|  |  |
| --- | --- |
|  | 学校代码： 10246 |
|  | 学 号： 15212010038 |
|  |  |



|  |
| --- |
| 硕 士 学 位 论 文 |

**（专业学位）**

|  |
| --- |
| **面向云存储的高效中间件优化** |

|  |
| --- |
| **optimization of Efficient Middleware of Backup Cloud** |

院 系： 软件学院

专业学位类别（领域）： 软件工程

姓 名： 刘武

指 导 教 师： 韩伟力 教授

完 成 日 期： 2015年 月日

**指导小组成员名单**

韩伟力 教授

目录

[摘要 1](#_Toc488066856)

[Abstract 1](#_Toc488066857)

[第一章 绪论 2](#_Toc488066858)

[1.1 背景介绍 2](#_Toc488066859)

[1.2 研究内容及意义 2](#_Toc488066860)

[1.3 论文组织结构 4](#_Toc488066861)

[第二章 核心技术 5](#_Toc488066862)

[2.1 中间件 5](#_Toc488066863)

[2.2 Java第三方开发包 5](#_Toc488066864)

[2.3 WebExtension 5](#_Toc488066865)

[2.4 大文件分块上传/下载 6](#_Toc488066866)

[2.5 断点续传 6](#_Toc488066867)

[2.6 本地缓存 6](#_Toc488066868)

[第三章 原云存储中间件系统概述 7](#_Toc488066869)

[3.1 原中间件系统整体架构 7](#_Toc488066870)

[3.1.1 数据传输模块 7](#_Toc488066871)

[3.1.2 数据压缩模块 7](#_Toc488066872)

[3.1.3 数据加密模块 7](#_Toc488066873)

[3.1.4 安全会话模块 7](#_Toc488066874)

[3.2 原中间件系统缺陷分析 7](#_Toc488066875)

[3.2.1 安全性分析 7](#_Toc488066876)

[3.2.1 稳定性分析 7](#_Toc488066877)

[3.2.1 性能分析 7](#_Toc488066878)

[第四章 云备份中间件系统优化分析 8](#_Toc488066879)

[4.1 系统总体框架 8](#_Toc488066880)

[4.2 优化项分析 8](#_Toc488066881)

[4.2.1 交互适配管理 8](#_Toc488066882)

[4.2.3 会话持久化管理 8](#_Toc488066883)

[4.2.3 本地缓存管理 8](#_Toc488066884)

[4.2.4 系统优化管理 8](#_Toc488066885)

[第五章 云备份中间件系统优化设计 8](#_Toc488066886)

[5.1 系统结构图 8](#_Toc488066887)

[5.2 系统框架设计 8](#_Toc488066888)

[5.3 类图设计 8](#_Toc488066889)

[5.4 流程设计 8](#_Toc488066890)

[5.4.1 系统总流程设计 8](#_Toc488066891)

[5.4.2 交互适配流程设计 8](#_Toc488066892)

[5.4.3 会话持久化管理流程设计 8](#_Toc488066893)

[5.4.4 本地缓存流程设计 8](#_Toc488066894)

[5.4.5 系统优化流程设计 8](#_Toc488066895)

[5.4 系统接口设计 8](#_Toc488066896)

[第六章 云备份中间件系统优化实现 9](#_Toc488066897)

[第七章 云备份中间件系统优化结果分析 9](#_Toc488066898)

[第八章 总结和展望 10](#_Toc488066899)

[6.1 总结 10](#_Toc488066900)

[6.2 展望 10](#_Toc488066901)

[参考文献 10](#_Toc488066902)

[致 谢 10](#_Toc488066903)

# **摘要**

~~随着互联网技术的高速发~~展，网络已经渗透到我们生活的个个角落，随之而产生的海量数据亟需一种高效的方式进行存储。传统的数据存储方式因其容量有限、使用不方便、数据容易丢失等局限性，早已无法满足用户的需求，因此云存储技术应运而生。在短短几年的时间内，就有数十个云存储产品面世，从国外的iCloud、GoogleDrive、Dropbox到国内的百度云、阿里云、腾讯云，众多公司都想在这个被众人看好的新兴市场中分到一杯羹。持续升温的云存储行业在近几年面临着激烈的竞争，随之而来挑战也刺激着各大云存储平台推出新的技术，以解决云备份服务衍生出的安全性、稳定性及网络等问题。

本论文主要通过研究云储存服务中的关键技术，对一款自行研制的云存储中间件进行优化。基于前期的成果，在Linux平台上，利用本地存储和计算能力，减少数据上传下载等I/O密集型服务，提高服务性能；通过加密的形式，保证用户信息和数据的安全性；同时，借助大文件分块传输和断点续传技术，解决大文件上传过程中遇到网络中断或服务器死机等故障而引发的数据重传问题；此外，优化并完善平台框架和API支持机制，解决不同应用和多种云存储服务之间适配的问题，以支持更多的桌面OS组件使用云备份与云存储服务。

**关键词：**

http://www.docin.com/p-1691130603.html

# Abstract

~~With the rapid development of Internet technology~~, the network has penetrated into every corner of our lives. It needs an efficient way to store the massive resulting data. However, the traditional data storage methods has long been unable to meet the needs of users because of its limited capacity, the use of inconvenient, easy to lose data and other limitations, so cloud storage technology came into being. In just a few years, there are dozens of cloud storage products launch on the market. From abroad iCloud, Google Drive, Dropbox to the domestic Baidu cloud, Ali Cloud, Tencent Cloud, many companies who are optimistic about the emerging markets want to be in this crowd. Continued warming of the cloud storage industry in recent years is facing fierce competition, In order to occupy more market share, the challenge has also spurred the new cloud storage platforms to introduce new technologies to solve the issues derived from cloud storage like security, stability and network problems.

This paper mainly studies the key technology in cloud storage service, and optimizes a self-developed cloud storage middleware. Based on previous results, in the Linux platform, exploiting the local storage and computing power to reduce data upload and download I / O-intensive services, to improve service performance. Through the form of encryption to ensure that user information and data security. At the same time, this paper uses the technology of block upload and breakpoint resume to avoid data retransmission caused by network problem or server problem. In addition, the platform framework and API support mechanisms are optimized and refined to address the adaptation issues between different applications and multiple cloud storage services to support more desktop OS components using cloud backup and cloud storage services.

**Keywords：**

# 第一章 绪论

## 1.1 背景介绍

近年来互联网技术飞速发展，计算机已渗透到我们生活的各个角落，从个人办公、娱乐，到大型企业的自动化管理，计算机都扮演着举足轻重的角色。早期计算机的存储空间十分有限，一般只有MB级的存储空间，计算机操作者对存储空间的使用都需经过严格的规划。但随着存储技术的发展，存储空间发生了质的变化，数据量的大小由MB级增长至PB级，并且仍在不断增长。传统的数据存储方式因其容量有限、使用不方便、数据容易丢失等局限性，早已无法满足用户的需求，这对现有的存储系统提出了尖锐的挑战，随之而产生了云存储技术成为了大数据存储的发展趋势。

云存储技术备份是通过集群应用、网格技术或分布式文件系统等功能，为用户提供远程的数据备份服务。用户可以通过云储存客户端，将个人的数据如个人信息、图片、文档等备份到云存储服务器上。现有的云存储服务供应商大多都能为用户提供海量的存储空间，数据的上传与访问的不受空间和设备的限制，并且数据不存在丢失的风险。

介于存储技术的各种优势，越来越多的用户愿意将他们的个人数据存储到云端，大型的机构如政府、医院、企业等也逐步将数据转移到云端。随之而来的用户需求也复杂，除了基本的海量数据存储外，用户更愿意选择性能和用户体验更好的云存储客户端。各大云存储供应商为了占有更多的市场份额，纷纷推出新的技术，以解决云备份服务衍生出的安全性、稳定性及网络等问题。

其中云**存储中间件**（Cloud Backup Middleware）技术是提升云存储服务性能的一项重要环节。中间件是在1964年产生的，IBM发布OS/360操作系统，软件与硬件分离，同时，软件成为一个独立的产业正式登上产业界的舞台。中间件就是软件产业不断发展过程中自然产生的。中间件的作用主要是屏蔽计算机软硬件之间的异构性，包括硬件、操作系统、数据库等。同时，中间件可以实现互操作，因为网络协议和通信机制的不同，不同的系统无法集成，移植也是很困难的，中间件能够很好的解决这一问题。由此可见，中间件可以解决硬件之间的差异性，以及不同操作系统之间的兼容性，因此对云存储系统来说是一项至关重要的技术。

## 1.2 研究内容及意义

原有的云存储中间件系统是国家“核高基”科技重大专项中自主研发的云存储平台中的一个子系统，作为服务器与客户端之间的“桥梁”，降低了服务器与客户端之间的耦合度，实现了服务器与客户端之间的数据传输功能等功能，为本文实现的高效的云存储系统提供了基础。原云备份系统具体实现了以下几点功能：

• 数据传输：实现了服务器与客户端之间的数据交互，主要是数据的上传与下载功能，这也是云存储系统的基本功能。

• 数据压缩：支持对大文件的压缩功能。当上传大文件时，用户可以通过COMPRESS参数指定对文件进行压缩，以减少数据传输的大小，降低所需的网络带宽，提高传输的效率。此外，整个压缩是透明，用户通过压缩上传的文件，下载后是解压的格式。

• 数据加密：为了保证中间件系统能够抵御各种恶意网络攻击，中间件系统会将数据以加密的形式进行网络传输。使用的是高级加密标准（Advanced Encryption Standard，AES）对称加密算法，该算法是当前最流行的对称加密算法。

• 安全会话：中间件作为服务器与客户端之间的桥梁，需要保存用户的登录信息，以方便客户端与服务器之间建立持续的连接。原有的中间件系统采取一种安全的方式以确保用户的信息的安全性。

本文拟对原有的云存储系统中间件系统进行优化，分析原有系统的不足，结合现有的其他云存储产品，在Linux平台上，完善其平台框架和API支持机制，以支撑更多的桌面OS组件使用云备份与存储服务。

本文的主要贡献如下：

• 多应用适配：云存储服务的普及使得云存储客户端呈现出多样化的格局，为了使现有的中间件系统支持更多的客户端，实现了对不同客户端请求的适配。

• 持久会话信息管理：云备份中间件在客户端和服务器之间工作，多个客户端的命令通过中间件提交给服务器。用户的命令会以会话的形式保存在中间件，当中间件系统出现异常时（如网络中断、中间件死机），会在下次重启中间件时自动恢复执行未完成的会话。跟进一步，对会话信息需要进行了物理隔离，并且以加密的形式存储在数据库中。

• 本地缓存：利用现代计算机普遍较大的存储空间以及高性能的计算能力，实现了本地缓存管理，进一步减少了系统I/O操作，提高了中间件性能。

• 系统优化：利用服务器提供的高级功能，实现了大文件分块上传功能，在受到网络状况影响的情况下，若文件上传不成功，可以通过断点续传功能，无需重传。此外，在原有系统中，客户端提交任务如果长时间不活跃，则会导致大量任务的失败和重新提交。优化后的中间件系统支持用户对中间件的授权，从而解决身份认证和授权信息过期所引发的问题。

## 1.3 论文组织结构

本文将分八个章节对论文进行阐述。第一章介绍本文研究内容的背景以及相关的研究工作。第二章介绍本文在研究和实现过程中使用到的核心技术，包括中间件技术、加密算法、Java SDK、WebExtension、大文件上传/下载技术、断线续传技术以及本地缓存技术。第三章将对原有的云备份中间件系统进行分析，对其功能进行全面的介绍。第四章将在第三章的基础上分析原有云备份中间件的不足，指出本文需研究和实现的内容。第五章阐述云备份中间件系统的整体设计，包括优化后的系统框架、类图设计、流程设计以及接口设计。第六章介绍云备份中间件实现的细节。第七章将对云备份中间件优化后的结果进行评估，以实际的测试结果作为评估的标准。第八章对全文进行总结，并对未来进行展望。

# 

# **第**二**章 核心技术**

## 2.1 中间件

中间件是一种独立运行的系统应用软件，可以实现不同技术间的资源共享，被称为“软件胶水”，使得软件开发人员可以更加容易的完成输入与输出之间的通信。中间件把软件组件和企业应用程序连接起来，是一个处于应用程序和操作系统之间的软件。中间件可以通过网络连接两个独立的系统或软件，实现真正地跨平台、跨网络、跨硬件的功能。随着中间件技术的成熟，其在软件开发过程中的分工和重要性越来越明显，目前已被广泛应用于各个领域，显著降低了软件的开发成本。

## 2.2 Java第三方开发包

## 2.3 WebExtension

附加组件(Add-ons)扩展和修改Web浏览器的功能。它们使用标准的web技术编写-JavaScript、Html、CSS再加上一些专用的javascrip API。另一方面，附加组件可以为浏览器增加新的特性或者改变某些网站的外观。

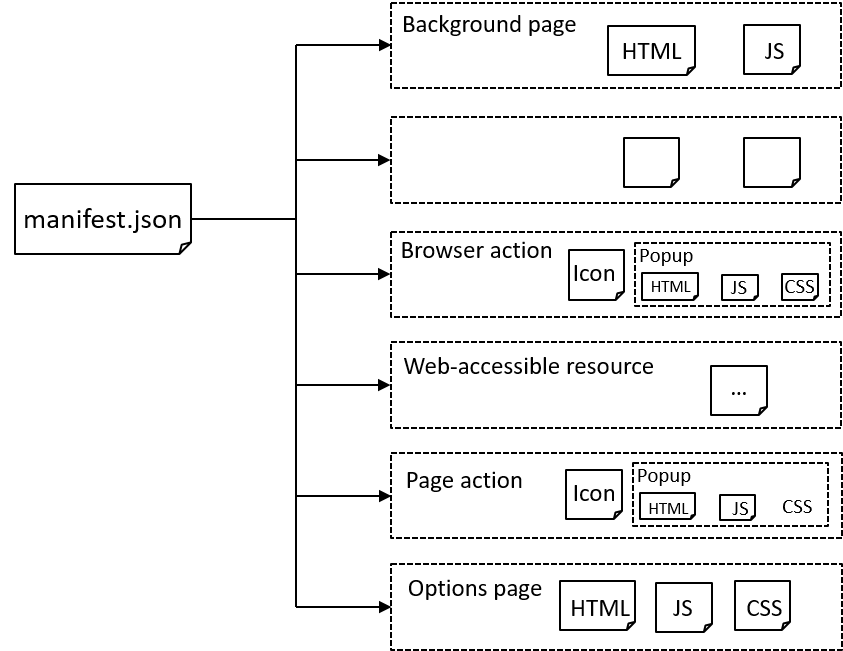
WebExtensions（扩展）是跨浏览器的用于开发附加组件的工具。在很大程度上，与谷歌浏览器Chrome和欧朋浏览器Opera所支持的扩展API 兼容。为这些浏览器所写的扩展在大多数情况下只需少量修改的便可在火狐浏览器FireFox和Microsoft Edge浏览器上运行。这些API与多线程Firefox完全兼容。

图2-1 WebExtension框架图

* background pages: 执行一个长时间运行的逻辑
* content scripts: 与网页进行交互
* browser action files: 在工具栏中添加按钮
* page action files: 在地址栏添加按钮
* options pages: 为用户定义一个可浏览的界面，可以改变插件设置
* web-accessible resources: 使打包好的内容可用于网页与目录脚本

## 2.4 大文件分块上传/下载

## 2.5 断点续传

客户端软件断点续传是指在下载或上传时，将下载或上传任务（一个文件或一个压缩包）人为的划分为几个部分，每一个部分采用一个线程进行上传或下载，如果碰到网络故障，可以从已经上传或下载的部分开始继续上传下载未完成的部分，而没有必要从头开始上传下载。用户可以节省时间，提高速度。

断点续传技术有以下几个特点：

* 断点续传功能，既可节约时间又可以节约金钱
* 定时下载功能，可以为将要下载的软件制定任务列表，让下载软件在规定的时间自动拨号上网并下载软件，下载完毕后再自动挂起Modem，断开与internet的连接，甚至自动关闭计算机
* 多文件同时下载
* 致命错误发生时的关闭机制
* 预防病毒侵害的安全机制，文件下载完毕，即可自动将其发送到指定的病毒的检测软件进行病毒扫描

## 2.6 本地缓存

# 第三章 原云存储中间件系统概述

## 3.1 原中间件系统整体架构

### 3.1.1 数据传输模块

### 3.1.2 数据压缩模块

### 3.1.3 数据加密模块

### 3.1.4 安全会话模块

## 3.2 原中间件系统缺陷分析

### 3.2.1 安全性分析

### 3.2.1 稳定性分析

### 3.2.1 性能分析

# 第四章 云备份中间件系统优化分析

## 4.1 系统总体框架

## 4.2 优化项分析

### 4.2.1 交互适配管理

### 4.2.3 会话持久化管理

### 4.2.3 本地缓存管理

### 4.2.4 系统优化管理

# 第五章 云备份中间件系统优化设计

## 5.1 系统结构图

## 5.2 系统框架设计

## 5.3 类图设计

## 5.4 流程设计

### 5.4.1 系统总流程设计

### 5.4.2 交互适配流程设计

### 5.4.3 会话持久化管理流程设计

### 5.4.4 本地缓存流程设计

### 5.4.5 系统优化流程设计

## 5.4 系统接口设计

# 第六章 云备份中间件系统优化实现

# 第七章 云备份中间件系统优化结果分析

# 第八章 总结和展望

## 6.1 总结

## 6.2 展望

# 参考文献

# 致 谢

**复旦大学**

**学位论文独创性声明**

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。论文中除特别标注的内容外，不包含任何其他个人或机构已经发表或撰写过的研究成果。对本研究做出重要贡献的个人和集体，均已在论文中作了明确的声明并表示了谢意。本声明的法律结果由本人承担。

作者签名： 刘武 日期： 2017.07.02

**复旦大学**

**学位论文使用授权声明**

本人完全了解复旦大学有关收藏和利用博士、硕士学位论文的规定，即：学校有权收藏、使用并向国家有关部门或机构送交论文的印刷本和电子版本；允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其它复制手段保存论文。涉密学位论文在解密后遵守此规定。

作者签名： 刘武 导师签名： 韩伟力 日期： 2017.07.02