低企业成本,节约资源。表 1-1 描述了云计算和传统计算的对比。

尽管很多研究机构认为云计算提供了的可靠安全的数据存储中心,但是安全问题仍然是云存储中一个重要的问题。从用户角度来说,数据都保存在云存储供应商,数据的可用性和安全性成为云存储系统的主要问题之一。所以随着云计算和云存储的流行,研究如何安全可靠地保存和传输云端的数据势在必行,这对信息化安全起到突出作用。



表 1-1 云计算与传统计算的对比

对比类型	云计算	传统计算
开发模式	只需购买云中心服务	自行开发系统，自行采购
付费模式	按需付费，使用自如	支付设备及劳动力费用
用户模式	多租户，有弹性	单一用户
计算能力	超大规模	特定大小规模
可扩展性	动态伸缩，易于扩展	固定模式，不可扩展
运行成本	低廉	费用高，维护成本大

## 1.2 国内外研究现状

虽然云计算和云存储都处于发展初期，但是安全问题关系到云计算的发展前景，所以目前来看，国内外对云计算数据存储安全的研究都颇有兴趣，已处于飞速发展阶段，许多安全问题已被提出并给出相应的解决方案。

### 1.2.1 亚马逊 S3

著名的云计算服务提供商之一就是亚马逊，比较典型的云计算应用主要包括 S3(Amazon Simple Storage Service)和 EC2(Amazon Elastic Compute Cloud)<sup>[5]</sup>。S3 提供数据存储和通过 REST, SOAP 和 BitTorrent 等 Web 服务接口来检索数据。S3 是一个键/值(key/value)存储<sup>[6]</sup>。Amazon S3 适合存储大文件，最大可达 5T 的数据。对小型数据的存储，更适合用 amazon 的其他数据存储服务，叫 SimpleDB。

Amazon S3 提供安全机制，用户可以控制谁可以访问它存储的数据，怎么样的、什么时候以及哪里的数据可以被访问。为了得到安全性，Amazon S3 提供四种访问控制机制：

1. 身份认证和访问管理策略。这使得在单个的 AWS 账户下创建多个用户，运用这个机制，每个用户能够控制其他用户进入到自己的文件或者桶中。
2. 访问控制列表。这使得用户能对每个文件选择性的授予特殊的权限。
3. 桶策略。这是用来授予或者拒绝桶内的部分或者全部文件。
4. 字符串查询认证是通过 URLs 共享对象文件。

除了这些机制，用户可以通过 HTTPs 协议，使用 SSL 存储/检索数据。Amazon S3 还提供数据加密机制叫服务端加密 (Server Side Encryption)。

### 1.2.2 Google 云存储

除了 Amazon, 许多跨国信息技术行业的公司都在云计算技术为用户提供云存储服务。Google 云存储是开放者在 Google 云端写或者读数据。除了数据存储, 用户还可以直接访问 Google 的网络基础设施, 认证和共享机制。Google 云存储可以用他的 REST API 或者其他 Google 提供的协议接入。

Google 云存储提供高容量和可扩展性。也就是说他支持每个用户存储 TB 级文件和许多桶。同时提供了强大的数据一致性, 这意味着成功上传数据后, 你可以直接访问删除或者得到元数据。对非开发用户, 就是只需获得少量服务的用户, Google 提供其他数据存储, 叫 Google Docs, 它支持存储达到 1GB 的文件。

Google 开源云平台是当前最热门的云计算平台之一。其平台技术架构主要包括四个方面的内容: 文件存储 (Google Distributed FileSystem, GFS)、并行数据处理 (MapReduce)、分布式锁 (Chubby)、结构化数据表 (BigTable)。它支持海量存储, 并具有较高的容错性, 可以控制负载均衡<sup>[7]</sup>。本文研究的 Hadoop 开源云平台中的 HDFS 是根据 GFS 改进得到, 因此, Google 的开源云平台为本文研究开辟了学术的前进道路。

### 1.2.3 Dropbox

Dropbox<sup>[8]</sup>是一个文件托管服务, 它允许用户通过网络存储和共享数据。它利用文件同步来共享用户设备间的文件或者文件夹。它由 MIT 的两位学生成立。现在它拥有超过 50 百万的用户。用户可以得到免费的 2GB 存储空间, 最多可到 1TB 的付费空间。Dropbox 可以提供用户客户端许多操作系统, 如 Microsoft Windows, Mac OS X and Linux, 同时也可以可以在移动设备上, 如 Android, Windows Phone 7, iPhone, iPad, WebOS and BlackBerry。然而, 在本地未安装客户端的时候, 用户也可以通过基于 WEB 客户端访问它的数据。2007 年成立的 Dropbox 目前正以年增长 10 倍的速度快速成长。它主要使用 SSL 文件传输协议, 存储的数据都是在服务端加密, 它不是在用户自己的设备端进行加密实现安全机制<sup>[9][10]</sup>, 因此, Dropbox 的内部员工可以私底下看到用户上传的文件数据。为此, Dropbox 正在努力实现公司的云存储安全计划, 加快云存储安全研究, 为用户提供一个安全又有保障的存储环境。

### 1.2.4 IBM 云存储

2009 年, IBM 公司提出了“企业级智能云存储”的战略计划。它为客户提供存储虚拟化和基于私有云的存储归档技术<sup>[11][12]</sup>。IBM 提供了同城、异地灾备云中心及方案, 提供具

有防入侵能力、防病毒保护及高级别加密功能；拥有 smart cloud storage access，允许任何用户创建账户、按需设置存储容量，并通过云上传文件；并且 IBM Easy Tier 提供实时分析数据的使用情况，自动将数据在不同层级进行迁移，保证热数据的响应速度，并将冷数据放置成本较低的存储空间，同时配合 3% 的 SSD 固态硬盘，性能可提升更多<sup>[2]</sup>。IBM 从现有存储系统到整合与虚拟化，到管理和安全服务，都已经走上了更快、更简单、更安全的应用层次。

### 1.2.5 国内三大运营商

2012 年 3 月，国内首家运营商级的云计算公司诞生---中国电信云计算公司，它是国内最大的云计算服务提供商，集约化统领中国电信全网包括 IDC、CDN 等在内的广义云业务。中国电信结合云计算应用的特点，创建了高品质的云数据机房保障机制，提供多种规格的虚拟主机和存储空间，各个虚拟机之间相互独立，不同客户之间安全隔离，集成虚拟化技术、数据加密和海量云端用户的身份认证与授权等技术手段，构建了一个相对比较完善的安全防御体系。

自 2007 年起，中国移动通信集团开始搭建大云平台（Big Cloud）。随着云计算的快速发展，中国移动一直以来把用户信息和隐私安全放在一定的战略高度上。2012 年 3 月移动推出“彩云”服务，这为用户提供了使用便捷、安全可靠的云备份方案，该服务采用了保险箱二次加密的认证功能，让用户使用得更为放心。

中国联通也不例外，它自主研发了面向企业、个人和政府单位等的云计算服务“沃·云”，目前该服务主要以存储服务为主，目的主要是实现用户个人信息和文件在多台移动设备上的协同功能，以及文件和资料的集中存储和安全保管。

此外还有国内的清华大学研究学者设计了一种基于 SSL 安全链接的解决方案<sup>[13]</sup>，冯登国等学者曾提出相应的安全机制和安全策略在国内有较深的影响<sup>[14]</sup>，道里云公司提供了安全虚拟监控系统保护数据安全性的解决方案。

以上是国内外的云存储服务提供商在云计算数据存储安全方面的工作，当前，很多云存储服务商都已经采用分布式存储平台来存储企业的海量数据，尤其当前开源的云存储平台 Hadoop<sup>[15]</sup>的出现，给企业带来了巨大的便利，为企业提供海量存储环境提供了基础。但是如何让用户对存储数据更加信任，这就需要进行进一步的探讨。基于 Hadoop 这一平台，如何为企业、用户设计出一个更可靠、更安全的数据存储平台意义重大，在后续的章节中本文将作出详细地分析和研究。

### 1.3 本文研究的主要内容

随着计算机技术的快速发展,在互联网时代背景下云计算应运而生。于是,越来越多的数据和信息需要存储和共享在因特网上,由此带来了云存储的概念。当然,在带来数据共享方便的同时,也带来了众多云计算数据存储的安全隐患。在网络存储环境中,数据可能不存储在自己直接能控制的服务器上,很多时候都将数据存储到分布式云存储环境下,这势必导致了用户对云存储环境的不信任。为使用户在当前云计算环境下数据的安全存储有一定的保证,本文通过相关研究,保证数据或者文件在云计算存储环境下的机密性和完整性,提供端到端的安全性保障。论文的研究内容主要包括:

1. 通过对当前云计算环境下数据存储安全的关键技术研究和分析,采用一种有效的加密算法来完成对数据的加密和数字签名,从而保证用户数据的私密性;
2. 利用当前开源云平台Hadoop,对HDFS进行分析和优化,设计一个基于Hadoop的安全存储平台(Hadoop-Based Secure Storage Platform, HBSSP),并在该系统中加入了安全资源池,保证整个系统资源的可扩展性和可靠性,通过ZFS和HDFS的集成设计,保证整个系统的高可靠性和高读写性能;
3. 对整个Hadoop云存储平台的性能和完整性进行分析和评估,对平台中无加密功能、加入AES加密和加入DES加密三种情况下的文件读写性能进行了测试,证明在保证存储安全的情况下,AES加密更适合于本系统,并且测试了系统加入ZFS资源池后的读写性能,验证了其有效性;
4. 当前个人或者企业都将自己的数据存储于云端,文章在上述提出的安全云存储平台下,对数据安全共享机制下的密钥管理进行进一步的探索研究。引入基于KAE加密算法的密钥管理研究方案,通过对其他三种算法的分析和比较,从而体现本文引入的算法较其他三种算法有一定的优势,从而保证用户间数据的安全存储和共享。

### 1.4 论文的结构

本文采用如下结构安排:

第一章:绪论部分。本章主要说明本文的研究背景及意义,同时又阐述了当前国内外对云计算数据存储安全方面的研究现状,接着对本文的主要研究内容作了一个简单的说明介绍,最后对本文的结构安排进行了说明。

第二章:云计算数据存储安全关键技术的研究。本章主要指出当前云计算环境下的一些存储安全关键技术,从身份认证、访问控制、数据加密和虚拟化安全等方面进行了详细

的研究说明。

第三章：分布式存储安全系统的研究和设计。本章通过对当前主流的开源分布式云平台 Hadoop 的分析和研究，主要针对当前流行的分布式存储平台 Hadoop 的安全机制不够完善和安全管理复杂，对 Hadoop 存在的安全问题进行分析和优化设计，提出一个基于 Hadoop 的安全存储平台，从而为更完善更安全的存储系统奠定了基础。

第四章：云存储平台实验和性能分析。本章主要搭建实验方案 Hadoop 分布式云存储环境并进行性能测试，分析实验结果。

第五章：基于 KAE 算法的密钥管理研究。本章是在上述提出的安全云存储平台下，对数据安全共享机制下的密钥管理进行进一步的探索研究。文章对当前主流的三种算法的分析和研究比较，引入基于 KAE 加密算法的密钥管理方案，将 KAE 算法和其他同类算法进行对比，从而体现本文引入的算法较其他三种算法有一定的优势，从而能有效地保证用户间数据的安全存储和文件间安全共享。

第六章：全文总结及展望。本章主要对论文进行总结，并提出云计算环境下和分布式云存储平台有待解决的安全问题。

## 第2章 云计算数据存储安全关键技术的研究

近年来,云计算发展迅速,它被认为是下一代网络技术的核心架构。然而,云计算在快速发展的同时存在着众多的挑战,其中首当其冲的是云计算安全问题。而在云计算安全问题中存在的最大障碍之一就是数据存储安全问题。近年来,云计算安全相关的研究成果主要集中表现在数据加密、身份认证、访问控制策略、数据灾备等方面<sup>[16]</sup>。云计算安全问题是关系到整个云计算产业发展的关键所在。下面本章通过身份认证、数据加密、数据备份和完整性、密钥管理几个方面对云计算数据存储安全问题作详细分析。

### 2.1 身份认证

认证技术是确保网络存储系统安全的首要门户。认证技术包括身份认证和消息认证两种。在进行某项未经授权操作(如读取文件)前,需进行身份认证,它是在确认网络操作者身份的过程中产生的有效解决方法。它能验证数据的提供者和使用者的身份。认证可以指相互认证,也就是说数据的提供者和使用者的身份之间互相进行认证,建立一种可信关系。认证通常采用动态口令、智能 IC 卡、静态/动态密码验证、数字签名或者消息认证码(Message Authentication Code, MAC)<sup>[17]</sup>等技术。下面简单地描述几种常用的身份认证技术。

#### 2.1.1 口令核对法

在身份认证技术中,常常使用的一种认证方法就是口令核对法。该方法采用如下过程:

1. 系统为每一个合法用户创建一个<用户名, 口令>;
2. 针对需要进行身份认证的用户, 将对其提示输入用户名和相应的口令;
3. 系统检查用户输入的<用户名, 口令>与存在系统内部的<用户名, 口令>进行校验, 如果校验结果一致, 则认为是合法用户; 不一致则认为是非法用户。

若用户的输入的口令较短, 则容易遭到口令的猜测和攻击, 甚至遭到破坏和篡改; 此外, 根据网络环境中明文传输密码口令, 攻击者可以截获网络通道上的密码口令, 使得这种身份认证方案变得极其不安全。一种改进的办法是将口令进行加密后再进行传输, 这可以在一定程度上防止网络的窃听, 但攻击者仍有可能对系统采取离线方式的口令进行猜测

攻击。

### 2.1.2 基于智能 IC 卡的身份认证

基于智能 IC 卡的用户身份认证主要是通过验证用户是否已经拥有物理媒体。其具体过程如下：

1. 将用户合法信息<用户名，口令>存储到智能 IC 卡中，身份认证服务器中存入某个事先由用户选择的随机数。
2. 当用户需要进行身份认证时，输入<用户名，口令>，系统以此判断智能卡是否合法。
3. 由智能卡自身身份的合法性进行验证，如果是合法用户，则在智能卡中的随机数将被自动发送到认证服务器进行进一步的认证；

这种方法是基于智能卡的物理安全特性，不容易被非法人员恶意改造，并且不能直接读取其中的数据<sup>[18]</sup>。即使智能卡被盗窃了，非法分子仍旧无法知道<用户名，口令>，这样就大大的增强了系统的安全性。

### 2.1.3 Kerberos 身份认证

Kerberos 认证<sup>[19]</sup>是基于 TCP/IP 设计的第三方可信认证协议，它提供了较高安全的身份认证和资源访问机制。该认证机制中除了认证服务器之外，还有一个授权服务器。认证服务器中保存了所有的用户口令，其认证过程具体可按照如下步骤：

1. 认证服务器从用户口令中产生一个密钥 K，并传递给用户一个可以访问授权服务器的票证(Ticket1，记为 T1)；
2. 用户将获得的 T1 连同其个人账户信息发送到授权服务器；
3. 授权服务器对用户身份信息认证后，若该用户是合法用户，则发送给用户一个新的票证(Ticket2，记为 T2)。T2 是用户访问某个系统 S 的合法凭证；
4. 用户将获得的 T2 连同个人账户信息发送给系统 S；
5. 系统 S 对信息认证后给用户提供相应的服务。

一个票证具有有效期限，用户可以使用多次票证直到过期为止。在以上的过程中，Kerberos 还为用户和服务器提供证明双方安全通信的会话密钥。

#### 2.1.4 PKI 身份认证

PKI(Public Key Infrastructure)即为公钥基础设施安全体系,它是利用一对互相匹配的密钥进行加密、解密<sup>[20]</sup>。其基本原理是:通过一个密钥进行加密的信息只能由与之相匹配的另一个密钥才能进行解密。公钥可以广泛地发给自己有关的通信方,而私钥则必须十分安全可靠地保存在一个服务器上。

PKI 为人们提供了一整套安全的机制,其典型系统中主要包括认证机构(Certificate Authority, CA)、证书和证书库、密钥备份和恢复系统、证书撤销和密钥更新机制、PKI 应用接口等基本组成部分<sup>[21]</sup>。其中,CA 是核心构件,在 PKI 系统中起着至关重要的作用。一个有效的 PKI 系统必须要保证系统的安全性和透明性,即用户在获得加密或者解密的服务时,并不需要详细了解整个 PKI 系统内部是怎样管理数字证书和密钥的。

除了上述详细说明了四种身份认证机制外,在实际应用中还有类似一次性口令机制、发送动态短信密码等方式进行用户的身份认证,此外还有基于生物特征的身份认证<sup>[22]</sup>给云计算带来了新的期望。

## 2.2 数据加密

日常生活中,人们与互联网之间进行数据的传递过程中涉及到许多私人信息,其个人信息的安全有着不可替代的重要性。数据加密技术<sup>[23]</sup>是保障数据安全的一个重要手段。

当一个数据/消息传送到目的地时,很有可能被入侵者 I(Intruder)介入,从而执行以下四种行为:

1. 它能阻止消息,从而数据不能到达目的地,这样就破坏了数据的可用性;
2. 它能窃听/拦截数据,这样就不能保证保密性;
3. 它能改变消息的内容,这样数据的完整性就被破坏了;
4. 它能伪造数据,然后再送到目的地,这样同样破坏了数据的完整性。

通信过程中两方想要进行安全消息的传递时候,常常会发生上述的四种危险情况。在密码学中,加密技术可以有效地处理这些安全问题。实际上数据加密是保证通信过程中信息安全的最重要途径。加密学中常常使用数据的加密解密,也被称作编解码<sup>[24]</sup>。加密过程或者说编码过程其实是对一种原始文本进行了改变,通常该原始文本称为明文。将明文变换成难以理解的字符串后通常称为密文。为了改变它的密文,将其还原成明文,必须经过解密,或者说是解码,其整个过程可以表示如图 2-1 所示。



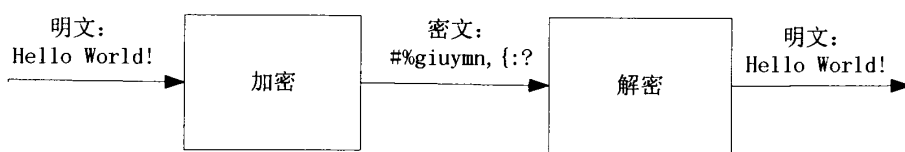


图 2-1 加解密过程

从图 2-1 可以知道，明文“Hello World!”可以看作一串序列，包括<H,e,l,l,o, ,W,o,r,l,d,!>这些字符，经过加密处理后产生了对应的一串乱码序列，最后经过解密还原成最先的序列。为此将包括数据加解密的系统称为加密系统。根据加解密的密钥是否相同分类，可以分为对称加密算法和非对称加密算法<sup>[25]</sup>。

### 2.2.1 对称加密

在加密系统中，密钥 K 常常运用在加密和解密数据过程中。如果在加解密过程中使用同样的密钥 K，那么称该过程为对称加密，密钥称为对称密钥（或者共享密钥），其具体过程如图 2-2 所示。

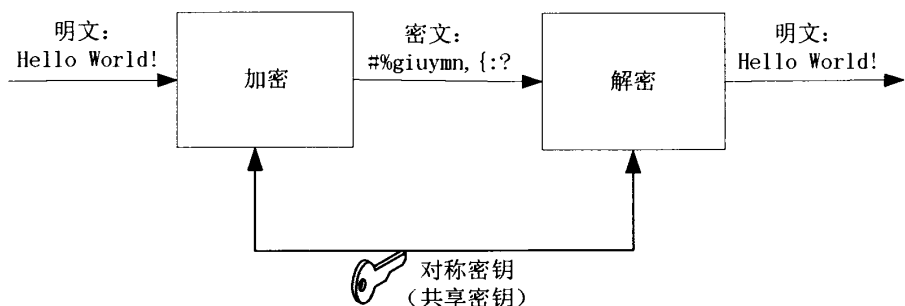


图 2-2 对称加密过程

在这样的情况下，加密和解密算法称为对称加密算法<sup>[26]</sup>。对称加密算法在很早之前就得到广泛应用，所以技术已经成熟。在对称加密算法中，数据的加密和解密可以看作是一个逆转过程，简单的可以记为如下操作：

$$C = E(K, P) \quad (2-1)$$

$$P = D(K, C) \quad (2-2)$$

$$P = D(K, E(K, P)) \quad (2-3)$$

其中，C 为密文，P 为明文，E 和 D 分别为加密和解密算法，K 为对称密钥（共享密钥）。

在计算机系统中，对称加密算法是一种比较经典的算法。对称加密算法的优缺点如表 2-1 所示。

表 2-1 对称加密算法优缺点

优点	缺点
加解密速度高	密钥传递和管理比较困难
计算开销小	相同密钥，安全性得不到保证
长密钥，保密性较高	缺乏签名功能

其在学术界比较有名的对称加密算法有 DES<sup>[27]</sup> (Data Encryption Standard,数据加密标准)算法,DES 的变形算法,如三重 DES(Triple DES)<sup>[28]</sup>,AES(Advanced Encryption Standard,高级加密标准)<sup>[29]</sup>和欧洲的 IDEA<sup>[30]</sup>, Blowfish<sup>[31]</sup>等, 其中 AES 加解密算法的原理流程如图 2-3 所示。

AES 是迭代、对称分组密钥，它可以使用 128、192、256 位密钥，以 128 位长度为例，即算法输入 128 位数据，密钥长度也是 128 位。对称加密算法使用相同的密钥进行加密和解密数据。通过分组密码返回的加密数据的位数与输入数据相同。迭代加密采用一个循环结构，在该循环中使用重复置换（permutations）和替换(substitutions)两种方法输入数据<sup>[29]</sup>。

AES 加密数据块大小最大是 256bit，但是密钥大小在理论上没有上限。AES 加密有很多轮的重复和变换。大致步骤如下：1、密钥扩展（KeyExpansion），2、初始轮（Initial Round），3、重复轮（Rounds），每一轮又包括：SubBytes、ShiftRows、MixColumns、AddRoundKey，4、最终轮（Final Round），最终轮没有 MixColumns。

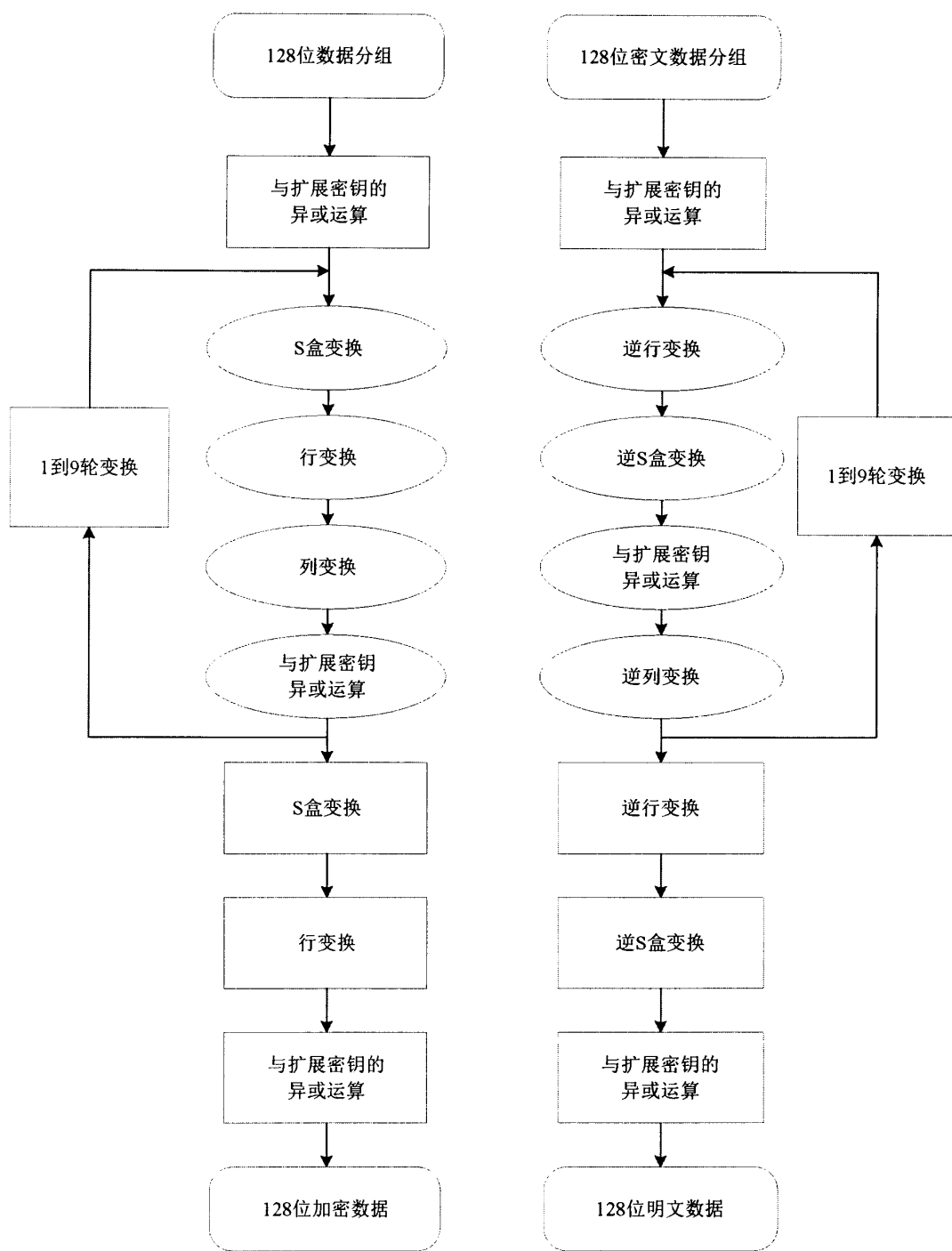


图 2-3 AES 加密解密流程

### 2.2.2 非对称加密

针对对称加密的缺陷，专家们设计了非对称加密算法。在加密系统中，如果加密过程中的密钥和解密过程中的密钥是不相同的，那么可以称该过程为非对称加密<sup>[32]</sup>。非对称

加密过程如图 2-4 所示。在非对称加密过程中有两种密钥，分别称为加密密钥（或者常称为 **private key**，私有密钥）和解密密钥（或者通常称为 **public key**，公开密钥），简称为公/私密钥对。可以简单的记为如下操作：

$$C = E(K_E, P) \quad (2-4)$$

$$P = D(K_D, C) \quad (2-5)$$

$$P = D(K_D, E(K_E, P)) \quad (2-6)$$

其中， $K_E$  和  $K_D$  分别为私钥和公钥， $C$  为密文， $P$  为明文， $E$  和  $D$  分别为加密和解密算法。

在非对称加密算法中，加密和解密使用两把不同的密钥，并且要从一个密钥中推导出另外一个密钥的可能性很小。加密密钥向公众公开，故可以简称为公钥，而解密密钥只有解密人自己知道，故可以称为私钥。如果一个人选择公布了他的公钥，其他任何人可以用这一公钥来加密传送给他的消息。私钥是秘密保存的，只有私钥的所有者才能利用私钥对密文进行解密，客观上来说，完成了对消息传送者的身份认证。

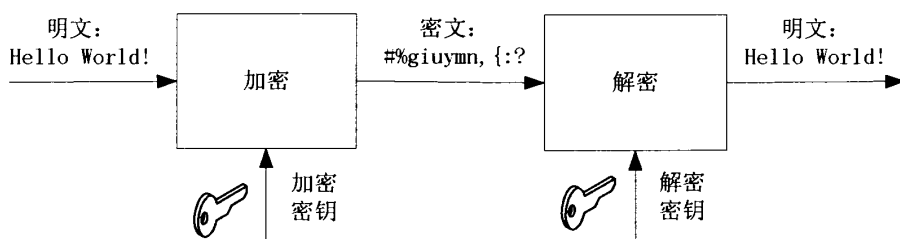


图 2-4 非对称加密过程

非对称加密算法是一种比较复杂的算法。其在学术界比较有名的非对称加密算法有 RSA（Rivest-Shamir-Adleman）<sup>[33]</sup>、DSA（Digital Signature Algorithm）<sup>[34][35]</sup>、零知识证明算法<sup>[36]</sup>和椭圆曲线算法<sup>[37]</sup>等。非对称加密算法的优缺点如表 2-2 所示。

为了充分发挥对称加密和非对称加密算法的各自优点，在实际应用中常常将两者结合起来使用，扬长避短，提高整个系统的安全性，如用 AES 或者 DES 来加密数据信息，利用 RSA 来进行数字签名，并传递对称加密算法中的密钥。

表 2-2 非对称加密算法优缺点

优点	缺点
密钥传递和管理比较简单	加解密速度慢
安全性高	复杂性高

在本文后续的章节中，文章还将多次利用数据加密技术，在系统设计过程中，本文采用 AES 加密和 RSA 加密结合方式，提高了整个系统的机密性和安全性，详细加密和解密设计过程通过文章第三章内容进行阐述。

2.3 数据备份和完整性

2011 年 3 月 11 日，日本发生 9 级大地震，除了令不少人痛失家园外，还有许多企业也被迫停止运作，因为灾难导致了企业的数据丢失，业务中断。为此，在云计算时代下，数据的灾难备份和恢复尤为重要。由于云计算的数据中心存储了与企业业务相关的所有信息，因此，其灾难备份和恢复技术应该支持文件级和系统级别的恢复备份，确保整个企业所有数据的完整性和可用性。

为保护重要的数据，用户需要频繁地对数据进行备份。传统的数据备份一般是采用冷备份，即备份时需要停止系统运行状态，这样就会导致数据在备份期间无法正常访问。但是许多关键性的应用，比如当今流行的淘宝电子商务系统等，这些系统需要连续运作，停机就意味着业务终止，这会对系统造成难以估量的损失。因此，需要一种技术来保证系统在进行备份期间仍然能够正常运行。快照技术(Snapshot)就是在这样的问题背景下应运而生的。

快照(Snapshot)能在不停止应用程序的情况下生成某一瞬间的数据映像(image)，用户可以对数据映像进行数据地保存和备份，当系统出现问题或者数据不慎丢失时，用户可以安全方便地获得快照创建时刻的数据映像<sup>[38]</sup>。

针对数据的备份问题和完整性问题，本文在后台文件系统采用动态文件系统(Zettabyte File System, ZFS)。ZFS 是一种新型的文件系统，它改进了现存技术，从而在根本上为新的数据管理提供了可靠的方法。ZFS 主要有如下特点<sup>[39]</sup>：

- 1. 大容量存储池，易扩展；
- 2. 防止数据损坏；
- 3. 磁盘数据始终保持一致；
- 4. 即时快照(snapshot)；

5. 管理模型简单;
6. 在线数据擦洗;
7. 高度可扩展。

ZFS文件系统是一个跨时代的创新型文件系统,是第一个128位的文件系统。2010年ZFS已经成功移植到Linux平台上,它从根本上改变了文件系统的管理方式。ZFS不是类似于HDFS的分布式文件系统或者Lustre之类的集群文件系统,但通常可以用作HDFS和Lustre的后端存储文件系统使用<sup>[40]</sup>。

ZFS文件系统主要使用存储池的概念来管理存储的物理空间。ZFS存储池的结构如图2-5所示。传统的文件系统采用卷管理的方式,其磁盘管理比较受限制<sup>[41]</sup>。ZFS完全抛开了这个念头,它不再创建虚拟的卷,而是把所有的硬件设备(如硬盘等)集中到一个存储资源池中进行管理<sup>[42]</sup>。这样,文件系统不仅仅只局限在单个的物理设备上,它可以随时向存储池中添加新的容量,所有池中的文件系统能立即使用新增加的空间。同时,系统中充分利用ZFS文件系统带有的快照和备份功能,能加强整个系统中存储文件的安全性。

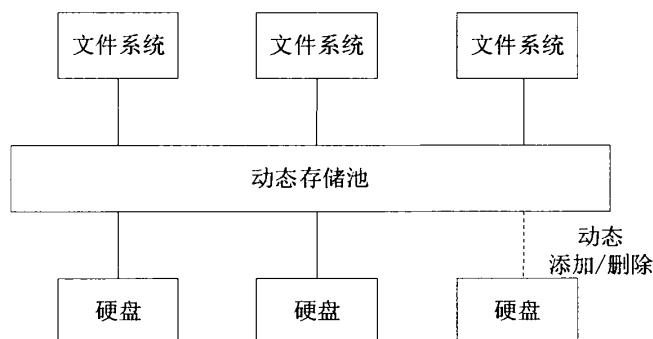


图 2-5 ZFS 存储池结构

虽然 ZFS 有众多的优点,但是现阶段对它的研究和使用的有限,当前其使用用户较少,新技术的关注点还有待提高。鉴于研究能力和时间有限,文章只对其中几个优点进行研究,并应用到实际云存储系统中,详细设计见文章第三章部分。

## 2.4 密钥管理

云计算存储环境下,多租户之间进行文件的共享是最常见的问题。当文件进行加密共享后,相应的文件密钥也必须共享,否则各个用户之间不能达到共享资源的目的,在共享文件带来方便的同时引来了密钥共享和密钥管理难题。

在当前的分布式文件存储系统中，可以设置一个单独的安全密钥管理服务器来完成对密钥的存储和管理，目的是可以提供给用户或者企业一个安全、高效的密钥管理服务，针对密钥管理的具体设计和实施，详细研究可以见第五章。

## **2.5 本章小结**

本章主要指出当前云计算环境下的一些存储安全关键技术，从身份认证、数据加密、数据备份及完整性和密钥共享管理等方面进行了详细的研究说明，这些关键技术为后面整个安全存储系统的设计提供了一定的理论基础。

## 第3章 基于 Hadoop 的安全存储系统的研究和设计

计算机和网络技术的快速发展促成了当前的云计算和云存储,而在云计算和云存储模式下推动了分布式存储技术的进步。文章主要针对当前流行的分布式存储平台 Hadoop 的安全机制不够完善和安全管理复杂,对 Hadoop 存在的安全问题进行分析和优化设计,提出基于 Hadoop 的安全存储平台,从而为更完善更安全的存储系统奠定了扎实的基础。

### 3.1 Hadoop 云存储平台

#### 3.1.1 Hadoop 框架结构

Hadoop 是当前热门的云计算存储平台之一。它作为一个开源的软件平台使得编写和运行用于处理海量数据的应用程序更加容易<sup>[43]</sup>。Hadoop 框架中从功能上来说可以简单地分为海量数据存储和海量数据处理两部分。如图 3-1 就是简单的 Hadoop 框架结构。

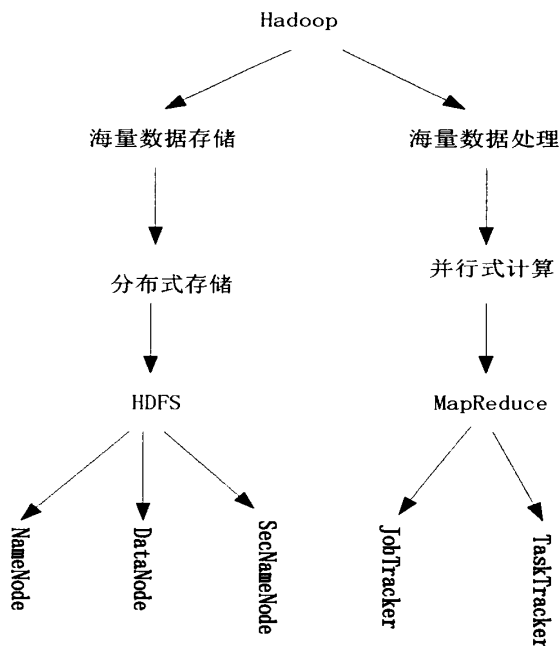


图 3-1 Hadoop 架构

在海量数据存储中,主要采用分布式存储,最核心的设计就是 HDFS 分布式文件系



统<sup>[44]</sup>。它主要参照 GFS(Google File System)<sup>[45]</sup>实现,拥有多机备份、扩展性强且经济廉价等特点,可以在普通的 PC 机上运行,同时存储采用 key-value 形式,适合视频和音频等非结构化数据存储。HDFS 采用 master/slave 架构。一个 HDFS 集群是由一个 Namenode 和多个 Datanodes 组成<sup>[46]</sup>。其中 Namenode 是中心服务器,负责管理文件系统的命名服务和客户端对文件的访问。Datanode 是存储节点,负责管理数据存储。在 HDFS 内部,一个文件被分割为一个或者多个数据数据块(block),这些数据块被存储在一组 Datanodes 上<sup>[47]</sup>。另外在分布式文件系统中还包括一个 Secondary Namenode.它并不是 Namenode 出现问题时的备用节点。它和 Namenode 负责不同的事情,其主要功能是周期性将 Namenode 的元数据信息 FsImage 和 Editlog 合并,以防止日志文件过大,合并过后的 FsImage 会在 Namenode 保存备份,保证 Namenode 失败的时候可以进行恢复<sup>[48]</sup>。

在海量数据处理中,主要采用并行式处理,最核心的设计就是 MapReduce 过程<sup>[49]</sup>。MapReduce 是 Google 提出的一个软件编程框架,用于大规模数据的并行运算。MapReduce 主要分成 Map 和 Reduce 两个阶段,Map 阶段对数据进行分布式处理,Reduce 阶段将 Map 输出的结果进行合并,得到最终结果<sup>[50]</sup>。它同样也包括两个部分,包括 JobTracker 和 TaskTracker。作业 Job 在客户端通过 Jobclient 类将输入集分解成小的数据集(data set),然后通过 RPC 向 JobTracker 提交作业,并将应用程序和配置参数打包存入 HDFS,并把路径转交给 JobTracker。JobTracker 进行分发数据,TaskTracker 进行数据的处理计算<sup>[51]</sup>。

作为一个分布式文件系统平台,Hadoop 具有以下一些优势:

1. 可扩展性。Hadoop 可以可靠地处理 PB 级的数据。
2. 经济性。Hadoop 整个集群可以由成千上万个节点组成,用户可以通过将数据分布到由廉价 PC 机组成的集群中进行处理。
3. 有效性。Hadoop 通过不同节点对数据进行分发,这样数据可以在不同的节点 Datanode 上并行处理,这样就大大地提速了数据的处理过程。
4. 可靠性。Hadoop 可以自动维护一份数据的多个拷贝,当计算任务失败时,它可以自动对失败的任务进行重新部署。

尽管 Hadoop 已经被大家公认为当今处理大数据的较好开源平台,很多企业都已经采用它来进行海量数据的分析和设计,HDFS 文件系统有很强的存储性能,但是在使用过程中,Hadoop 自身还是带有一定的局限性,其安全性能还不够高,为此,加强 HDFS 的安全性能势在必行。

### 3.1.2 Hadoop 安全性分析

Hadoop在设计之初并没有考虑平台的安全问题,任何用户都可以冒充合法用户来访问其HDFS或者MapReduce,在其集群上做出数据篡改、伪造任务等恶意行为<sup>[52]</sup>。随着技术的发展,Hadoop平台开始增强了安全性,从2009年开始,Apache公司专门组织工作人员对Hadoop进行安全认证技术和授权机制的研究,在HDFS中加入了Kerberos 进行身份认证。

HDFS 采用Kerberos 进行身份认证,客户端认证过程如图3-2所示。

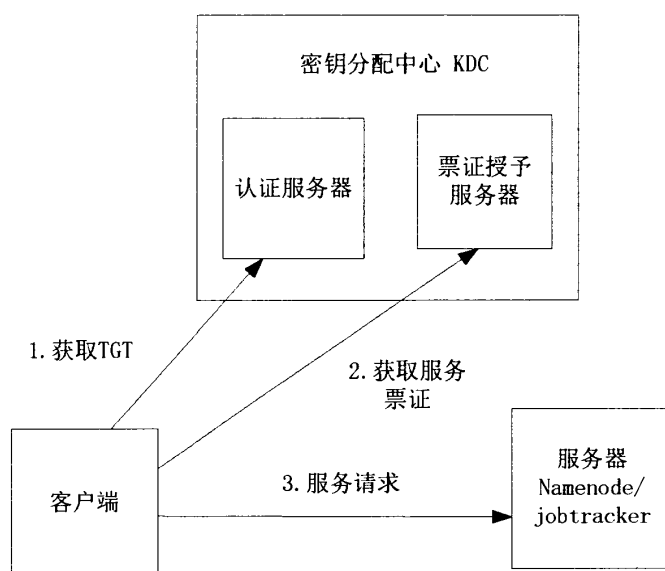


图3-2 客户端Kerberos身份认证

具体过程如下:

1. 首先,客户端Client 向密钥分配中心(Key Distribution Center, KDC) 发送自己的身份信息,向认证服务器发送一条报文,并且获取一个含时间戳的票证授予票证(Ticket-Granting Ticket, TGT)。

2. 客户端client利用TGT向票证授予服务器(Ticket-Granting Server, TGS)发送一条报文,并且获取一个服务票证。

3. 客户端client向服务器发送服务请求,出示服务票证,服务器得到验证后方可证明自己的合法性。在Hadoop应用中,该服务器可以是namenode,也可以是jobtracker,并且可以为客户端提供它所需的服务。

但是,Hadoop采用Kerberos身份认证技术,每次认证都要请求它的密钥分配中心KDC,而Hadoop本身不自带KDC,这就需要用用户额外安装一个KDC,这样就增加了更多的麻烦。

同时,在Hadoop中,一个MapReduce过程中包含成千上万个子任务,每个任务的执行都要使用Kerberos身份认证。这时候在一个短时间内同时向票证授予服务器请求票证服务,就会使得KDC负载急剧增大,从而影响整个系统的性能,而且,在客户端向KDC申请TGT时发送的身份信息很容易被截取而发生篡改。HDFS文件系统的存储和传输过程中并不带有加解密技术,Kerberos身份认证也不能给文件加密带来帮助,因此,文章有必要设计一种更为合理的方案来提高Hadoop的安全性。

## 3.2 HBSSP 的分析与设计

### 3.2.1 需求分析与系统总体框架

由于用户把数据存储云端,意味着用户对数据失去了真正意义上的管理权限,用户的数据存在着被非法访问和泄露的隐患,因此,需要在用户把数据上传之后,且真正存储在云端之前做相关的数据加密处理,以保证数据的完整性、机密性,并保证该方法是可行的。在前文中提到,本文使用的云端存储平台是Hadoop,而当前版本的Hadoop并没有在传输和存储过程中对数据进行加密处理。因此,本文必须对存储的数据进行安全加密处理。下面是详细的问题考虑列表:

1. 数据的保密性,数据加密是有必要的。但是数据是在客户端加密还是云端加密是需要考虑的问题之一,同样数据在从云端取出来后解密的过程也需考虑。
2. 数据的完整性,使用数字签名将是一个明智的选择。签名过程最好在数据加密之后。同样,数据需在云端取下来后在客户端进行认证。
3. 数据的可靠性,在云端应该采取安全可扩展的云存储服务,充分保证用户存储在云端的数据的安全性。
4. 在对称加密算法中,所有的加密密钥都很敏感。因为这些密钥是用来访问数据的,因此如何有效地管理好数据密钥也是本文后面章节的探索和研究重点。

针对上述问题考虑列表中的各个问题,本文设计了一个基于Hadoop的安全存储平台来解决相关问题,如图3-3系统框架所示。通过该系统架构,文章将逐个分析和解决上述问题列表中问题,从而提供一个优化的安全云存储平台。

在2.2.1节和2.2.2节本文提到了两种加密方式,在本文提出的架构中,用户将文件写入到分布式云存储系统中,在写入的过程中使用对称加密算法AES-128bit对数据本身进行加密。在云端,为了隔离不同用户的数据,同时适应访问权限控制的要求,再对元数据使用非对称加密算法RSA进行数字签名,这样使得数据存储在云端之后的机密性和完整性有了

保证。

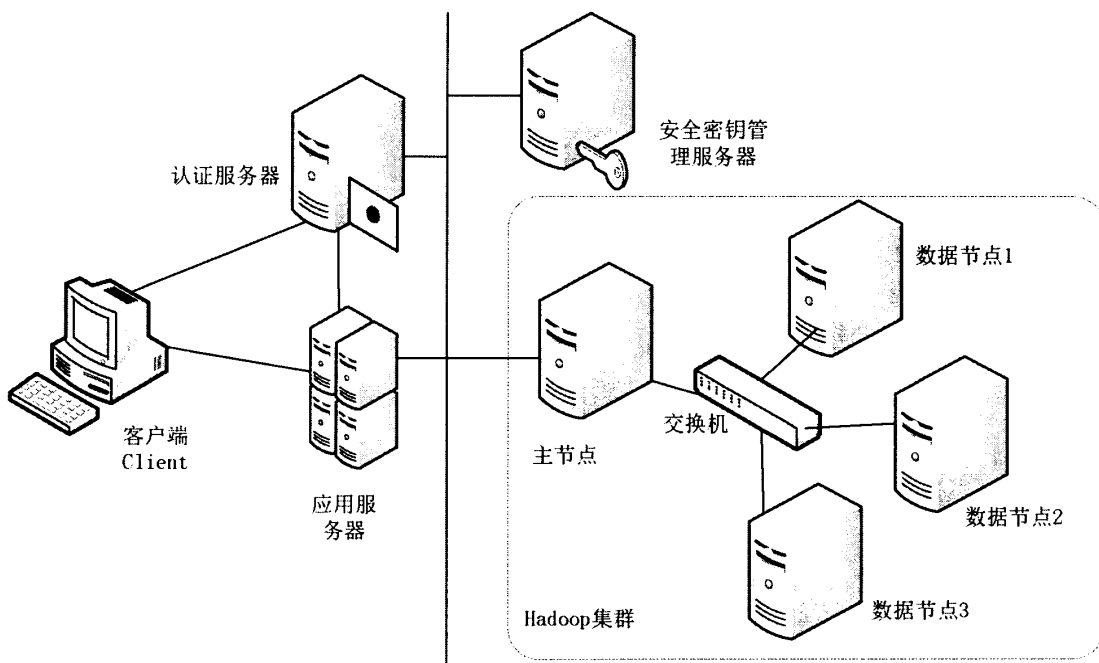


图3-3 HBSP系统架构

HBSP系统主要包括客户端、认证服务器、应用服务器、安全密钥管理服务器和分布式Hadoop集群五个部分，其中客户端主要负责用户文件的保存和持有；应用服务器作为访问存储系统的代理来访问文件；认证服务器完成用户在访问文件时候进行身份认证；安全密钥管理服务器用于管理文件密钥，并提供安全可靠的密钥服务，包括密钥的存放、生成、更新等操作；分布式文件系统主要负责加密文件的存储及HDFS文件系统的可靠性和完整性保证。整个分布式存储安全系统的具体模块分析将在接下来的叙述中做详细说明，其中关于安全文件共享和密钥访问控制管理将在第五章进行展开说明。

### 3.2.2 基于PKI身份认证技术的HDFS

在3.1.2中，本文分析了Hadoop在安全性上存在的问题，由于KDC的瓶颈，很难实现提供安全性和效率性的共存。因此本文放弃了Kerberos的机制，采用了基于公钥基础设施PKI的HDFS身份认证机制。认证详细的过程如图3-4所示。

具体步骤按照如下过程执行：

1. 用户通过客户端client访问Hadoop集群；

2. 该集群向CA认证中心发出用户身份认证请求；
3. CA认证中心接收到认证请求后，用户可以访问认证中心，通过自己的私钥完成数字签名进行身份验证；
4. CA认证中心完成身份验证后，返回给客户端以示证明身份合法并附加自己的数字签名；
5. 用户通过已验证的身份证明，可以访问Hadoop集群。

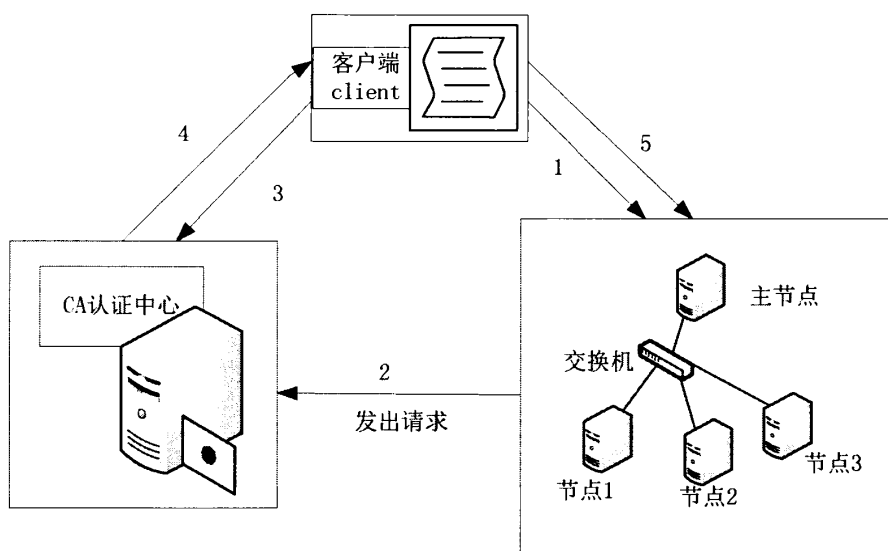


图3-4 基于PKI的HDFS身份认证机制

在上述的五步中，有一个关键点就是数字签名，为了保证在传输密钥时信息不会被篡改，文章需要设置一个HDFS的数字签名机制，具体过程如图3-5所示。

1. HDFS客户端向CA认证中心发出认证请求服务；
2. 当CA认证收到请求后，建立连接，然后产生时间戳和一个大的随机数；
3. CA认证中心向客户端发送时间戳和那个大的随机数；
4. 当客户端接收到时间戳和随机数后，采用RSA进行私钥签名，产生密文；
5. 当已签名的密文产生后，客户端将其签名数据发送给CA认证中心；
6. CA认证中心通过用户对应的公钥对签名数据进行身份认证，若验证通过，则其向客户端发送身份证明和CA认证中心数字签名；如果验证不通过，则返回认证失败消息。

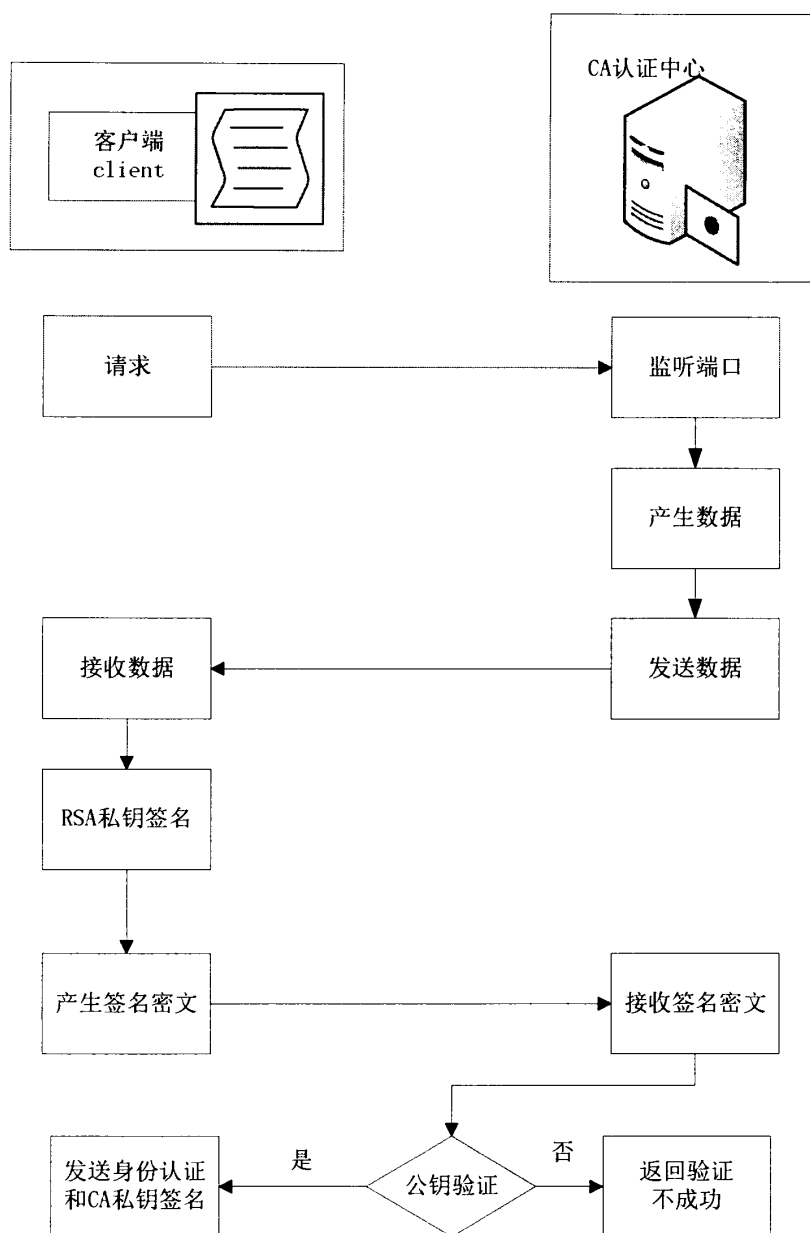


图3-5 HDFS数字签名过程

为防止用户伪造身份，CA认证中心需要对用户的身份证明用私钥进行签名，并返回给客户端，这样可以保证用户向分布式文件系统发送来的是真实身份，并非伪造信息。

对比Kerberos认证方案，基于PKI的身份认证应用于HDFS时，节点一旦获得CA颁发的数字证书，在随后向服务器申请身份认证的过程将不会有PKI管理实体的参与，这就有效地避免了在业务量较大时候的系统瓶颈，而且，密钥传输时的数字签名机制也保证了在身份认证时用户信息不会被篡改，进一步加强了认证时的可靠性。

### 3.2.3 HBSSP 中文件加密设计

任何用户将自己的文件传到云端存储系统后都希望有较高的保密性和安全性。然而，HDFS在设计之初没有考虑到数据存储的安全加密问题，因此，它在设计之初并没有以加密的方式存储文件。原始的HDFS中文件的写入过程如图3-6所示：

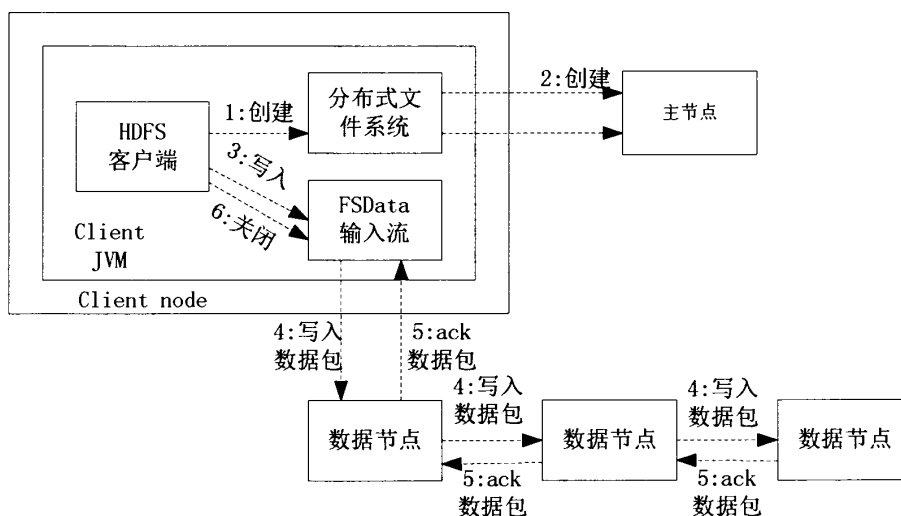


图3-6 HDFS文件写入

HDFS文件的写入过程是这样的：

1. 在文件的写入过程中，HDFS客户端client通过对DistributedFilesystem这个对象并调用create()函数来创建一个文件（如图中3-6中第一步1:create）；

2. DistributedFilesystem对namenode创建一个RPC调用，在文件系统的命名空间中创建一个新的文件，此时该文件还没有相应的数据块（如图中3-6中第二步2:create）；

3. namenode执行各种不同的检查以确保这个文件不存在，并且客户端有创建该文件的权限，如果通过所有的检查，namenode就会为创建新文件记录一条新的记录，否则就会在客户端抛出一个异常操作；

4. 由此创建好记录后，DistributedFilesystem给HDFS客户端返回一个FSDataOutputStream对象，由此客户端才真正开始写入数据，数据由FSDataOutputStream分成一个个的数据包（data packet），并写入内部队列，称为数据队列（Data Queue, DQ）。写入数据的过程（如图中3-6中第三步3:write）；

5. `FSDaOutputStream`分成一个个的数据包（data packet）通过流式传输到第一个数据节点`datanode1`，并由该数据节点存储数据包，类似操作，第二个数据节点、第三个数据节点都存储该数据（如图中3-6中第四步4:write packet）；

6. 数据节点`datanode`返回给`FSDaOutputStream`一个确认值，称为“确认队列”（Ack Queue, AQ），直到收到确认消息后才会将该数据包从队列中删除（如图中3-6中第五步5:ack packet）；

7. 直到整个写入过程完成后，客户端会调用一个`close()`方法，结束数据包的写入（如图中3-6中第六步6:close）。

以上是HDFS对文件的写入过程，然而在该过程中并没有使用加密操作，这样写入的数据安全性和完整性得不到保障，为此，本文在原先的HDFS文件系统写入过程中引入加密操作，HBSSP的写入操作可用如图3-7表示：

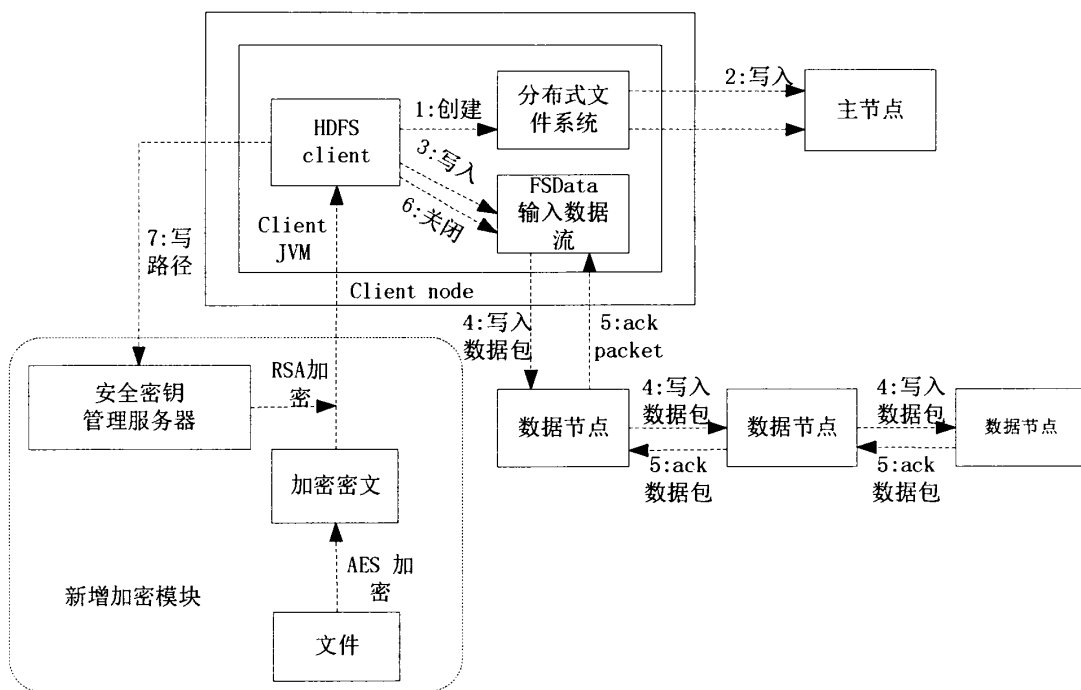


图3-7 HBSSP文件写入

在图3-7中，对比原始的HDFS操作引入了Secure key management server，即整体框架图3-3中的安全密钥管理服务器。具体修改操作是在文件写入之前，首先对文件进行128位的AES加密，接着由安全密钥管理服务器中生成一个随机密钥，该密钥可以对已经加密密文



密钥再进行RSA加密，最后将文件的密文写入HDFS中，并返回密文地址写入安全密钥管理服务器的MySQL中。

在上述过程中，关键需要注意两点：

### 1. 文件AES加密过程

本文文件的AES加密过程是采用Hadoop的分布式计算模型MapReduce进行加密，具体的操作过程如图3-8所示。

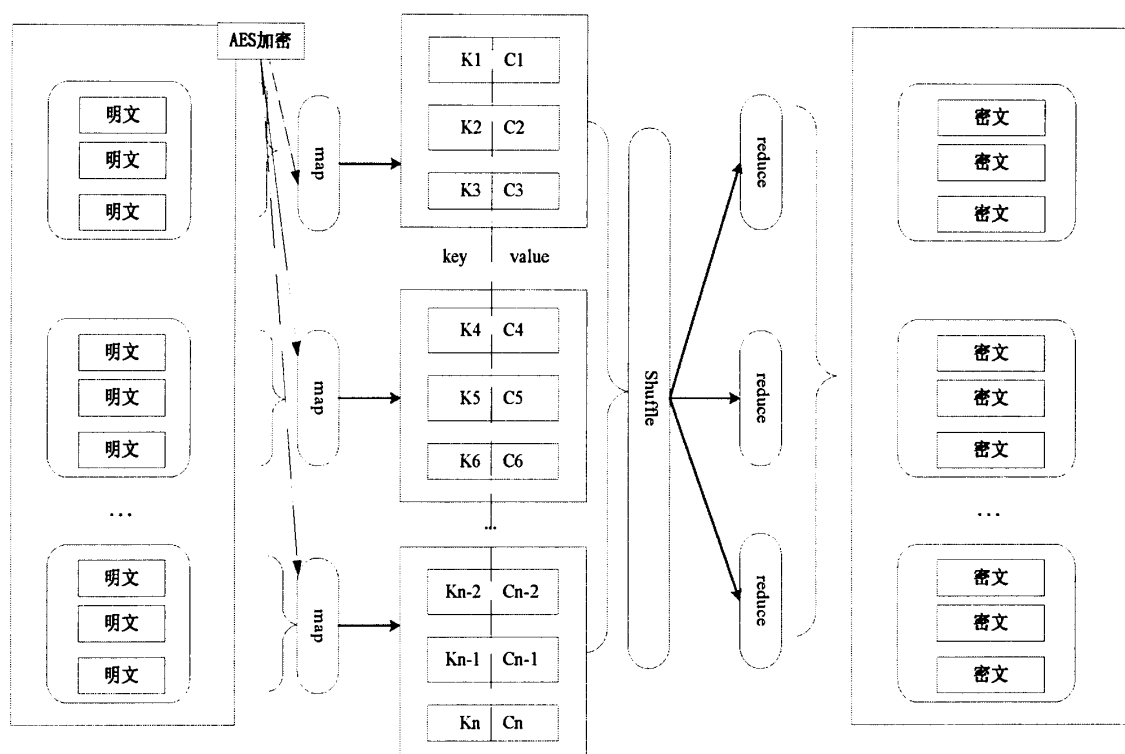


图3-8 AES加密的MapReduce过程

在图3-8中主要分为三个阶段：Map、Shuffle和Reduce。

在Map阶段，首先需要将明文文件按照块(默认64MB)进行Map任务的划分，然后针对同一块中的明文数据按照AES算法的块大小(AES为64位)生成<key, value>值对。其中key是明文相对于块文件的偏移量，value值是实际的64位长的明文信息。然后对每个<key, value>根据用户提供的密钥AES加密算法，将明文块  $p_i$  加密为对应的密文块  $c_i$ 。根据输入的key值和块号生成一个表征密文位置信息的值  $K_i$ ， $K_i$ 的内部结构为(*group, offset*)。那么每个

Map的输出为 $(K_i, c_i)$ 。

由于MapReduce本身的特性，在执行Map任务时是并行的，这也意味着明文文件的加密的过程中，无法控制Map任务的执行顺序，加密后的密文顺序会被打乱。因此，文章采用Shuffle的过程来保证原来的内容顺序。Shuffle阶段将Map阶段的输出进行分组，根据 $K_i$ 中的group值将相同的分在同一组中，并根据offset的值进行排序，最终将统一组的数据传送到单一的Reduce节点上，这大大减少了系统Reduce任务的数量，提高了整个系统的并行执行效率。

在Reduce阶段，只需要按照组内offset的顺序将密文排列好，输出到HDFS文件系统中即可。如果用户希望最终的加密结果存放到单一文件，那么Reduce的任务数据应该为1，但是这样会影响Reduce的并行度。因此，对于较小的文件，可以将Reduce任务设为1，而较大的文件存储到多个文件，规定好文件名来表征最终的密文文件来自同一明文文件即可。

同样，对文件的解密过程也采用Hadoop的分布式计算模型MapReduce进行，其解密过程和加密过程类似，只是解密过程中map阶段采用解密算法，其他部分基本类似，这里不作赘述。

## 2. RSA加密过程

RSA是一种公钥加密算法，因此在生成随机密钥时，会用公钥对AES加密的密文密钥进行加密，而对应的私钥需要进行妥善的保管。由于HDFS文件系统不善于保存小文件，本文在安全密钥管理服务器中引入MySQL数据库，用来保存RSA的私钥。在保存的过程中，为了后面的解密过程能顺利找到AES加密的密钥和对应密文，本文还需要之前AES加密后的密文路径。因此，可以在MySQL按这个行格式来保存数据：密文路径---RSA私钥---AES密钥密文。

由于安全密钥管理服务器管理着所有的私钥，因此对其具有较高的安全要求，对此，安全密钥管理服务器需要最大程度的物理隔离，用户无法直接访问它。安全密钥管理服务器只和HDFS中的主节点通信，在访问时需要进行PKI身份认证。具体的过程描述和之前3.2.2节类似。

同样对于文件的读取，本文先用RSA进行解密，获得密钥密文，再进行AES解密，获得最终的明文。过程和写入文件顺序相反，但是具体方法类似，此处不详加说明。

本系统在文件存储时进行了加密操作，它在一定程度上可能会影响整个系统的读写性能，但是，由于用户的很多信息或者数据文件的私密程度尤其高，如个人的私密照，个人

的身份证明消息等需要进行最高层次的安全加密措施，为此引入的安全密钥管理服务器，保证了系统的安全性能。通过最后整个实验平台的性能测试分析，整个系统的读写性能和CPU、内存效率等都会有所下降，但有些时候，在性能下降可接受范围内，个人的信息安全比系统性能更为重要，为此该系统还是有一定的可用价值。

### 3.3 HBSSP 的资源池化

#### 3.3.1 安全云服务资源池化

云计算的核心理念就是通过互联网网络按需提供给用户各种IT资源。为达到这个目标，云服务提供商首先要保证拥有一个容量充足的资源池以满足在并发业务高峰时刻仍能满足用户的服务要求，这就是云服务的资源池化，而这个容量的充足、可扩展的资源池也就是按需业务提供的基础。

若将云计算技术通过互联网为客户提供了按需的更为安全的网络服务时，现阶段已经实现了一种全新的网络安全服务模式，称为安全云服务资源池化。安全云服务资源池化是指在多台安全设备中采用集群技术或者在安全处理软件中引入了分布式计算技术，从而形成了对用户透明的统一网络安全能力资源池的过程。

安全云服务资源池化过程中要解决的主要问题是可扩展性、可管理性和可靠性。可扩展性指的是资源池的容量可以根据用户业务需求的增加进行弹性的扩充；可管理性是指在资源池化之后系统具有对资源池进行统一的监控调度和分配能力；可靠性指的是资源池化后的资源池具有统一的一体化的协同处理和容错能力，不会因为特定的物理安全设备单元的故障或者失效影响到整个资源池的正常工作。

#### 3.3.2 ZFS 与 HDFS 集成设计

针对这些问题，在上图的HBSSP系统中，本文在后台分布式文件系统中引入了动态文件系统（Zettabyte File System, ZFS），实现了安全云服务资源池化，能够很好的解决上述问题。

在Hadoop集群主要实现了一个分布式文件系统HDFS，用户可以轻松地在Hadoop上开发和运行处理海量数据的处理。但是HDFS并不是一个万能的文件系统，当文件想写到HDFS上时，首先需要将文件缓存到本地的临时存储。

传统的存储是基于物理卷作映射的，受物理设备的限制，数据相对独立，无法有效的整合资源，在云存储时代，传统存储已经很难适应。ZFS文件系统采用存储池来管理空间，

能够提供真正的海量存储和高度可靠性<sup>[53]</sup>。基于ZFS和HDFS的优秀特性，本节着重分析了ZFS文件系统与HDFS的结合，可以满足当今日益扩展的大数据存储需要和较高的数据完整性及可靠性。

首先，ZFS与Hadoop的结合，能够更好地实现云存储系统的扩展性问题，结合过程如图3-9所示。

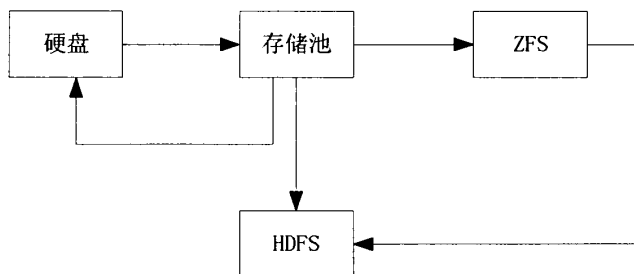


图 3-9 ZFS 与 HDFS 结合过程

1. 需要将硬盘资源构建成一个存储池,使用命令: `zpool create tank sda sdb sde`, 其中, `sda`, `sdb`, `sde`分别代表3个硬盘, `tank`为存储池名称。
2. 使用该存储池构建ZFS文件系统, 使用命令: `zfs create tank/hadoop`, 并将其挂载到名为hadoop的文件夹上, 使用命令: `zfs set mountpoint=/hadoop tank/hadoop && zfs mount -a`, 如此, 即可在hadoop文件中对ZFS存储池进行操作。
3. 将存储池及文件系统都挂载好之后, 在HDFS文件系统中修改配置文件, 其核心部分的配置为<property>

`<name>dfs.data.dir</name>`

`<value>/hadoop</value>`

`</property>`

之所以设置为/hadoop, 是因为在之前ZFS设置挂载点时已经设置为hadoop, 将两者结合, 当分布式文件系统运行起来后, 数据都在已经建好的池中读取。该配置需要在每个节点上设置, 否则配置不一致, 该节点就无法启动, 从而影响整个存储集群。至此, 其实已经完成了ZFS作为HDFS后端文件系统的构建。当构建好资源池后, 可以通过Hadoop集群中的命令查看到系统的存储空间得到了有效的扩展。

另外, 针对保存在该分布式文件系统上的数据, 本文采用ZFS的快照(Snapshot)技术能有效地避免数据的丢失<sup>[38]</sup>。快照技术是一种极其重要的存储数据的技术, 它可以在不停止

机器的应用程序情况下对数据进行备份，所以对系统的安全性有了进一步的提高。

ZFS文件系统通过使用Copy-on-write的方式提供Snapshot功能，为系统提供了更安全有效的数据保护，其工作原理如图3-10所示：

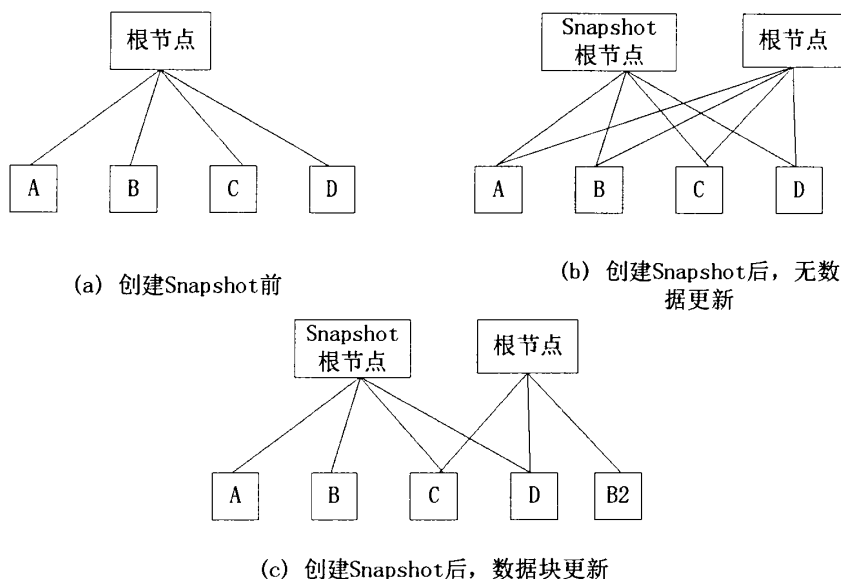


图3-10 Snapshot工作原理

图3-10(a)为创建Snapshot之前的文件块状态，其根节点指向A、B、C、D四个数据块(data block)，图3-10(b)为创建了一个新的Snapshot之后，系统就对根节点产生了一个相同的Snapshot根节点，并且该新的根节点和原来的根节点指向完全一样的数据块，此时没有任何数据块的更新。这时候，若文件系统发生了一定的变化，如不小心误删除了一个数据块B，那么在变化的文件系统中此文件已经不再存在了，但是，在磁盘上并未真正删除，仍旧保存着该文件，Snapshot中仍然记录该文件的消息，如图3-10(c)中B2数据块。同样，若修改文件系统中某个数据块，系统就会在修改前先将数据块进行复制，然后对根节点的数据进行修改，同时对复制好的数据块进行修改。这样不仅提高了复制效率，而且还提高了系统磁盘的空间利用率。

在本文提出的基于Hadoop的安全存储平台下，引入了ZFS文件系统，并通过具体的实际案例操作，在创建快照后，对文件系统中的数据进行了删除或者修改，实验表明经过删除或者修改操作后，并没有真正从磁盘上删掉已被删除或者修改过的数据，避免了不正常操作情况下误删数据的情况发生，从而有效地保护了该系统下的数据，进一步保证了整个

分布式文件系统的安全性和可靠性。

### 3.4 本章小结

本章主要对当前开源云存储平台 Hadoop 存在的安全问题进行一个优化，提出了一个基于 Hadoop 的安全存储平台。

首先，本章分析了 HDFS 中存在的 Kerberos 身份认证安全性不够高，从而提出了 HBSSP 的基于 PKI 的身份认证机制和 HDFS 数字签名机制；

接着，分析了 HDFS 的加密设计，对原有 HDFS 文件系统中文件写入过程的基础上，增加了文件的加密设计过程，对文件进行混合加密后再存储到云端分布式系统中，提高了文件信息的安全性和完整性；

最后，为保证整个系统数据完整性和可靠性，提出安全服务资源池化，引入了 ZFS 文件系统，使得整个系统的扩展性和可靠性有了明显的提高。

## 第 4 章 云存储平台实验和性能分析

为了进一步证明上述部署云存储安全系统的性能,本文搭建了如下的云存储实验平台,并对整个优化后的安全分布式云存储平台进行性能测试和分析。

### 4.1 测试环境

系统的测试环境包括 Hadoop 集群、认证服务器、应用服务器、安全密钥管理服务器以及客户端。其中 Hadoop 存储集群由 4 台服务器组成,通过千兆交换机互联。每台服务器都安装了 Centos 6.3 操作系统,内核版本是 3.6.6,基本配置是 CPU: Intel Xeon E5606 2.13GHz, 4GB 内存,双硬盘(WD 7200 RPM 500GB 和 WD 7200 RPM 2TB)。Hadoop 的版本是 Hadoop-1.0.4,后端资源池所采用的 ZFS 版本是 zfs-2.6.32-358.2.1。

认证服务器和应用服务器硬件配置与之前的服务器相同,操作系统同样为 Centos 6.3,需安装 tomcat6 与 Apache2。安全密钥管理服务器中采用的 MySQL 版本为 mysql server 5.1.61,客户端为普通的 windows 7 系统。各节点和服务器的 IP 地址分配如表 4-1 所示。

表 4-1 各节点和 web 服务器对应的 IP 地址分配

机器名称	IP地址	担任角色	备注
Master.Hadoop	192.168.5.99	Namenode-Master	Server1
Slave1.Hadoop	192.168.5.41	Datanode-Tasktracker	Server2
Slave2.Hadoop	192.168.5.42	Datanode-Tasktracker	Server3
Slave3.Hadoop	192.168.5.43	Datanode-Tasktracker	Server4
应用服务器	192.168.5.44	Web站点服务	Server5
认证服务器	192.168.5.45	Web站点服务	Server6
安全密钥管理服务器	192.168.5.46	密钥服务	Server7

### 4.2 测试结果与性能分析

考虑到系统中增加加密功能后对系统性能带来的影响,本文采用不同大小的文件来进行数据写入和读取速度的对比测试。在具体的测试方案中,本文对比了三种不同的情况,分别是未使用加密、使用 AES 加密和使用 DES 加密。表 4-2 为三种情况下不同大小的文件写入时间对比。

表 4-2 不同操作下写入时间对比

文件大小	未使用加密所需时间	使用AES加密所需时间	使用DES加密所需时间
0.2G	24s	38s	45s
0.4G	47s	71s	90s
0.6G	80s	155s	181s
0.8G	114s	189s	230s
1.0G	142s	248s	302s
1.2G	147s	293s	350s
1.5G	163s	322s	384s

将表 4-2 中的数据进行抽象化成图表后如图 4-2 所示。图 4-2 表示系统在三种不同操作下写入文件时的速度对比。

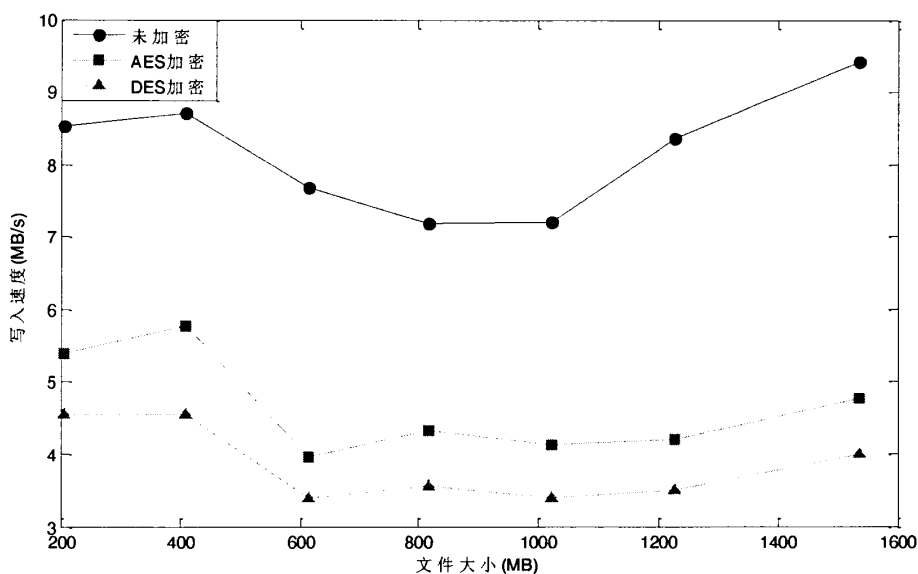


图 4-2 系统在三种不同操作下写入文件时的速度对比

从表 4-2 和图 4-2 的分析结果可以知道：在系统中采用加密操作后，整个系统的文件写入时间增加，并且随着文件大小的增加，对文件的写入时间也相应的增加，从而导致整个文件写入速度有所下降。但是，对用户来说，因为有些文件涉及到更多的私密信息，因此需要采用安全的加密手段来保护信息安全。从测试结果中可以看到，在写入大约为 1G 左右的文件时，采用 AES 加密比采用 DES 加密所花费的时间约少 22%，而且，AES 相比于 DES 具有更高的安全性，因此，本系统采用 AES 加密来维护数据安全是较为合理的。



同样，将文件以加密方式写入到分布式文件系统中后，在文件的读取时候用户需要对文件进行解密操作，进而对文件内容有效的读取。

表 4-3 为本文所设计的安全云存储平台下系统采用不同操作后文件读取时间对比。

表 4-3 不同操作下文件读取时间对比

文件大小	未使用加密所需时间	使用AES加密所需时间	使用DES加密所需时间
0.2G	9s	13s	17s
0.4G	21s	28s	35s
0.6G	45s	70s	90s
0.8G	72s	103s	142s
1.0G	112s	128s	153s
1.2G	118s	142s	169s
1.5G	130s	171s	212s

将表 4-3 中的数据进行抽象化成图表后如图 4-3 所示。图 4-3 表示系统在三种不同操作下读取文件时的速度对比。

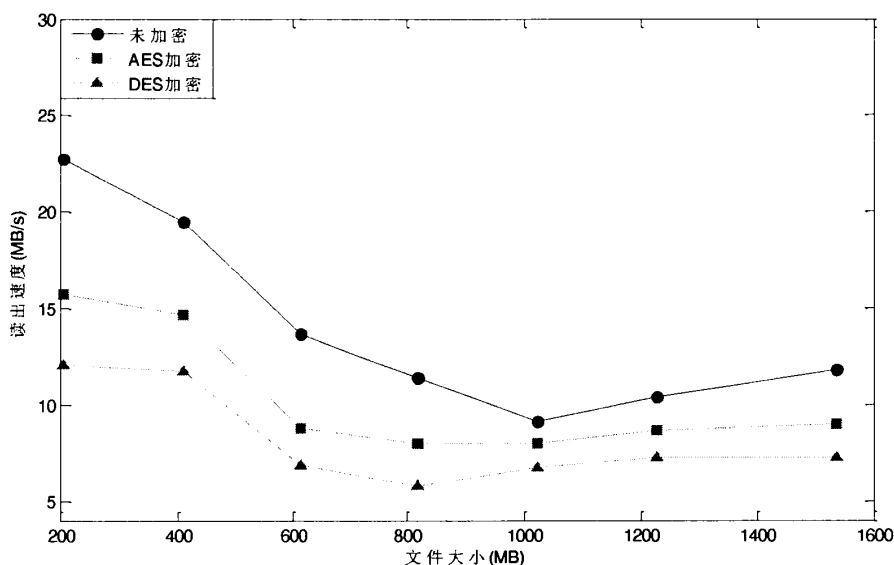


图 4-3 系统在不同操作下读取文件速度对比

从表 4-3 和图 4-3 的结果可知：在系统中引入 AES 加密操作后，文件的读取时间和文件写入时间趋势基本相同，即文件在加密后读取出来比未采用加密所消耗的时间更大，

这是现实不可避免的问题,当用户读取大文件的时候所需的时间相对小的文件需要更多时间,但是,从表 4-2 和表 4-3 中可以表明,用户从云存储服务端读取文件所需的时间相对写入时间更少,同时,从图 4-3 和图 4-2 对比可知,在三种不同情况下,文件的读取速度差相较于写入速度差更小,这也意味着用户在云端下载文件时,采用加密之后体验到的下载时间差别较小,这样也就满足了云存储系统下文件的共享需求,适合于 HDFS 中文件单次写入多次读出的特点,从而更好地利用整个云存储平台。

此外,在上述HBSSP系统中采用了ZFS固态缓存管理技术,在实验中本文采用FIO工具对ZFS缓存加速前后随机读取不同大小文件进行每秒读写操作次数(Input / Output Operations Per Second, IOPS)性能测试,得到了良好的效果,如图4-4所示。

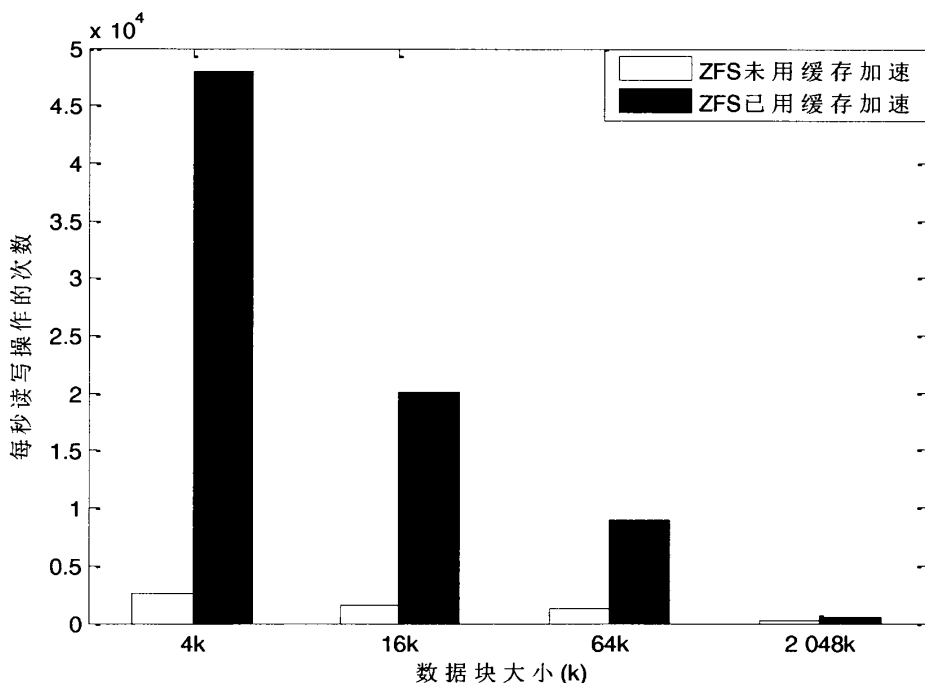


图4-4 ZFS做缓存加速前后每秒读写操作次数对比

采用FIO工具对ZFS缓存加速前后随机读取不同大小文件进行IOPS性能测试,文件数据块大小分别为4kB,16kB,64kB和2048kB。通过柱状图可以看到,在随机读取文件时,采用ZFS缓存加速后,其IOPS性能与用缓存加速相比较性能大约提升了10倍。另外一个重要指标是吞吐量,ZFS做缓存加速后吞吐量大约为500Mb/s,比未加速的提高近5倍,而对云端存储的文件比较大时,采用本文设计的分布式文件存储系统有一定的优势,像对视频、大

量照片等文件的读取操作时，性能要求比较高，因此ZFS在这方面起到非常大的作用。

### 4.3 本章小结

本章主要对上述设计的安全分布式云存储系统进行了实际性能测试，得到了基本的分析结果。通过对文件加密后读写性能的比较测试，得知整个系统性能有所下降，在性能下降可接受范围内，个人的信息安全比系统性能更为重要，同时在安全资源池中，本文将ZFS和HDFS结合，对ZFS进行不同文件的IOPS性能测试，证明ZFS有较高的读写性能，并能保证数据的完整性，为此该系统还是具有一定的可用价值。

## 第 5 章 基于 KAE 算法的密钥管理研究

在前面的章节中，文章已经实现了基于 Hadoop 的安全存储平台的研究和设计，在该平台中本文使用了 AES 和 RSA 加密算法来保证数据的安全性，而现有的密码学中不管是对称密钥加密还是非对称密钥加密甚至是两者结合的加密即使都是基于密钥安全的，密钥是整个安全系统的关键所在，因此密钥管理成为安全分布式云存储系统中的一个核心环节。为此，本文针对云存储中数据共享环境下用户数量增加导致密钥数量也急剧增加的情况，对用户间文件共享产生的密钥进行安全有效地管理，从而使得整个云存储平台共享效率提高。

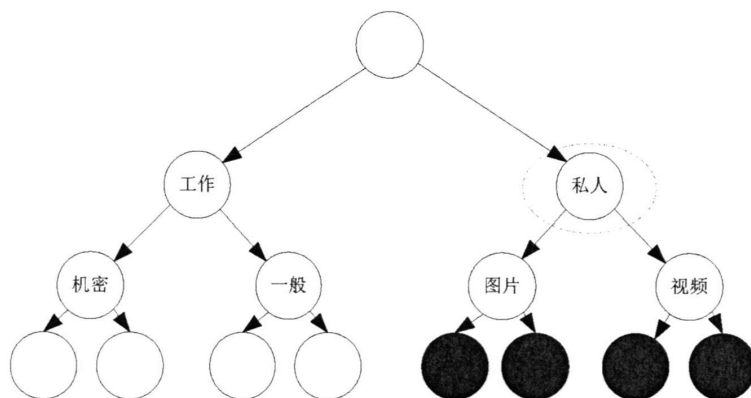
本文针对上述第三章分布式存储系统中的部分数据如何进行安全有效的共享，引入一个特殊类型的非对称加密算法叫密钥聚合加密（Key Aggregate Encryption, KAE）方案，并将该算法和其他同类算法进行一个比较，证明该算法在非对称加密中对数据的安全共享有一定的优势，从而为用户间进行部分数据安全共享提供了可靠的保障。

### 5.1 同类算法分析

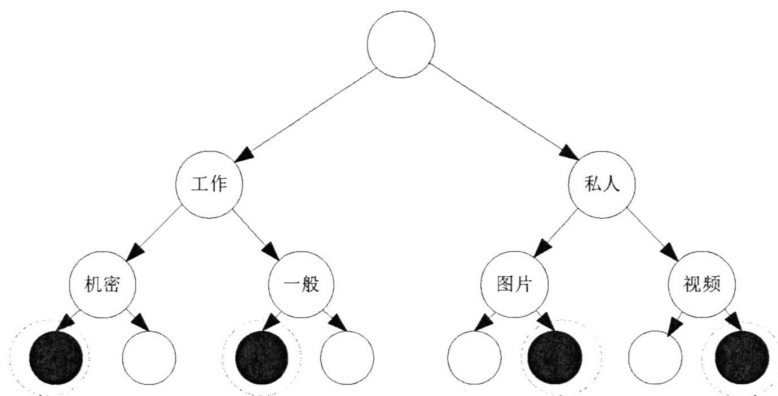
#### 5.1.1 分层加密密钥算法

在现代加密学中，人们常常做的研究是将数据分成很多小部分，然后进行多次加密。这其中必然会涉及到多个密钥管理问题。而密钥分配方案旨在最小化存储和管理密钥的开销。使用树结构，人们可以通过一个给定的分支密钥来推导出其子节点的密钥，同样，保证父节点的密钥就能隐性地保证了子节点的密钥。Sandhu 提出了一种方法，它是通过使用对固定密钥进行伪随机或者是分组密码块的重复评估，从而生成一个树状分层结构的对称密钥<sup>[54]</sup>。

下面采用树结构来对分层加密密钥算法进行研究。例如，首先 userA 根据他的主体将密文分类，如图 5-1 所示：



(a) 需要一个密钥



(b) 需要四个密钥

图 5-1 树状结构

上面树中一个节点代表着需要一个密钥，叶子节点代表着私人密文信息的密钥。实心圆代表着密钥可以分配给整个类，虚线画出来的代表密钥需要授权。注意：非叶子节点的每一个密钥可以推导出他的子节点密钥。

在图 5-1(a)中，如果 userA 想要将“私人”这类文件都共享，他只需要将节点“私人”授权一个密钥，这样就会自动的给下面的子节点“图片”“视频”赋予委派的密钥。这是一种理想的状况，因为这里大部分的共享类属于同一个分支，这样只要父节点的密钥就足够了。

然而，在一般条件下，情况并没有这么简单。如图 5-1(b)所示，如果 userA 想要共享“机密”中的一个文件和“一般”中的一个文件给他的同事，这时候 userA 能做的事只有给 userB 多个密钥才能访问这些文件，这样就导致了整个密钥大小的增加。当碰到更为复杂的情况时，这个方法并不够灵活，并且基于分层的加密算法中，解密密钥大小往往根据分层多少来定，其大小并不固定。

一般来说,如果一个人想要共享他某个分支下的所有文件时,分层方法能够解决部分问题。但是,一般来说,密钥的数量随着分支数量的增加而增加。但是不可能提出一个分层方法来同时保证密钥总数的减少,又保证所有个人有访问不同的叶子节点的权限。

### 5.1.2 基于对称密钥的紧凑密钥算法

受上面灵活的分层结构问题的启发, Benaloh et al.在原先提出的广播环境下简单传输大量密钥的情况下进行了改进,提出了一种新的加密方案<sup>[55]</sup>。构建比较简单,这里简单地介绍下密钥推导过程的具体内容,密钥中一个类的集合(这是所有可能的密文的子集)的推导过程如下:

1. 选择一个  $N = p \cdot q$ ,  $p$  和  $q$  是两个大的随机素数;

2. 再从随机域  $\mathbb{Z}_N^*$  选择一个主密钥  $Y$ 。每一类结合一个特定的素数  $e_i$ 。所有的这些素数放在一个公共系统参数中。固定大小的密钥集合  $S'$  可以由下面的产生:

$$K_{S'} = Y^{\prod_{e_i \in S'} (e_i)} \bmod N \quad (5-1)$$

这样,可以分配给  $S$  (其中  $S' \subset S$ ) 一定的访问权限。

3.  $K_{S'}$  可以由  $K_S^{\prod_{e_i \in S'} (e_i)}$  计算得到。这个方法可以得到类似的性能和特性,但是这种方法是用对称加密算法的。这样密钥管理服务器需要知道更多的相关私人密钥来加密数据,这在很多情况下是不适用的。由于该方法用来产生一个私人密钥值而不是公钥/私钥对,因此如何在公私密钥对情况下来使用这个想法还并不明确。

在对称算法中,已经有专家提出如何减少密钥长度来实现消息认证,但是,如何提高共享解密能力仍未考虑。

### 5.1.3 基于身份的紧凑密钥算法

IBE(Identity-Based Encryption)<sup>[56]</sup>是一种非对称加密类型,也叫公钥加密类型,该加密算法是将用户的标识字符串(如邮件地址)等设为自己的公钥。在 IBE 算法中使用有一个可信方叫做私钥生成器,该私钥生成器根据根据不同用户的身份授予不同的主密钥和私密密钥。接着加密服务器能够用公共的参数和用户的身份来加密消息。接收者能够根据自己的秘密密钥来解密密文。

Guo<sup>[57]</sup>等人试着采用密钥聚合的方法来建立 IBE,这些算法中有一个是假设随机产生

数据库但是其他的都不是。在这些算法中，密钥聚合受到限制，从某种程度上说他们的密钥是根据各个不同身份个体来进行聚合的。尽管有很多的身份和秘密密钥，但是他们中间只能通过一个多项式来聚合。更重要的是这个密钥聚合相对密文和公共参数而言必须以牺牲 $O(n)$ 为代价的，这里 $n$ 指秘密密钥，它能被聚合成一个固定大小。然而，这大大增加了存储和传输密文的开销，这在共享的云存储应用中是不切实际的。

在 Fuzzy IBE 加密算法中，单个紧凑密钥可以解密多个身份下的密文，这些身份是在某个特定的尺度空间上是相近的，但是不能用来解密任意一个身份，因此，该算法在一定程度上也受到了限制。

通过以上这些相关类似算法的分析和研究，分层密钥管理算法会随着分支的增加，其密钥数量也会随之增加；而基于对称密钥的紧凑密钥算法不能解决高效共享解密的能力；而基于身份的紧凑密钥算法提出聚合的思想，但是该聚合相对密文和公共参数而言要付出更多的代价，因此在存储和开销上也有所限制。因此，本文利用聚合的思想，提出基于密钥聚合加密（Key Aggregate Encryption, KAE）的数据安全共享研究方案。

## 5.2 KAE 算法设计

### 5.2.1 问题提出

在第三章设计到的分布式云存储系统中，假设 userA 将自己的个人照片或者私人视频放在云存储上，但是云存储服务提供商必须确保数据的隐私性。由于数据泄露的可能性，userA 对云存储服务商自己提供的隐私保护机制不是很放心，所以在上传照片之前，她用私钥对照片进行了加密。但是 userA 的同事 userB 想要查看这些年来照片中有 userB 本人的相关信息，如部分照片和视频。userA 能够使用云存储的共享机制，但是现在的问题是怎样分配这些照片的解密权限给 userB。其中有一种办法就是 userA 会通过安全通道或者相应的安全协议传送给 userB 这些照片的所有密钥。但是一般情况下会采取下面两种极端方式：

1. userA 用单个加密密钥加密了所有的文件，然后直接给 userB 相应的解密密钥；
2. userA 用不同的密钥加密每一个文件，然后将相关的密钥再发给 userB。

显而易见，第一种方法的不足是 userA 将自己的所有私密信息全部暴露在外。第二种方法是考虑到实际生活中的效率问题。可以说有这么多成千上万的共享照片就有成千上万的共享密钥。传输这些密钥需要通过一个安全的信道，存储这些密钥需要更昂贵的安全存储空间。一般来说，消耗的费用和复杂度会随着解密密钥的增加而增加，可以说这么做非

常麻烦而且非常浪费的。

从第二章 2.2 数据加密这小节，可以知道加密算法有两种形式：对称加密和非对称加密。使用对称加密，其加密和解密效率会比较，开销相对较小，但是安全性并不能保证；使用非对称加密，在上述情况下会变得相当复杂。

因此，针对上述情况，最好的解决方法是 *userA* 用一个特别的公开密钥进行加密文件，但是只对 *userB* 发送一个固定长度的解密密钥。由于解密密钥需要通过一个安全信道传送并且要秘密保存起来，因此应采取较小的长度。本文研究的是如何使得解密密钥更为强大，从某种意义上说是仅仅使用一个解密密钥就可以来解密多个密文，但不增加密钥的长度。

### 5.2.2 算法设计及应用

针对上述的问题，本文引入一个特殊类型的公钥加密算法叫做密钥聚合加密 (Key Aggregate Encryption, KAE)。在 KAE 中，用户加密不只是简单的用公钥加密，而是在密文中添加了一个标示符(Identifier)，可以称为类(class)。这意味着密文被分成几类。这个密钥的拥有者(Key Owner, KO)持有一个主密钥(Master-Secret Key, MSK)，该密钥可以分离出不同类中的私密密钥，更重要的是，被分离出来的密钥进行聚合后称为聚合密钥(Aggregate Key, AK)，这对单个类来说如同一个紧凑密钥 (Compact Key, CK)，但是这么多密钥聚合起来的聚合密钥能力等同于任何密文类中的任何一个子集的解密能力。

一个密钥聚合加密算法由下面五个主要部分组成：

1. 数据拥有者(Data Owner, DO)通过 *Setup* 建立一个公共系统参数；
2. 通过 *KeyGen* 生成一个公共密钥/主密钥对；
3. 消息通过 *Encrypt* 并结合明文数据决定密文中的类 *class*；
4. 数据拥有者使用主/私密钥来产生一个聚合解密密钥给密文类子集，通过分离函数

*Extract* 产生。产生的密钥通过安全的邮件或者安全设备传给授权者(*delegatee*)；

5. 用户能够使用 *Decrypt* 函数，通过聚合密钥来解密密文，在聚合密钥中包含了密文的标示符 *class*。

#### A. 算法详细描述和分析：

设  $G$  和  $G_T$  是两个素数  $p$  阶循环群，定义  $\hat{e}: G \times G \rightarrow G_T$  是一个满足下面两个特性的映射：

双线性：



$$\begin{aligned}
&\forall g_1, g_2 \in G, \\
&a, b \in Z, \\
&\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}
\end{aligned} \tag{5-2}$$

非简并性:

$$g \in G, \hat{e}(g, g) \neq 1 \tag{5-3}$$

如果上述涉及到的所有操作都是可效计算的, 则  $G$  是一个双线性群。许多椭圆曲线分类都具有双线性群的特征。

本文根据 Boneh et al. 等人提出的抗共谋广播加密算法<sup>[58]</sup>而引入 KAE 优化算法, 之前的算法设计开始之初为固定大小的私密密钥, 但是每个密钥只能解密相关特定指数的密文。因此, 为了让系统有更好的性能, 对算法 *Extract* 和 *Decrypt* 进行了优化。

1. *Setup*( $1^\lambda, n$ ): 随机选择一个双线性素数  $p$  阶群  $G$ , 其中:

$$2^\lambda \leq p \leq 2^{\lambda+1}, \quad g \in G, \alpha \in_{\mathcal{R}} Z_p \tag{5-4}$$

对  $i = 1, \dots, n, n+2, \dots, 2n$ , 计算下面公式(5-5):

$$g_i = g^{\alpha^i} \in G \tag{5-5}$$

输出系统参数记为  $param = \langle g, g_1, \dots, g_n, g_{n+2}, \dots, g_{2n} \rangle$ , 其中  $\alpha$  可以在 *Setup* 过程后删除。注意: 每个密文分类可以用整数记为  $\{1, 2, \dots, n\}$ ,  $n$  是最大密文标示类。

2. *KeyGen*( ): 选择一个  $\gamma \in_{\mathcal{R}} Z_p$ , 输出一个公钥( $pk$ )和主私钥  $msk$  对:

$$\begin{aligned}
pk &= v = g^\gamma \\
msk &= \gamma
\end{aligned} \tag{5-6}$$

3. *Encrypt*( $pk, i, m$ ): 对任何一个消息  $m \in G_T$ , 一个指数  $i \in \{1, 2, \dots, n\}$ , 随机选择一个  $t \in_{\mathcal{R}} Z_p$ , 然后计算出密文:

$$C = \langle g^t, (vg_i)^t, m \cdot \hat{e}(g_1, g_n)^t \rangle \tag{5-7}$$

4. *Extract*( $msk = \gamma, S$ ): 对于指数  $j$  的  $S$  集, 这个聚合密钥可以通过公示(5-8)计算得到:

$$K_S = \prod_{j \in S} g_{n+1-j}^\gamma \tag{5-8}$$

由于  $S$  不包括 0, 所以从参数  $param$  中总是能得到  $g_{n+1-j} = g^{\alpha^{n+1-j}}$ .

5.  $Decrypt(K_s, S, i, C = \langle c_1, c_2, c_3 \rangle)$ : 如果  $i \in S$ , 返回消息输出, 方程如(5-9)所示:

$$m = c_3 \cdot \hat{e}\left(K_s \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}, c_1\right) / \hat{e}\left(\prod g_{n+1-j}, c_2\right) \quad (5-9)$$

对数据的拥有者, 只要知道  $\gamma$ , 那么很容易就能得到如下公示:

$$\hat{e}(c_1, g_n)^\gamma = \hat{e}(g', g_n)^\gamma = \hat{e}(g_1, g_n)^i \quad (5-10)$$

B. 算法紧凑性分析和正确性证明:

➤ 紧凑性分析: 对于任意的整数  $\lambda$ ,  $n$ , 集合  $S$ , 指数  $i \in S$  和任意消息  $m$ , 可以知道:

$$Setup(1^\lambda, n) \rightarrow param, \quad KeyGen() \rightarrow (pk, msk),$$

$$Extract(msk, S) \rightarrow K_s, \quad Encrypt(pk, i, m) \rightarrow C$$

其中  $|K_s|$  和  $|C|$  仅仅取决于安全参数  $\lambda$ , 而并不取决于密文类  $n$ .

➤ 正确性分析: 对任意整数  $\lambda$  和  $n$ , 任意集合  $S \subseteq \{1, \dots, n\}$ , 任意指数  $i \in S$  和任意消息  $m$ , 可以得到

$$P_r[Decrypt(K_s, S, i, C) = m : param \leftarrow Setup(1^\lambda, n),$$

$$(pk, msk) \leftarrow KeyGen(), C \leftarrow Encrypt(pk, i, m),$$

$$K_s \leftarrow Extract(msk, S)] = 1.$$

➤ 正确性证明:

$$\begin{aligned}
& \frac{c_3 \cdot \hat{e}\left(K_S \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}, c_1\right)}{\hat{e}\left(\prod_{j \in S} g_{n+1-j}, c_2\right)} \\
&= c_3 \cdot \frac{\hat{e}\left(\prod_{j \in S} g_{n+1-j}^\gamma \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}, c_2\right)}{\hat{e}\left(\prod_{j \in S} g_{n+1-j+i}, (vg_i)'\right)} \\
&= c_3 \cdot \frac{\hat{e}\left(\prod_{j \in S, j \neq i} g_{n+1-j+i}, g_i^t\right)}{\hat{e}\left(\prod_{j \in S} g_{n+1-j}, g_i^t\right)} \\
&= c_3 \cdot \frac{\hat{e}\left(\prod_{j \in S} g_{n+1-j+i}, g_i^t\right) / \hat{e}\left(g_{n+1}, g_i^t\right)}{\hat{e}\left(\prod_{j \in S} g_{n+1-j+i}, g_i^t\right)} \\
&= m \cdot \frac{\hat{e}\left(g_1, g_n\right)^t}{\hat{e}\left(g_{n+1}, g_i^t\right)} \\
&= m
\end{aligned} \tag{5-11}$$

在本文提出的解决方案中，userA 只要通过安全邮件给 userB 发送一个聚合密钥就可以了。userB 能够从云存储中下载加密的照片然后使用聚合密钥来解密这些照片。情景描述可以如图 5-2 所示。

在 KAE 方案中，密文、密钥、主密钥、聚合密钥都是相同的固定大小的。公共系统参数是随着密钥的分类呈线性增长的，但是每次都只需要其中的一小部分，它从巨大的但非机密的云存储中按需获取到。而以前的研究结果可能达到类似的效果，即具有相同的固定大小的解密密钥，但是这些类需要一些预定义的分层关系。而本文提出的方案是灵活的，即消除了这种约束关系，即这些类中没有特殊的关系。

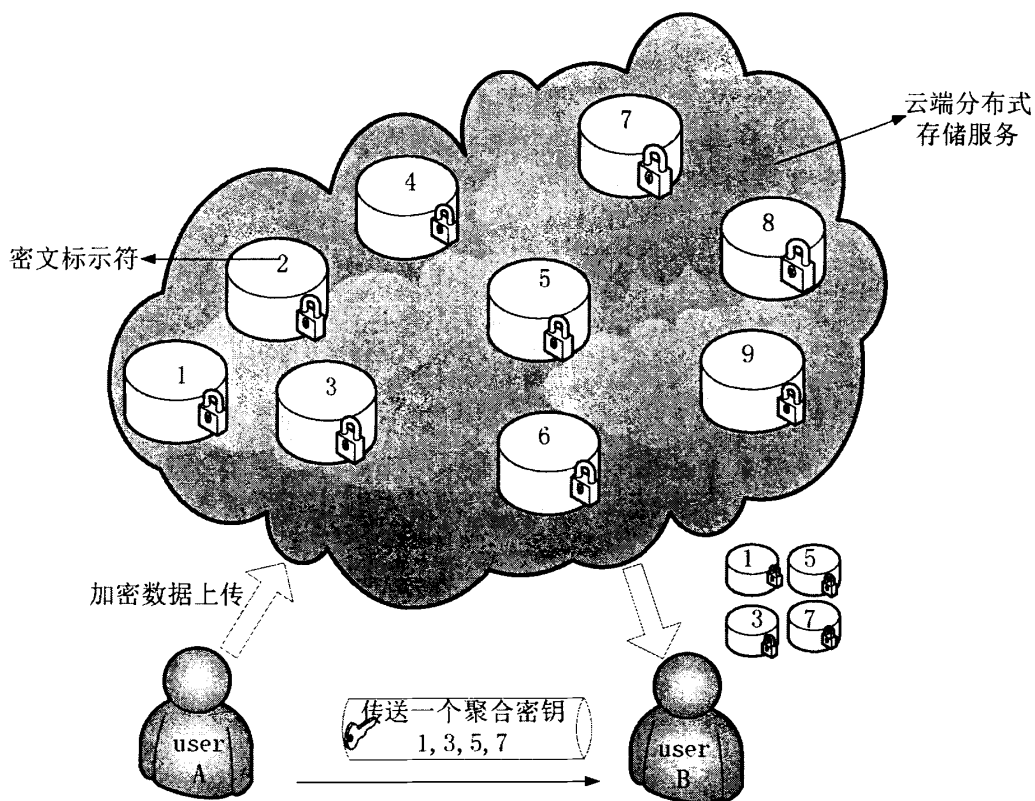


图 5-2 聚合密钥

在图 5-2 中，使用 KAE 算法来进行云端分布式存储文件之间的共享。

1. 从图中可知，userA 想要共享服务器上的数据  $m_1, m_2, \dots, m_v$ ，userA 执行  $Setup(1^\lambda, n)$  来获得参数  $param$ 。

2. 然后，执行  $KeyGen()$ ，得到公/私密钥对  $(pk, msk)$ ，并将这些密钥保存在本文系统设计中的安全密钥管理服务器中。在本系统中，安全密钥管理服务器主要功能就是负责生成密钥和保存管理用户密钥，若将生成的密钥由用户自己管理，在很大程度上不能保证密钥的安全性，如用户可能意外将密钥丢失。

3. 执行  $Encrypt(pk, i, m)$ ，将加密好的数据传送到分布式存储系统中。

4. 当得知了  $param$  和  $pk$  后，需要和 userA 共享的用户可以通过私钥来更新服务器上的数据。当 userA 愿意将自己一部分数据集  $S$  共享给他的朋友 userB 时，userA 只要执行  $Extract(msk, S)$  来计算出聚合密钥  $K_s$ 。由于  $K_s$  是一个固定大小的密钥，可以通过安全邮件

传送给 userB。

5. 当 userB 获得访问权限后, 用聚合密钥下载云端数据。也就是说, 对于任意  $i \in S$ , userB 能够从服务器上下载  $C_i$  和一些必要参数。通过  $Decrypt(K_s, S, i, C_i)$  函数解密下载下来的各类密文  $C_i$ 。

### 5.3 KAE 算法分析

通过上面的分析, 本文对基于 KAE 加密的算法和其他数据安全共享算法进行一个比较, 具体可以由表 5-1 所示:

表 5-1 KAE 算法和其他数据安全共享算法比较

算法类型	加密类型	密文大小	解密密钥大小
预定义分层的加密密钥算法	对称加密/非对称加密	固定	根据分层决定, 大多数情况不固定
基于对称密钥的紧凑密钥算法	对称加密	固定	固定
基于身份的紧凑密钥算法	非对称加密	固定	不固定
KAE 算法	非对称加密	固定	固定

为了更具体地进行比较, 本文对 5.1.1 节提到的基于树的密钥分配方法进行了分析和仿真。它使用了完全子树的方法, 是广播加密问题中最具代表性的解决方案。令完全二叉密钥树的高度是  $h$ , 它能表示  $2^h$  数量的密钥类。

图 5-1(a)表示了一种理想的情况, 即在只拥有一个密钥的情况下可以获得  $2^h$  个类的权限,  $h_s$  表示某个子树的高度。图 5-1(b)表示了另一种极端情况, 即为了解密某一密文类集合, 需要持有大量的密钥。因此, 本文引入了一个参数  $n_a$ , 用来表示在这种分层密钥方法下的所需要的平均密钥数目。

下面通过 MATLAB 进行数值仿真分析。假设密文类的数目是  $2^h$ ,  $r$  是分配系数, 表示在所有密文类中分配的授权密文类的比例。显然, 当  $r=0$  时,  $n_a$  应该为 0, 意味着任何类都没有权限; 当  $r=100\%$  时,  $n_a$  应该为 1, 意味着只有根密钥可以获取所有  $2^h$  类的权限。一般情况下, 人们希望  $n_a$  首先随着  $r$  的增加递增, 而后递减。设  $r=10\%, 20\%, \dots, 90\%$ ,

随机选择比例以保证授权方式的任意性。对于每一种  $r$  和  $h$  的组合，随机生成  $10^4$  个不同的类组合，输出的  $n_a$  则表示授权密钥的平均值，结果如表 5-2 所示。

表 5-2 不同分配系数  $r$  和  $h$  下的压缩比

$h$	$r$	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95
16	$n_a$	6226	11744	16567	20552	23501	25310	25441	23265	17341	11731
	$\frac{n_a}{N}(\%)$	4.75	8.96	12.64	15.68	17.93	19.31	19.41	17.75	13.23	8.95
17	$n_a$	12452	23488	33135	41104	47002	50620	50882	46530	34682	23462
	$\frac{n_a}{N}(\%)$	4.75	8.96	12.64	15.68	17.93	19.31	19.41	17.75	13.23	8.95
18	$n_a$	24904	46976	66269	82208	94004	101239	101764	93061	69363	46923
	$\frac{n_a}{N}(\%)$	4.75	8.96	12.64	15.68	17.93	19.31	19.41	17.75	13.23	8.95

在表 5-2 中， $h$  分别为 16,17 和 18，对于一个给定的  $h$ ， $n_a$  随着分配系数  $r$  的增加而增加，直到  $r$  达到 0.7 左右。同时，通过数据发现  $n_a$  和  $N$  的比例只和  $r$  有关，和  $h$  无关，其中  $N=2^{h+1}-1$ ，表示密钥的总数。这是因为在密文类数量( $2^h$ )很大且分配比例固定的情况下，这种随机授权的方式几乎达到了相同的密钥分配比例。因此，在  $r$  相同的情况下， $n_a$  随着  $h$  指数增长。因此，在给定  $r$  和  $h$  的情况下，可以很容易的估算出需要分配多少授权密钥。

对于一个特定的  $h$  值，本文引入了一个参数压缩因子  $F$ ，用来表示每一个授权密钥可以解密的密文类的数目，即所有分配的授权密文类  $r2^h$  和所需授权密钥数  $n_a$  的比率。越高的压缩因子表示每个授权密钥可以解密越多的密文。图 5-3 表示压缩因子和分配系数的关系。

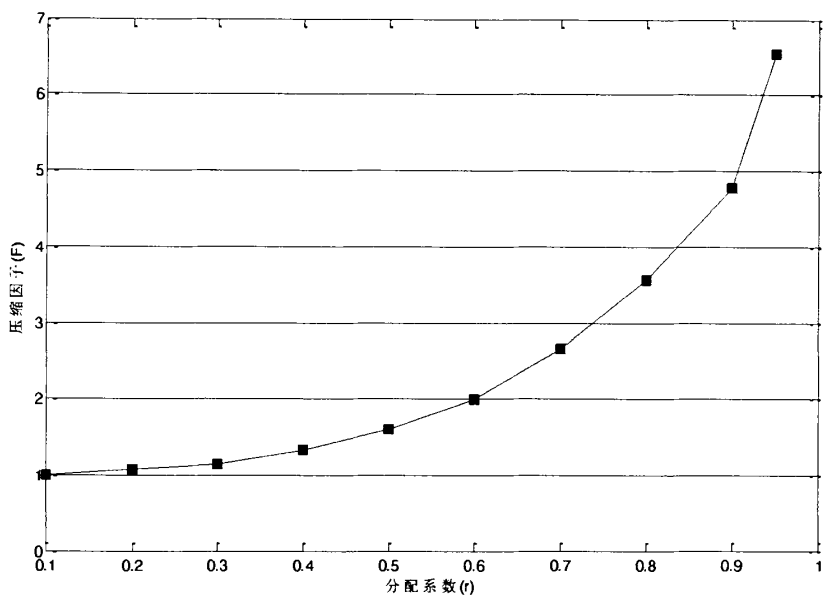


图 5-3 压缩因子和分配系数的关系

从图 5-3 中可以看到，当分配系数越接近 1，压缩因子越大。

为了比较算法的优劣势，本文在不同分配系数比较了三种不同方法的授权密钥数，结果如图 5-4 所示。

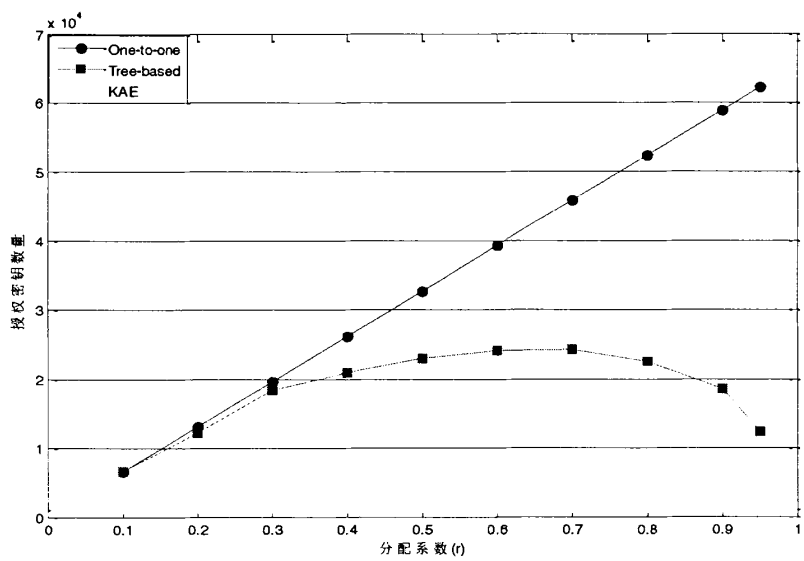


图 5-4 算法比较

图 5-4 中给出了三种方法的比较结果,第一种是按顺序单个授权密钥(One-to-one),从图中可以看出授权密钥的数目和分配的密钥类数目相等;第二种是基于树的分层密钥方法(Tree-based),结果和之前分析的类似,密钥数目先增加,在 0.7 左右的位置开始减少;第三种即本文的 KAE 算法,密钥数目是一个固定的数值,且远远小于前两种方法。由此可见,本文的 KAE 算法在密文数目巨大,且需要进行数据共享的情况下具有比较的优势,因此,对于云存储安全共享,本文的 KAE 算法具有较大的参考意义。

## 5.4 本章小结

为保证合理有效地解决密钥管理问题,文章进行了进一步的探索和研究。在本章节中引入基于 KAE 加密算法,将众多私钥聚合起来,使它们变成一个单一的紧凑密钥,通过对当前主要的三种同类算法的分析,并进行算法之间的比较,从而体现本文引入的算法较其他三种算法有一定的优势,通过对 KAE 算法原理的研究后,在接下来的课题研究中会将算法运用到之前设计的云存储系统中,从而能有效地保证用户间数据的安全存储和文件间安全共享。



## 第 6 章 全文总结与展望

### 6.1 全文总结

大数据时代的到来促进了云存储技术的发展,当前,很多国内外研究机构和企业都在云存储系统上进行了大量的研究,但是在云计算环境下,数据的存储安全问题变得越来越突出,云存储的安全性问题已经成为人们研究的热点。

本课题是基于云计算的数据存储安全的研究,在云存储安全问题日益突出的环境下,文章通过对数据安全的各项关键技术的研究,利用 Hadoop 开源平台,设计和实现了一套安全云存储系统平台,实现安全应用。下面对文章作如下几个方面的总结:

1. 本文分析了当前云计算和云存储安全技术的发展和国内外的研究现状,重点研究了本课题所采用的数据存储安全相关技术等,为本文的方案提供理论基础。
2. 在上述理论技术背景下,文章利用当前开源云平台 Hadoop,对 HDFS 进行分析和优化,完成安全分布式云存储平台的设计和实现,设计一个基于 Hadoop 的安全存储平台,并在系统中加入了安全资源池,将 ZFS 文件系统和 HDFS 集成结合使用,保证整个系统资源的可扩展性和可靠性。
3. 对整个 Hadoop 安全存储平台的性能进行分析和评估,虽然在系统读写性能上有所下降,但是在人们可接受范围内,保障了数据的存储安全性能,接着还对 ZFS 文件系统的 IOPS 性能进行分析和测试,表明该文件系统有很高的读写性能。
4. 为保证部分数据能够有效的安全共享,文章引入基于 KAE 加密算法的安全文件共享方案,通过与其他三种算法比较,证明本文引入的算法较其他三种算法有一定的优势,并将算法运用到之前设计的云存储系统中,有效地保证用户间数据的安全存储和共享。

### 6.2 研究展望

经过相应的研究和设计,系统从应用安全角度出发,基本实现了相应的安全需求,但由于时间和个人能力有限,今后在如何全方面完善系统的安全性仍需进行大量的研究。针对本文设计的基于 Hadoop 的安全分布式文件系统也要进行改善。本文存在的不足和下一步的研究的工作有以下几点:

1. 本文设计的系统中带有安全密钥管理服务器,该密钥管理服务器也是基于安全认

证的，若其遭遇到不合法的安全攻击，就有可能导致密钥丢失，引发新的安全问题。

2. 本文采用的分布式文件系统适合对大文件的存储和加密，如何实现对小文件进行有效存储和加密也是今后要研究的问题之一。

3. 本文第五章提到的算法存在一定的局限性，在云存储环境中，密文的数量随时有所变化，而算法中密文类的最大数是有预定义边界值，需要预留足够多的密文类。

4. 本文引入的基于 KAE 加密算法在探索和研究成果方面还存在着一定的局限性，由于时间和个人能力有限，并没有在真实的网络环境下进行分析和测试，在今后的工作中需要进一步研究。

## 参 考 文 献

- [1] 马晓昊. 基于云计算的安全数据存储服务的研究与实现[D]. 上海: 同济大学, 2008.
- [2] 郭雪梅. IBM 云存储路线图:更快,更简单,更安全[EB/OL].  
<http://www.csdn.net/article/2814416-Draw-2013-IBM-cloud-storage-roadmap>, 2013-03-08/2014-03-21.
- [3] 冯丹. 网络存储关键技术的研究及进展[J]. 移动通信, 2009, 33(11): 35-39.
- [4] 赵旺. 分布式并行文件系统锁管理的研究与设计[D]. 武汉: 华中科技大学, 2007.
- [5] Amazon SimpleDB (beta) [EB/OL]. <http://aws.amazon.com/simpledb/#sdb-vs-s3>.
- [6] Amini A. Secure Storage in Cloud Computing [D]. Kongens Lyngby, Denmark: Technical University of Denmark, 2012.
- [7] 王鹏. 云计算的关键技术与应用实例[M]. 北京: 人民邮电出版社, 2011.1.
- [8] Farentinos C. Security dropbox with pivoting service bin [P]. U.S. Patent : 6719195, 2004-4-13.
- [9] Ruff N, Ledoux F, Eads I W. A critical analysis of dropbox software security [A]. ASFWS 2012 [C]. Application Security Forum. 2012.
- [10] Rong C, Nguyen S T, Jaatun M G. Beyond lightning: A survey on security challenges in cloud computing [J]. Computers & Electrical Engineering, 2013, 39(1): 47-54.
- [11] 吕文俊. 基于 HADOOP 技术云存储管理层的设计与实现[D]. 哈尔滨: 哈尔滨工业大学, 2011
- [12] Kelly Sims. IBM Introduces Ready-to-Use Cloud Computing [EB/OL].  
<http://www-03.ibm.com/press/us/en/pressrelease/22613.wss>, 2007
- [13] Kotla R, Alvisi L, Dahlin M. SafeStore:a durable and practical storage system [A]. Proceeding of the 12<sup>th</sup> USENIX Annual Technical Conference [C]. Berkeley, CA, USA: USENIX Association, 2007, 129-142.
- [14] 冯登国, 张敏, 张妍. 云计算安全研究[J]. 软件学报, 2011, 22(1): 71-83.
- [15] 周敏奇, 王晓玲, 金澈清等. Hadoop 权威指南(第 2 版)[M]. 北京: 清华大学出版社, 2011.7
- [16] 张云勇, 陈清金, 潘松柏等. 云计算安全关键技术分析[J]. 电信科学, 2010 (9): 64-69.
- [17] Menezes A J, Oorschot P C, Vanstone S A. Handbook of Applied Cryptography [M]. CRC Press, 1996.
- [18] 余彬彬. 动态身份认证系统的研究与实现[D]. 上海: 上海交通大学, 2008.
- [19] 崔培枝, 王朝君. Kerberos 认证技术研究及分析[J]. 计算机与现代化, 2001, (5): 35-39.
- [20] 杨宇. 基于 PKI 身份认证系统的研究和实现[D]. 成都: 电子科技大学, 2009.
- [21] 振胜. 公钥基础设施 PKI 及其应用[M]. 北京: 电子工业出版社, 2008.
- [22] 刘影. 基于生物特征的身份认证研究与设计[D]. 成都: 西南交通大学, 2006.
- [23] Litigation A. Encrypt Your Data [J]. InfoTech Research Group, 2006.
- [24] Yu S, Wang C, Ren K, et al. Achieving secure, scalable, and fine-grained data access control in cloud computing [A]. INFOCOM, 2010 Proceedings IEEE [C]. San Diego, CA: IEEE, 2010, 1-9.
- [25] 陈晓明. 数据加密方法的分析[J]. 科技资讯, 2008 (9).
- [26] 胡乐明, 冯明, 唐宏. 云计算安全技术与应用[M]. 北京: 电子工业出版社, 2012.2.

- [27] Biham E, Shamir A. Differential cryptanalysis of the data encryption standard [M]. New York: Springer-Verlag, 1993.
- [28] Nadeem A, Javed M Y. A performance comparison of data encryption algorithms [A]. Information and communication technologies [C]. IEEE, 2005, 84-89.
- [29] Daemen J, Rijmen V. The design of Rijndael: AES-the advanced encryption standard [M]. Springer, 2002.
- [30] Pethe H B, Pande S R. A survey on different secret key cryptographic algorithms [J]. IBMRD Journal of Management & Research, 2014, 3(1).
- [31] Schneier B. Description of a new variable-length key, 64-bit block cipher (Blowfish) [A]. Fast Software Encryption [C]. Springer Berlin Heidelberg: Lecture Notes in Computer Science, 1994, 191-204.
- [32] 卓先德, 赵菲, 曾德明. 非对称加密技术研究[J]. 四川理工学院学报: 自然科学版, 2010, 23(5): 562-564.
- [33] Blakley G R, Borosh I. Rivest-Shamir-Adleman public key cryptosystems do not always conceal messages [J]. Computers & Mathematics with Applications, 1979, 5(3): 169-178.
- [34] Kravitz D W. Digital signature algorithm [P]. U.S. Patent : 5231668. 1993-7-27.
- [35] Kant K, Iqbal F, Alam M I. Digital Signature and Key Agreement [J]. IJSEAT, 2014, 1(7): 266-270.
- [36] 王尚平, 王育民, 王晓峰等. 基于零知识证明的前向安全数字签名方案[J]. 通信学报, 2003, 24(9): 42-47.
- [37] 许德武, 陈伟. 基于椭圆曲线的数字签名和加密算法[J]. 计算机工程, 2011, 37(4): 168-169.
- [38] 倪琴琴, 孙卫真, 刘峥. ZFS 文件系统 Snapshot 技术的分析[J]. 计算机教育, 2009, 14: 01-03.
- [39] 敖一峰. 磁盘级的 ZFS 数据跟踪与隐藏技术[D]. 上海: 上海交通大学, 2010.
- [40] Casualfish. ZFS 文件系统简介: 发展史及特性[EB/OL].  
<http://storage.chinabyte.com/183/12593683.shtml>, 2013-04-16/2014-03-21
- [41] Rodeh O, Teperman A. ZFS-a scalable distributed file system using object disks [A]. Mass Storage Systems and Technologies, 2003. Proceedings. 20th IEEE/11th NASA Goddard Conference on [C]. IEEE, 2003: 207-218.
- [42] 李嘉. ZFS 文件系统浅析[J]. 广播电视信息, 2008(3): 31-36.
- [43] 张韶英. 基于 Hadoop 的企业海量数据存储与计算平台的设计与实现[D]. 成都: 电子科技大学, 2010.
- [44] Borthakur D. The hadoop distributed file system: Architecture and design [J]. Hadoop Project Website, 2007, 11: 21.
- [45] Ghemawat S, Gobioff H, Leung S T. The Google file system [A]. ACM SIGOPS Operating Systems Review [C]. ACM, 2003, 37(5): 29-43.
- [46] Hadoop Makes Sense of Lots of Data [EB/OL].  
<http://www.enterprisestorageforum.com/article.php/3890191/Hadoop-MakesSense-of-Lots-of-Data.htm>.
- [47] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system [A]. Mass Storage Systems and Technologies [C]. IEEE, 2010: 1-10.
- [48] Le H H, Hikida S, Yokota H. NameNode and DataNode Coupling for a Power-Proportional Hadoop Distributed File System [A]. Database Systems for Advanced Applications [C]. Springer Berlin Heidelberg: Lecture Notes in Computer Science, 2013, 7826: 99-107.
- [49] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters [J]. Communications of

the ACM, 2008, 51(1): 107-113.

- [50] 杨帆, 沈奇威. 分布式系统 Hadoop 平台的视频转码[J]. 计算机系统应用, 2011, 20(11): 80-85.
- [51] Li F, Ooi B C, Ozsu M T, et al. Distributed data management using mapreduce [J]. ACM Computing Survey, 2013.
- [52] 马媛. 基于 Hadoop 的云计算平台安全研究[J]. 信息安全与通信保密, 2012 (6): 89-92.
- [53] 许孝明. 浅析 ZFS 文件系统及应用[J]. 数字技术与应用, 2012 (6): 219-220.
- [54] R.S. Sandhu. Cryptographic Implementation of a Tree Hierarchy for Access Control [J]. Information Processing Letters, 1988, 27(2): 95-98.
- [55] Benaloh J, Chase M, Horvitz E, et al. Patient controlled encryption: ensuring privacy of electronic medical records [A]. Proceedings of the 2009 ACM workshop on Cloud computing security [C]. ACM, 2009: 103-114.
- [56] D. Boneh and M.K. Franklin. Identity-Based Encryption from the Weil Pairing [A]. Proc. Advances in Cryptology [C]. Springer Berlin Heidelberg: Lecture Notes in Computer Science, 2001, 2139: 213-229.
- [57] F. Guo, Y. Mu, and Z. Chen. Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key [A]. Proc. Pairing-Based Cryptography Conf [C]. Springer Berlin Heidelberg: Lecture Notes in Computer Science, 2007, 4575: 392-406.
- [58] D. Boneh, C. Gentry, and B. Waters. Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys [A]. Proc. Advances in Cryptology Conf [C]. Springer Berlin Heidelberg: Lecture Notes in Computer Science, 2005, 3621: 258-275.

## 致 谢

时光荏苒，转眼间在浙江工业大学的七个年头即将结束。这里有我最美好的本科时光，也有我最充实的研究生生涯。本科毕业时候的场景还历历在目，一晃三年研究生生涯又将结束。在这毕业之际，我要感谢母校给我提供良好的学习资源，让我在自己的漫漫人生路上取得不断的进步。

研究生阶段的三年时光里，无论在为人处事方面，还是在科研能力方面，我都有了很大程度的提高，这一切都离不开老师的谆谆教诲。在这里，首先我要感谢的导师方路平教授和陈清华教授，他们在我的学术生涯中给予了很多帮助。从论文的选题到论文的开题，再到最后论文的撰写阶段，都给予了我很多宝贵的意见，让我在学术道路上少走了很多弯路。

接着，我要感谢学校提供我一个去浙江清华长三角研究院合作交流的机会，无论在学术和做人方面都得到了提升。在此，特别感谢研究院裘韵芳老师、王吉文、孙瑾、毕莎莎等人对我生活和学术上的帮助，谢谢你们对我的关心，让我在这里快速成长。

还要感谢实验室的师弟师妹们和寝室同学，他们认真、严谨、踏实的科研态度深深地感染了我，在论文写作阶段给我创造一个良好的实验环境，使我跟他们共同成长。此外，在最后的论文撰写阶段，我要特别感谢潘清老师对我的论文提出的宝贵意见和建议，谢谢你们！

特别感谢父母和家人对我的无私奉献，你们一如既往的理解、支持和关怀让我度过了一个又一个难关，使我顺利完成学业！

最后感谢在百忙之中帮我评阅论文的各位专家和教授，感谢所有支持和帮助过我的人，谢谢！

## 攻读学位期间参加的科研项目和成果

### 参加的科研项目

- [1] 浙江省十二五重大科技专项资助项目（ZJC01037）

### 录用和发表的论文

- [1] 高煜红, 陈清华, 方路平. ZFS 在基于 Hadoop 视频点播系统中的应用[J]. 2013, 33(5): 102-105.(已发表).

# 云计算数据存储安全的研究

作者：[高煜红](#)  
学位授予单位：[浙江工业大学](#)

引用本文格式：[高煜红](#) [云计算数据存储安全的研究](#)[学位论文]硕士 2014