(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 102685148 A
(43)申请公布日 2012.09.19

(21)申请号 201210176807.1

(22)申请日 2012.05.31

(71)申请人 清华大学
地址 100084 北京市海淀区清华园 1 号

(72)发明人 舒继武 傅颖勋

(74)专利代理机构 北京思海天达知识产权代理有限公司 11203
代理人 楼艮基

(51)Int.Cl.
H04L 29/06(2006.01)
H04L 29/08(2006.01)

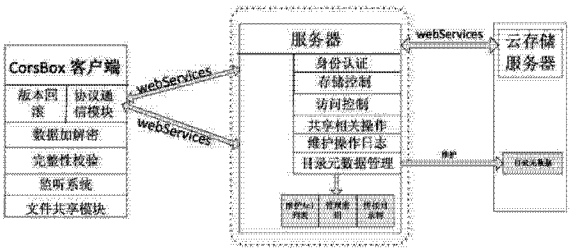
权利要求书 4 页 说明书 14 页 附图 4 页

(54)发明名称

一种云存储环境下的安全网盘系统的实现方法

(57)摘要

在云存储环境下安全网盘系统的实现方法属于存储安全技术领域，其特征在于在云存储环境下的网盘系统的信任体系中，发明了一种系统架构，在服务器中根据用户需求建立信任域，然后利用公钥基础设施 PKI 进行身份认证，保证了用户的不可欺骗性和不可抵赖性；利用哈希算法对文件计算哈希值，利用密钥和对称加密算法 AES 算法对数据加密，之后再将文件密文上传到云存储区中的文件服务器，保证了数据的机密性和完整性；使用基于目录树的同步方式，提高了系统的同步效率与机密性；使用了层级密钥管理的方式，在保证数据安全性的同时减少了系统的管理负担；提供了版本控制功能，保证了文件版本的连续性；提供了多种粒度的加密密钥选择方式，增加了系统的灵活性。



CN 102685148 A

1. 一种云存储环境下的安全网盘系统的实现方法其特征在于是在由客户端计算机、服务器以及云存储服务器共同组成的云存储环境下的安全网盘系统的网络中，依次按以下步骤实现的：

步骤(1)，系统初始化；

客户端计算机以下简称客户端，设有数据加解密模块，数据完整性验证模块，本地监听模块，文件夹共享模块以及协议通信模块；

服务器端设有：身份验证模块，存储控制模块，访问控制模块，版本控制模块以及目录元数据管理模块，其中：

目录元数据管理模块用于对目录元数据进行查询，更新在内的维护操作，所述目录元数据包括：目录类型，拥有者用户名，绝对路径，访问控制列表，授权用户数量以及密钥生成方式，其中访问控制列表包括：用户名和对应的密钥密文，所述密钥密文是使用所对应用户的公钥加密的，对应用户可使用自己的私钥解密；

云存储服务器以下简称云存储端，设有：存储模块和数据可靠模块，其中：

存储模块设有一套供所述服务器端调用的存储接口，以便把需要存储的信息以对象的形式存放在所述云存储端；

数据可靠性模块，根据所述服务器端的需要，为所有存放在云存储端的文件创建一定数量的副本；

步骤(2)，为所述云存储端构建并初始化一个云存储平台，步骤如下：

步骤(2.1)，在装有 Linux 操作系统的各子云存储中分别安装一个多副本，可扩展的分布式云存储系统 swift，并指定其中的一台机器作为 Proxy 代理节点，重配置 Proxy 代理服务，创建账户，容器以及对象，将其配置成三层环的形式，启动 Proxy 代理服务；

步骤(2.2)，配置各云存储服务计算机，实现所述三层环之间的关联；

步骤(2.3)，启动所述 swift 系统，提供云存储服务；

步骤(3)，用户初始化个人信息，步骤如下：

步骤(3.1)，在客户端用户通过调用所述的服务器接口向所述服务器注册新用户；

步骤(3.2)，所述服务器端在确认用户注册所用的用户名与密码后，向注册机构申请一个由公钥基础设施所授予的 X509 证书；

步骤(3.3)，用户申请步骤(3.2)所述的证书后把其中的公钥保存在所述的服务器端；

步骤(3.4)，所述服务器端把包括用户名、密码的哈希值绑定邮箱在内的基本信息保存在数据库中的 users 表中，并创建该用户的服务器端目录树，以文件的形式保存在所述云存储端；

步骤(4)，客户端采用一种目录树比对的方式按以下步骤生成一种在客户端目录树、服务器端目录树和磁盘目录树执行同步操作用的操作队列：

步骤(4.1)，所述客户端向所述服务器端发送同步请求；

步骤(4.2)，所述服务器端在验证当前用户身份后返回用户的最近操作列表，其中包括：此用户在该机器上两次同步之间的这段时间里，该用户在其他机器上对目录或文件所作的删除、移动和重命名操作，其他用户对此用户所共享的目录进行的删除、移动、重命名操作以及当前用户的服务器端目录树；

步骤(4.3)，所述用户逐项执行所述服务器端返回的操作列表，完成文件或目录的以上

三类操作的同步；

步骤(4.4),所述客户端向服务器请求共享密钥,所述服务器验证身份后根据访问控制列表中的信息将所有该用户被授权的所有文件加密密钥返回给客户端；

步骤(4.5),客户端构造初始化目录树,步骤如下：

步骤(4.5.1),所述客户端读取保存在本地的客户端目录树生成文件,在内存中创建该用户的客户端目录树和组成该客户端目录树的客户端目录树结点：

所述客户端目录树是一个二叉树结构,它的格式包括:root、nodes、nodesCount 和 maxnodesCount, 其中,

nodes,以数组方式记录了所述客户端目录树的所有结点；

nodesCount,记录了所述客户端目录树中的总结点数；

maxNodesCount,表示所述客户端目录树最多允许包含的结点数；

客户端目录树结点的格式包括:nodeType、name、appendAttribute、lastModifyTime、lchild 和 rchild,其中：

nodeType,记录所述客户端目录树中的结点所对应的是目录还是文件；

name,记录所述客户端目录树中所述目录或文件的名称；

appendAttribute,对所述客户端目录树中的目录而言,记录的是是否被共享,对所述客户端目录树中的文件而言,记录的是最新版本的版本号；

lchild,记录了在所述客户端目录树中该结点的左孩子的索引号；

rchild,记录了在所述客户端目录树中该结点的右孩子的索引号；

lastModifytime,记录所述云存储环境下的安全网络存储系统 CorsBox 所维护的该结点的最新修改时间；

步骤(4.5.2),客户端计算机根据步骤(4.5.1)的返回的服务器端目录树在内存中创建该用户的服务器端目录树,用于记录所述云存储服务器中的实时数据状态,用户服务器端目录树的结构与客户端目录树完全相同,只是 nodes 数组中的结点类型是用户服务器端目录树结点,用户服务器端目录树结点包括:nodeType、name、appendAttribute、lchild 和 rchild,其中：

nodeType,记录所述用户服务器端目录树中的结点所对应的是目录或文件；

name,记录所述的目录或文件的名称；

appendAttribute,对所述用户服务器端目录树中的目录而言,记录的是是否被共享,对所述服务器端目录树中的文件而言,记录的是最新版本的版本号；

lchild,记录了在所述用户服务器端目录树中该结点的左孩子的索引号；

rchild,记录了在所述用户服务器端目录树中该结点的右孩子的索引号；

步骤(4.5.3),所述客户端计算机在每次同步时扫描磁盘中的所述客户端目录树的目录及其子目录,在内存中创造磁盘目录树记录用户的实时数据,用户的磁盘目录树的结构与客户端目录树完全相同,只是 nodes 数组中的结点类型是磁盘目录树结点,磁盘目录树结点包括:nodeType、name、lastModifyTime、lchild 和 rchild,其中：

nodeType,记录所述磁盘目录树中的结点所对应的是目录还是文件；

name,记录所述磁盘目录树中所述目录或文件的名称；

lastModifytime,记录了在所述客户端计算机中该结点的最后修改时间；

lchild，记录了在所述磁盘目录树中该结点的左孩子的索引号；

rchild，记录了在所述磁盘目录树中该结点的右孩子的索引号；

步骤(4.6)，客户端通过比对步骤(4.5)中构建的目录树，生成待执行操作的操作队列，具体步骤如下：

步骤(4.6.1)，客户端比对客户端目录树与磁盘目录树，并将比对结果保存在集合A中，用来记录用户在此机器上离线进行的操作；

步骤(4.6.2)，客户端比对客户端目录树与服务器端目录树，并将比对结果保存在集合B中，用来记录此用户在此机器上两次同步之间其他客户端进行的操作；

步骤(4.6.3)，客户端比对集合A与集合B中的内容，并根据比对结果生成待执行的操作队列；

步骤(4.7)，逐项执行步骤(4.6)中生成的操作队列，完成同步操作；

步骤(5)，客户端修改访问权限：

步骤(5.1)，客户端共享某文件夹：客户端首先检查待共享文件夹的祖先目录与子孙目录是否被共享，若已被共享，则提示用户此文件夹不能被共享，否则向服务器发起共享请求，服务器收到请求后修改该目录元数据中的共享标志位，然后为此文件夹创建新的操作日志，接着删除该文件夹下所有历史版本，最后向客户端返回共享成功的提示；

步骤(5.2)，客户端添加/删除某用户的访问权限：只有数据的拥有者才能够修改自己共享文件夹的访问权限；若是添加权限，客户端向服务器发起添加权限请求，服务器验证用户身份后将待添加用户的公钥返回给客户端，客户端用此公钥加密该文件夹对应的加密密钥后发送给服务器端，服务器将此密钥的密文写入目录元数据中，并在共享表中添加一条共享记录，最后向客户端返回添加权限成功的提示；

若为删除权限，服务器端验证用户身份后将被取消权限用户及其密钥密文从目录元数据中删除，同时删除共享表中对用的记录，然后向客户端返回删除权限成功的提示；

步骤(5.3)，客户端查看共享内容：客户端查看共享内容依次按照以下步骤来完成：

步骤(5.3.1)，客户端在启动时向服务器发起查看最新共享请求(可以随时手动请求)；

步骤(5.3.2)，服务器收到请求后查找共享表，将客户端未选择是否接受共享的文件名称和拥有者ID返回给客户端，并在共享表中将这些路径置为已处理；

步骤(5.3.3)，客户端弹出对话框请求用户操作，若用户选择接受共享则转步骤(4.3.4)，否则直接返回；

步骤(5.3.4)，服务器端将该共享目录下的所有文件密文和该目录的元数据中对应此用户的ACL项返回给客户端；

步骤(5.3.5)，客户端用自己的私钥从返回的ACL项解密出该文件夹对应的加密密钥；

步骤(5.3.6)，客户端解密出所有文件的明文，然后根据明文后面附着的哈希值做完整性校验；

步骤(5.4)，客户端取消共享：

只有文件夹的拥有者才能取消共享，服务器收到验证用户身份后删除共享表中所有关于此文件夹的共享，接着将该文件夹的目录元数据中的共享标志位改为非共享并删除掉所有的ACL项，然后向客户端返回修改成功的提示；客户端接收到成功提示后修改客户端目录树中该结点的共享状态；

步骤(6):文件版本控制:

步骤(6.1),当客户端上传文件时,服务器首先对该文件的大小进行判断,若是小文件则在文件名末尾添加版本号,并保存在云端;若不是小文件,则直接保存在云端(若已存在则直接覆盖);

步骤(6.2),客户端随时可以进行小文件的历史版本回滚,服务器验证用户身份后返回文件版本列表,用户可以根据自己的需要选择合适的版本进行回滚;

步骤(7),用户选择加密方式:

客户端会在工作目录下自动生成一个 public 文件夹,所有保存在此文件夹下的内容都是不加密的;另外,用户在客户端可以手动修改某个文件夹(非 public 夹)的密钥生成方法,但一旦某个文件夹的密钥生成方式被指定,其祖先目录与子孙目录都不得再被指定密钥生成规则;密钥生成是将用户的加密密钥和该文件夹的名称通过一定的算法生成的,若非特别指定则默认直接使用用户的文件加密密钥对其数据进行加密。

一种云存储环境下的安全网盘系统的实现方法

技术领域

[0001] 云存储环境下的安全网盘系统的实现方法属于存储安全领域，尤其涉及其中的安全访问控制、数据同步、密钥分发管理和文件管理等技术领域。

背景技术

[0002] 随着云计算技术的飞速发展，云存储也逐渐受到了广泛的关注和应用，文件所有者可将自己的机密文件上传到云存储中，由云存储服务提供商进行统一管理，网盘系统便是云存储的一个典型应用。通过网盘系统，文件所有者可以授权其他用户使用自己的文件，通过共享和数据同步的方式来实现用户间的协同工作。

[0003] 虽然网盘系统能够以云存储作为媒介，方便的实现用户间的协同工作，但如果没有安全机制的保护，用户将自己的隐私数据存放在云存储中将存在着巨大的安全隐患。就数据安全来说，第一个要素是数据的机密性。用户将数据若以明文的形式存放在云存储中，云存储服务提供商将可以随意的访问这些数据，如果云存储服务提供商将这些数据用于非法目的，那将给用户带来巨大的损失，甚至产生一些不可预计的后果。安全的第二个要素是数据的完整性。如果数据以明文形式在不安全的网络中传输过程中被非法用户篡改，或者云存储服务提供商有意的删除用户数据中不利于自己的信息，以达到自己的非法目的，用户需要能够检测出该数据是否真的是自己上传的并且未被任何人篡改过。安全的第三要素是可用性，这一点一般的云存储服务提供商都能够保证。另外，在云存储环境下的网盘系统中，安全的权限控制也是一个很重要的问题，因为数据的共享一定会破坏数据的机密性，那么如何在新的信任体系下构建安全机制的架构显得十分重要。

[0004] 一般来说，在云存储环境下的安全网盘系统中，数据拥有者和授权用户都是可信的，他们不会恶意的泄漏或者破坏数据；而云存储服务提供商、网络及非授权用户都是不可信的，他们随时可能泄漏用户的数据，网络和非授权用户甚至有可能恶意的将数据彻底毁坏。但云存储服务提供商基于信誉的考虑，不会轻易做出用户能够发现并可以有力的指控自己的举动来。

[0005] 安全网盘系统是目的是在此信任体系下保证用户数据的安全。它的一般实现方法如下：首先数据拥有者计算本地文件的哈希值，接着在自己本地对的本地文件和哈希值进行加密，然后才将数据与哈希值存放在云存储中。这样一来，云存储服务提供商和网络窃听者均无法获取文件的明文，保证了数据的机密性。用户从云存储服务提供商出取回文件和该文件哈希值的密文后对其解密，然后再次对解密后的文件计算哈希值，若与先前存放在云端的哈希值相符，则该文件是完整且未被篡改的，由此保证了数据的完整性。用户可以将自己的文件加密密钥后传递给授权用户，以达到数据共享中访问权限控制的目的。但由于文件加密密钥的传递极有可能破坏系统本身的机密性和完整系，在云存储环境下的安全网盘系统的架构中，如何进行访问权限的控制是最重要的一个环节。

[0006] 现有的云存储环境下的安全网盘系统大致可分为两类：一类是将数据的访问控制权完全交由数据拥有者来进行，其他用户若是想访问数据必须先与数据的拥有者取得联系

进行身份认证，通过身份认证后才能获得密钥与数据的访问权限。这种方式看起来非常的安全合适，却给系统带来了新的弊端：共享用户必须等到共享数据拥有者上线后才能获得访问权限，大大降低了系统的效率，同时对共享数据拥有者的系统造成了额外的开销；另一类是由共享用户自己保存密钥，以外链的方式进行数据共享过程中密钥的传递，也在一定程度上破坏了数据的安全性。

[0007] 本发明实现了一种云存储环境下的安全网盘系统，保证了用户数据的机密性和完整性，同时在数据共享的情况下能够对用户的权限进行有效地管理，为用户提供了高效的数据同步、版本控制等功能。该系统还具有良好的可拓展性。

发明内容

[0008] 本发明的目的在于提供一种云存储环境下的安全网盘系统的系统架构，使得用户在安全网盘系统的信任体系中，即使失去了对系统物理资源的控制也仍然可以安全高效地保证数据及其在共享过程中的机密性、完整性，同时提供数据同步、版本控制等功能。

[0009] 本发明的框架包括：云存储服务提供商、服务器、客户端和网络。其作用分别如下：

[0010] 1) 云存储服务提供商：一般的云存储提供方，通过提供一套 API 提供数据的存取等服务，在本系统中用来保存所有的数据及目录元数据信息；

[0011] 2) 服务器：其作用是保存用户信息，包括用户操作日志，服务器端目录树等，同时作为桥梁将客户端上传的数据保存在云存储中，并在客户端需要时从云存储中将数据取出并返回给客户端；

[0012] 3) 客户端：数据的拥有者，对自己的数据拥有绝对的控制权，同时也是上传、下载、访问权限修改等操作的发起者；

[0013] 4) 网络：数据传输的介质，传输用户的访问请求和文件信息；

[0014] 本发明的思路是：

[0015] 1) 使用开源的云存储服务平台：

[0016] 本云存储环境下的安全网盘系统采用了 OpenStack 的 swift 项目作为系统的底层云存储系统支持。OpenStack 是一个美国国家航空航天局与 Rackspace 共同研发的云端运算软件，以 Apache 许可证授权。Swift 是 OpenStack 的存储子项目，它是一个多副本、可扩展的分布式对象存储系统，可以扩展到 PB 级。在实际应用中，swift 项目具有开源、高可靠性、易于扩展等优点，能够很好的满足安全要素中对底层存储介质的可靠性、可用性要求。

[0017] 2) 数据的机密性和完整性保护：

[0018] 在本系统中，用户在上传数据之前先在本地使用哈希算法——SHA1 算法对文件以块为单位计算哈希值，然后使用自己的文件密钥和加密算法 AES 算法对文件以块为单位进行加密，然后再将数据密文和哈希值上传到云存储区中存储，这样就保证了用户数据存放在云存储中的机密性；用户下载文件时先将数据密文和哈希值下载到本地，然后使用文件密钥对其进行解密并对解密后的数据重新计算哈希值，最后对两个哈希值进行比较来进行数据完整性校验，以保证数据的完整性。

[0019] 3) 使用 PKI 体系对密钥进行层级管理：

[0020] 本系统将密钥分为三层进行管理，理由是利用层级的组织方式对密钥进行组织管

理，可以在保证安全性的同时降低系统开销，同时可以减轻管理员的负担。密钥管理的层次如下：

[0021] i. 文件加密密钥：文件加密密钥是层级密钥中的第一层。文件加密密钥采用用户指定或者随机生成的方式，对于同一用户，所有的数据文件及哈希值均采用此密钥（或由此密钥生成的新密钥）进行加密。在本类系统的信任体系中，用户及被授权共享的用户都是可以信赖的，而服务器（云存储服务提供方）、非授权用户和通信网络都是不可信的。因此，在保证文件加密密钥不被泄露给不可信角色的前提下，这种密钥的管理方式能够带来以下几点好处：第一，密钥数量少，越少则越易于系统管理，同时能够更方便的加强保护机制。第二，文件加密密钥在写进目录元数据的时候存在着一定比例的放大，因此密钥数量的节省能够为服务器节省不少空间；

[0022] ii. 目录元数据：目录元数据是层级密钥中的第二层。为了记录目录的附加属性，我们为每个目录设置了目录元数据。目录元数据与数据一起存放在云端，由服务器端控制，记录了每个目录的目录类型（是否被共享）、目录的绝对路径、目录拥有者、访问权限列表（包括授权用户的用户名和经该用户公钥加密后的数据拥有者的文件密钥对）；

[0023] iii. 用户私钥：用户私钥是层级密钥中的第三层，也是层级密钥管理中的最高一层。用户使用自己的私钥从目录元数据里面的访问控制列表中对应自己的那项解密出文件拥有者的加密密钥，然后再使用解密出的密钥解密出数据的明文及哈希值，然后利用哈希值进行完整性校验。

[0024] 4) 对数据文件提供版本控制：

[0025] 用户上传的小文件的历史版本将在云存储中保存，以方便用户进行历史版本回滚。记录在服务器端的目录树里面包含了版本信息记录，用来对文件的所有历史版本进行管理维护。

[0026] 5) 可选择的数据加密方式：

[0027] 用户数据可以选择每个文件夹的加密方式，包括不加密、直接用文件密钥加密、文件密钥与目录名共同生成加密密钥等方式，可以灵活的调整用户的加密粒度。

[0028] 本发明的特征在于：所述的方法是在是在由客户端计算机、服务器以及云存储服务器共同组成的基于个人用户共享的安全云存储网络中，依次按以下步骤实现的：

[0029] 步骤（1），系统初始化；

[0030] 客户端计算机以下简称客户端，设有数据加解密模块，数据完整性验证模块，本地监听模块，文件夹共享模块以及协议通信模块；

[0031] 服务器端设有：身份验证模块，存储控制模块，访问控制模块，版本控制模块以及目录元数据管理模块，其中：

[0032] 目录元数据管理模块用于对目录元数据进行查询，更新在内的维护操作，所述目录元数据包括：目录类型，拥有者用户名，绝对路径，访问控制列表，授权用户数量以及密钥生成方式，其中访问控制列表包括：用户名和对应的密钥密文，所述密钥密文是使用所对应用户的公钥加密的，对应用户可使用自己的私钥解密；

[0033] 云存储服务器以下简称云存储端，设有：存储模块和数据可靠模块，其中：

[0034] 存储模块设有一套供所述服务器端调用的存储接口，以便把需要存储的信息以对象的形式存放在所述云存储端；

[0035] 数据可靠性模块，根据所述服务器端的需要，为所有存放在云存储端的文件创建一定数量的副本；

[0036] 步骤(2)，为所述云存储端构建并初始化一个云存储平台，步骤如下：

[0037] 步骤(2.1)，在装有 Linux 操作系统的各子云存储中分别安装一个多副本，可扩展的分布式云存储系统 swift，并指定其中的一台机器作为 Proxy 代理节点，重配置 Proxy 代理服务，创建账户，容器以及对象，将其配置成三层环的形式，启动 Proxy 代理服务；

[0038] 步骤(2.2)，配置各云存储服务计算机，实现所述三层环之间的关联；

[0039] 步骤(2.3)，启动所述 swift 系统，提供云存储服务；

[0040] 步骤(3)，用户初始化个人信息，步骤如下：

[0041] 步骤(3.1)，在客户端用户通过调用所述的服务器接口向所述服务器注册新用户；

[0042] 步骤(3.2)，所述服务器端在确认用户注册所用的用户名与密码后，向注册机构申请一个由公钥基础设施所授予的 X509 证书；

[0043] 步骤(3.3)，用户申请步骤(3.2)所述的证书后把其中的公钥保存在所述的服务器端；

[0044] 步骤(3.4)，所述服务器端把包括用户名、密码的哈希值绑定邮箱在内的基本信息保存在数据库中的 users 表中，并创建该用户的服务器端目录树，以文件的形式保存在所述云存储端；

[0045] 步骤(4)，客户端采用一种目录树比对的方式按以下步骤生成一种在客户端目录树、服务器端目录树和磁盘目录树执行同步操作用的操作队列：

[0046] 步骤(4.1)，所述客户端向所述服务器端发送同步请求；

[0047] 步骤(4.2)，所述服务器端在验证当前用户身份后返回用户的最近操作列表，其中包括：此用户在该机器上两次同步之间的这段时间里，该用户在其他机器上对目录或文件所作的删除、移动和重命名操作，其他用户对此用户所共享的目录进行的删除、移动、重命名操作以及当前用户的服务器端目录树；

[0048] 步骤(4.3)，所述用户逐项执行所述服务器端返回的操作列表，完成文件或目录的以上三类操作的同步；

[0049] 步骤(4.4)，所述客户端向服务器请求共享密钥，所述服务器验证身份后根据访问控制列表中的信息将所有该用户被授权的所有文件加密密钥返回给客户端；

[0050] 步骤(4.5)，客户端构造初始化目录树，步骤如下：

[0051] 步骤(4.5.1)，所述客户端读取保存在本地的客户端目录树生成文件，在内存中创建该用户的客户端目录树和组成该客户端目录树的客户端目录树结点：

[0052] 所述客户端目录树是一个二叉树结构，它的格式包括：root、nodes、nodesCount 和 maxnodesCount，其中，

[0053] nodes，以数组方式记录了所述客户端目录树的所有结点；

[0054] nodesCount，记录了所述客户端目录树中的总结点数；

[0055] maxNodesCount，表示所述客户端目录树最多允许包含的结点数；

[0056] 客户端目录树结点的格式包括：nodeType、name、appendAttribute、lastModiftTime、lchild 和 rchild，其中：

[0057] nodeType，记录所述客户端目录树中的结点所对应的是目录还是文件；

[0058] name，记录所述客户端目录树中所述目录或文件的名称；

[0059] appendAttribute，对所述客户端目录树中的目录而言，记录的是是否被共享，对所述客户端目录树中的文件而言，记录的是最新版本的版本号；

[0060] lchild，记录了在所述客户端目录树中该结点的左孩子的索引号；

[0061] rchild，记录了在所述客户端目录树中该结点的右孩子的索引号；

[0062] lastModifytime，记录所述云存储环境下的安全网盘系统 CorsBox 所维护的该结点的最新修改时间；

[0063] 步骤（4.5.2），客户端计算机根据步骤（4.5.1）的返回的服务器端目录树在内存中创建该用户的服务器端目录树，用于记录所述云存储服务器中的实时数据状态，用户服务器端目录树的结构与客户端目录树完全相同，只是 nodes 数组中的结点类型是用户服务器端目录树结点，用户服务器端目录树结点包括：nodeType、name、appendAttribute、lchild 和 rchild，其中：

[0064] nodeType，记录所述用户服务器端目录树中的结点所对应的是目录或文件；

[0065] name，记录所述的目录或文件的名称；

[0066] appendAttribute，对所述用户服务器端目录树中的目录而言，记录的是是否被共享，对所述服务器端目录树中的文件而言，记录的是最新版本的版本号；

[0067] lchild，记录了在所述用户服务器端目录树中该结点的左孩子的索引号；

[0068] rchild，记录了在所述用户服务器端目录树中该结点的右孩子的索引号；

[0069] 步骤（4.5.3），所述客户端计算机在每次同步时扫描磁盘中的所述客户端目录树的目录及其子目录，在内存中创造磁盘目录树记录用户的实时数据，用户的磁盘目录树的结构与客户端目录树完全相同，只是 nodes 数组中的结点类型是磁盘目录树结点，磁盘目录树结点包括：nodeType、name、lastModifyTime、lchild 和 rchild，其中：

[0070] nodeType，记录所述磁盘目录树中的结点所对应的是目录还是文件；

[0071] name，记录所述磁盘目录树中所述目录或文件的名称；

[0072] lastModifytime，记录了在所述客户端计算机中该结点的最后修改时间；

[0073] lchild，记录了在所述磁盘目录树中该结点的左孩子的索引号；

[0074] rchild，记录了在所述磁盘目录树中该结点的右孩子的索引号；

[0075] 步骤（4.6），客户端通过比对步骤（4.5）中构建的目录树，生成待执行操作的操作队列，具体步骤如下：

[0076] 步骤（4.6.1），客户端比对客户端目录树与磁盘目录树，并将比对结果保存在集合 A 中，用来记录用户在此机器上离线进行的操作；

[0077] 步骤（4.6.2），客户端比对客户端目录树与服务器端目录树，并将比对结果保存在集合 B 中，用来记录此用户在此机器上两次同步之间其他客户端进行的操作；

[0078] 步骤（4.6.3），客户端比对集合 A 与集合 B 中的内容，并根据比对结果生成待执行的操作队列；

[0079] 步骤（4.7），逐项执行步骤（4.6）中生成的操作队列，完成同步操作；

[0080] 步骤（5），客户端修改访问权限：

[0081] 步骤（5.1），客户端共享某文件夹：客户端首先检查待共享文件夹的祖先目录与

子孙目录是否被共享，若已被共享，则提示用户此文件夹不能被共享，否则向服务器发起共享请求，服务器收到请求后修改该目录元数据中的共享标志位，然后为此文件夹创建新的操作日志，接着删除该文件夹下所有历史版本，最后向客户端返回共享成功的提示；

[0082] 步骤（5.2），客户端添加 / 删除某用户的访问权限：只有数据的拥有者才能够修改自己共享文件夹的访问权限；若是添加权限，客户端向服务器发起添加权限请求，服务器验证用户身份后将待添加用户的公钥返回给客户端，客户端用此公钥加密该文件夹对应的加密密钥后发送给服务器端，服务器将此密钥的密文写入目录元数据中，并在共享表中添加一条共享记录，最后向客户端返回添加权限成功的提示；

[0083] 若为删除权限，服务器端验证用户身份后将被取消权限用户及其密钥密文从目录元数据中删除，同时删除共享表中对用的记录，然后向客户端返回删除权限成功的提示；

[0084] 步骤（5.3），客户端查看共享内容：客户端查看共享内容依次按照以下步骤来完成：

[0085] 步骤（5.3.1），客户端在启动时向服务器发起查看最新共享请求（可以随时手动请求）；

[0086] 步骤（5.3.2），服务器收到请求后查找共享表，将客户端未选择是否接受共享的文件名称和拥有者 ID 返回给客户端，并在共享表中将这些路径置为已处理；

[0087] 步骤（5.3.3），客户端弹出对话框请求用户操作，若用户选择接受共享则转步骤（4.3.4），否则直接返回；

[0088] 步骤（5.3.4），服务器端将该共享目录下的所有文件密文和该目录的元数据中对应此用户的 ACL 项返回给客户端；

[0089] 步骤（5.3.5），客户端用自己的私钥从返回的 ACL 项解密出该文件夹对应的加密密钥；

[0090] 步骤（5.3.6），客户端解密出所有文件的明文，然后根据明文后面附着的哈希值做完整性校验；

[0091] 步骤（5.4），客户端取消共享：

[0092] 只有文件夹的拥有者才能取消共享，服务器收到验证用户身份后删除共享表中所有关于此文件夹的共享，接着将该文件夹的目录元数据中的共享标志位改为非共享并删除掉所有的 ACL 项，然后向客户端返回修改成功的提示；客户端接收到成功提示后修改客户端目录树中该结点的共享状态；

[0093] 步骤（6）：文件版本控制：

[0094] 步骤（6.1），当客户端上传文件时，服务器首先对该文件的大小进行判断，若是小文件则在文件名末尾添加版本号，并保存在云端；若不是小文件，则直接保存在云端（若已存在则直接覆盖）；

[0095] 步骤（6.2），客户端随时可以进行小文件的历史版本回滚，服务器验证用户身份后返回文件版本列表，用户可以根据自己的需要选择合适的版本进行回滚；

[0096] 步骤（7），用户选择加密方式：

[0097] 客户端会在工作目录下自动生成一个 public 文件夹，所有保存在此文件夹下的内容都是不加密的；另外，用户在客户端可以手动修改某个文件夹（非 public 夹）的密钥生成方法，但一旦某个文件夹的密钥生成方式被指定，其祖先目录与子孙目录都不得再被

指定密钥生成规则；密钥生成是将用户的加密密钥和该文件夹的名称通过一定的算法生成的，若非特别指定则默认直接使用用户的文件加密密钥对其数据进行加密。

[0098] 本发明在清华大学计算机系高性能计算技术研究所进行过测试，测试结果表明，这种云存储环境下的安全网盘系统可以在云存储环境下为用户提供高效的数据同步与文件共享功能的同时，保证了数据的机密性、完整性和访问控制，性能开销也在用户可以接受的范围之内。

附图说明：

[0099] 图 1 系统架构图。

[0100] 图 2 用户上传文件示意图。

[0101] 图 3 用户下载文件示意图。

[0102] 图 4 数据拥有者添加访问权限示意图。

具体实施方式：

[0103] 本发明的具体实施方式如下：

[0104] ➢步骤 1：系统初始化。

[0105] 客户端计算机以下简称客户端，设有数据加解密模块，数据完整性验证模块，本地监听模块，文件夹共享模块以及协议通信模块；

[0106] 服务器端设有：身份验证模块，存储控制模块，访问控制模块，版本控制模块以及目录元数据管理模块，其中：

[0107] 目录元数据管理模块用于对目录元数据进行查询，更新在内的维护操作，所述目录元数据包括：目录类型，拥有者用户名，绝对路径，访问控制列表，授权用户数量以及密钥生成方式，其中访问控制列表包括：用户名和对应的密钥密文，所述密钥密文是使用所对应用户的公钥加密的，对应用户可使用自己的私钥解密；

[0108] 云存储服务器以下简称云存储端，设有：存储模块和数据可靠模块，其中：

[0109] 存储模块设有一套供所述服务器端调用的存储接口，以便把需要存储的信息以对象的形式存放在所述云存储端；

[0110] 数据可靠性模块，根据所述服务器端的需要，为所有存放在云存储端的文件创建一定数量的副本；

[0111] ➢步骤 2：为所述云存储端构建并初始化一个云存储平台。

[0112] ✧步骤（2.1）：在装有 Linux 操作系统的各子云存储中分别安装一个多副本，可扩展的分布式云存储系统 swift，并指定其中的一台机器作为 Proxy 代理节点，重配置 Proxy 代理服务，创建账户，容器以及对象，将其配置成三层环的形式，启动 Proxy 代理服务；

[0113] ✧步骤（2.2）：配置各云存储服务计算机，实现所述三层环之间的关联；

[0114] ✧步骤（2.3）：启动所述 swift 系统，提供云存储服务；

[0115] ➢步骤 3：用户初始化个人信息：

[0116] ✧步骤 3.1：用户在注册的同时申请获得用户标识。用户标识是用户在系统中唯一的身份标识，服务器是通过用户的标识来确定用户的身份，判断其的访问权限；为了安全有效地识别系统中主体（包括服务器和用户）的身份，以便系统对进行操作的用户建立起相互

之间的信任关系，系统需要一种独立于底层存储系统的安全的用户身份标识机制。在本系统中采用公钥基础设施（PKI，Public Key Infrastructure），通过数字证书来为系统提供用户标识。数字证书是由公正、权威的机构签发给主体的电子文档，该文档中记录有主体名称、证书序号、签发方名称、证书的有效期、密码算法标识、公钥信息和其它信息，并经过签发方的数字签名公钥基础设施是包括了硬件、软件、人力、策略和过程的平台或框架，它利用公钥技术提供了对数字证书进行创建、管理、分发、使用、存储以及撤销的功能。证书颁发机构（CA，Cerficate Authority）和注册机构（RA，Registration Authority）是公钥基础设施的重要组成部分。前者是公钥基础设施的核心，它是一个可信的第三方，通过将用户的公钥与用户的其他信息（包括用户身份）绑定在一起来为用户签发数字证书，并提供证书的查询、撤销、生命周期管理以及密钥管理；后者主要是面向用户履行证书颁发机构委派的一些责任。公钥基础设施是一种成熟的、被广泛应用的技术体系，具有统一的规范和标准，并有很多较为完备的实现。利用公钥基础设施为系统提供用户标识，可以将维护用户标识唯一性真实性的工作交给这个成熟的体系来完成，同时使系统用户在不必了解复杂管理细节的情况下安全高效地验证其他系统主体的身份，实现用户之间的相互信任，从而保证用户信息的真实性、完整性、机密性和不可否认性；

[0117] ✧步骤 3.2：服务器将用户的基本信息（包括用户名、密码的哈希值、绑定邮箱等）保存至数据库中的 users 表中；

[0118] ✧步骤 3.3：客户端初始化工作目录，并在工作目录下生成 shares、public 以及 public/shares 三个文件夹，其中 shares 文件夹下的内容其他用户的共享内容（加密），public 夹下的内容为自己不加密上传的文件、public/shares 为其他用户共享的非加密内容；

[0119] ✧步骤 3.4：客户端为当前用户生成初始的客户端目录树，然后将它持久化到文件中，并在配置文件中标明该文件所在的位置；

[0120] ✧步骤 3.5：服务器初始化此用户的服务器端目录树，然后将它持久化到文件中，并将此文件按照一定的命名规则保存至云存储中；

[0121] ➢步骤 4：客户端执行同步操作。客户端使用一种基于目录树的同步方式，执行同步操作生成带操作列表，然后根据列表中每个元素中记录的待执行操作类型逐一执行同步操作：

[0122] ●若待执行操作为用户上传文件，用户则按照如下步骤上传文件，具体如图 2 所示：

[0123] ✧步骤①：客户端使用某种哈希算法计算待上传文件的哈希值；

[0124] ✧步骤②：客户端使用相应的加密密钥对文件和哈希值进行加密；

[0125] ✧步骤③：客户端将文件密文和加密后的哈希值拼接成一个文件，然后向服务器发起上传请求；

[0126] ✧步骤④：服务器验证用户身份后将客户端上传的数据以最新版本的方式（小文件）存放在云存储中，并向客户端返回上传成功；

[0127] ●若待执行操作为用户下载文件，用户按照如下步骤下载文件，具体如图 3 所示：

[0128] ✧步骤①：用户向服务器发起某个文件的下载请求；

[0129] ✧步骤②：服务器验证用户身份后将该文件从云存储中取出并返回给客户端；

[0130] ✧步骤③：客户端使用相应的密钥对服务器返回的文件进行解密，然后根据解密出的文件分离出数据文件和哈希值，然后将数据文件保存在本地；

[0131] ✧步骤④：客户端计算数据文件的哈希值，并将其与下载下来的哈希值做比对，进行数据的完整性校验；

[0132] ●若待执行操作为客户端冲突处理，则说明此文件同时被多个客户端修改，那么客户端需要对该文件进行冲突处理操作，具体步骤如下：

[0133] ✧步骤①：用户将本地文件重命名；

[0134] ✧步骤②：客户端调用下载接口下载服务器端的最新版文件；

[0135] ✧步骤③：客户端调用上传接口上传本地重命名后的文件；

[0136] ✧步骤④：客户端可以根据需要选择删除或重命名冲突文件；

[0137] ➢步骤 5：客户端修改访问权限。客户端添加访问权限可以分为以下两类：

[0138] ●用户添加访问权限：只有数据拥有者才能按照如下步骤添加访问权限，具体如图 4 所示：

[0139] ✧步骤①：客户端向服务器取得待授权用户的公钥；

[0140] ✧步骤②：用户可以选择是否向证书颁发机构校验证书的真实性；

[0141] ✧步骤③：客户端使用该公钥加密自己的文件密钥；

[0142] ✧步骤④：客户端将此文件密钥上传至服务器，服务器验证权限后写入该共享目录的目录元数据，并在共享表中添加共享记录；

[0143] ●用户撤销访问权限：只有数据拥有者才能撤销访问权限，具体步骤如下：

[0144] ✧步骤①：客户端向服务器发起撤销请求；

[0145] ✧步骤②：服务器验证用户权限后删除该共享目录对应的目录元数据中的 ACL 项，同时删除共享表中的记录。

[0146] ➢步骤 6：文件版本控制：

[0147] ✧步骤（5.1）：当客户端上传文件时，服务器首先对该文件的大小进行判断，若是小文件则在文件名末尾添加版本号，并保存在云端；若不是小文件，则直接保存在云端（若已存在则直接覆盖）；

[0148] ✧步骤（5.2）：客户端随时可以进行小文件的历史版本回滚，服务器验证用户身份后返回文件版本列表，用户可以根据自己的需要选择合适的版本进行回滚；

[0149] ➢步骤 7：用户选择加密方式：

[0150] ✧客户端会在工作目录下自动生成一个 public 文件夹，所有保存在此文件夹下的内容都是不加密的；另外，用户在客户端可以手动修改某个文件夹（非 public 夹）的密钥生成方法，但一旦某个文件夹的密钥生成方式被指定，其祖先目录与子孙目录都不得再被指定密钥生成规则；密钥生成是将用户的加密密钥和该文件夹的名称通过一定的算法生成的，若非特别指定则默认直接使用用户的文件加密密钥对其数据进行加密。

[0151] 本发明的系统架构如图 1 所示，发明的核心是提出了一种云存储环境下的安全网盘系统，其实现主要是由以下几个部分以及其相应的模块组成：

[0152] ●云存储端

[0153] 云存储端主要由以下几个模块构成：

[0154] 1. 存储模块

[0155] 本模块提供了一套存储接口，由服务器端调用这些接口，将需要存储的信息以对象的形式存放在云存储端。

[0156] 2. 数据可靠性模块

[0157] 本模块为所有存放在云端的文件创建了 N 个副本（N 由服务器端根据需要指定），保证了数据的可靠性。

[0158] ●客户端

[0159] 客户端主要由以下几个模块构成：

[0160] 1. 数据加解密模块

[0161] 该模块承担了所有的加解密相关的密码学操作操作，包括使用文件块密钥加解密文件块等，使用公私钥加解密文件密钥等。

[0162] 2. 数据完整性验证模块

[0163] 该模块提供文件块内容完整性验证等操作，以及文件块内容哈希值计算等操作。

[0164] 3. 本地监听模块

[0165] 本模块用来监听用户对本地工作目录内进行的修改，包括删除、移动与重命名等操作。

[0166] 4. 文件夹共享模块

[0167] 本模块提供了所有用户共享文件夹的操作，包括请求文件夹、修改访问权限和撤销共享等操作。

[0168] 5. 协议通信模块

[0169] 本模块是客户端最大的一个模块，它包含了客户端所有的涉及到通信的操作，主要有上传、下载、冲突处理、注册新用户等。

[0170] ●服务器

[0171] 1. 身份认证模块

[0172] 本模块用来验证用户的身份是否跟其在注册中心注册的相符，以防止网络黑手或其他非法人员伪造用户身份。

[0173] 2. 存储控制模块

[0174] 本模块调用云存储接口，将用户的服务器端目录树、目录元数据和数据文件保存至云端。

[0175] 3. 访问控制模块

[0176] 根据用户权限对数据进行访问权限的控制，但客户端并不依赖于服务器的访问控制，即使服务器恶意的将数据交给非法用户也无法读取到数据明文。

[0177] 4. 版本控制模块

[0178] 服务器对用户上传的小文件进行了版本控制，允许用户将某个小文件回滚到某个历史版本。本模块包含了所有版本控制相关的操作。

[0179] 5. 目录元数据管理模块

[0180] 本模块用来对目录元数据进行维护，包括查询、更新等操作。目录元数据在内存中的结构如下：

[0181]

目录类型	拥有者 ID	绝对路径	访问控制列表	授权用户数量	密钥生成方式

[0182] 访问控制列表(ACL)的结构如下：

[0183]

用户名	密钥密文
用户名 1	密钥密文
用户名 2	密钥密文
…	密钥密文
用户名 n	密钥密文

[0184] 其中每个用户对应的密钥密文都是使用该用户的公钥进行加密的，用户可以使用自己的私钥对其进行解密。

[0185] 系统测试

[0186] 本发明在清华大学计算机科学与技术系高性能计算研究所对本系统进行了测试，其内容包括功能测试和性能测试，分别对系统的安全性和性能进行了测试，测试结果表明，本发明在云存储共享的信任体系中能够保证用户数据的安全性，同时性能开销在一个可以接受的范围。

[0187] ●功能测试

[0188] 我们使用了十台服务器对本系统进行了功能测试。其中四台服务器用来部署云存储端，三台服务器用来部署服务器端，另外三台服务器作为客户端，分别服务器并进行操作。测试内容与测试结果如下表所示：

[0189]

测试内容	测试结果
数据机密性保护：服务器绕过本系统，直接调用云端接口查看用户数据内容	数据内容为加密后的乱码，无法得知被查看数据明文。
数据完整性校验：服务器绕过本系统篡改用户数据。	系统弹出数据已遭到破坏的警告。
权限管理：测试某个用户在其被授予某个共享目录的使用权限之前，被授予权限之后，以及被撤销权限之后的结果。	被授予权限前，用户看不到其他用户的任何数据。 被授予权限后，用户可以查看并使用所有共享给自己的数据。 被撤销权限后，用户的共享文件被从共享夹中移除，作为自己的本地文件保存在磁盘中。
文件版本控制：测试某个用户将某个文件回滚到某个历史版本。	文件版本回滚成功。

[0190] ●性能测试

[0191] 性能测试的硬件环境是七台配置相同服务器，每台服务器均采用了型号为 Intel(R)Xeon(R)X5472，主频为 3.0GHz 的四核 CPU，8GB 内存，服务器间以千兆局域网连接。其中四台做云存储服务器，三台做 CorsBox 服务器。软件环境是 Ubuntu Linux 2.6.33 内核，openstack-swift 1.3.0，bcprov-jdk16-146.jar。客户端的硬件环境为 :Intel(R) Core(TM)2, 主频为 2.40GHz 的双核 CPU，2GB 内存，软件环境为 Windows 7 操作系统。

[0192] 1. 上传操作

[0193] 如图 2 所示，客户端向服务器上传一个 64M 大小的文件，各项操作的时间消耗如下（单位均为 ms）：

[0194]

操作步骤	16M 文件	32M 文件	64M 文件
本地读文件的时间	91	123	1003
计算文件哈希值	0	1	1
加密文件及哈希值	965	1185	2266

上传至服务器	9055	15636	31255
服务器保存至云端	2100	2663	3801
总时间	12211	19608	38326

[0195] 从消耗时间上来看，计算哈希值的时间在 1 毫秒以内，可以忽略不计；密码学相关操作为磁盘开销增加了大约 45% 的负担，但是，所有的对数据的加解密都是以上传或下载为前提的，而客户端与服务器之间是通过普通不可信网络连接的，因此密码学为整个 I/O 过程额外增加了大约 6% 的额外负担，这个开销完全是在用户可接受范围内的。

[0196] 2. 下载操作

[0197] 如图 3 所示，客户端从服务器下载一个 64M 大小的文件，各项操作的时间消耗如下（单位均为 ms）：

[0198]

操作步骤	16M 文件	32M 文件	64M 文件
服务器从云端读取数据	202	385	603
客户端从服务器下载数据	3765	6422	10703
客户端解密文件	1148	1285	2245
客户端进行完整性校验	0	0	0
写入磁盘	44	411	823
总时间	5159	8503	14374

[0199] 从额外的消耗来看，密码学为整个 I/O 过程额外增加了约 15% 的开销。下载过程明显比上传要快，这是因为我们对编码方式进行了优化，使用了客户端几倍于文件的内存空间大小还换取时间上的效率。

[0200] 3. 添加权限操作

[0201] 如图 4 所示，客户端将其共享目录 photo 授权给用户 B，各项操作的时间消耗如下（单位均为 ms）：

[0202]

操作步骤	消耗时间（ms）
客户端获取用户 B 公钥	2496
客户端使用公钥加密文件密钥	450
客户端将密钥上传至服务器	1386

服务器更新目录元数据	6
总时间	4338

[0203] 4. 撤销权限操作

[0204] 客户端撤销用户 B 对目录 photo 的访问权限，各项操作的时间消耗如下（单位均为 ms）：

[0205]

操作步骤	操作时间（ms）
请求时间	654
服务器操作时间	1166
总时间	1712

[0206] 由此可以看出在本发明中修改用户访问权限操作的时间开销较少，具有良好的高效性。

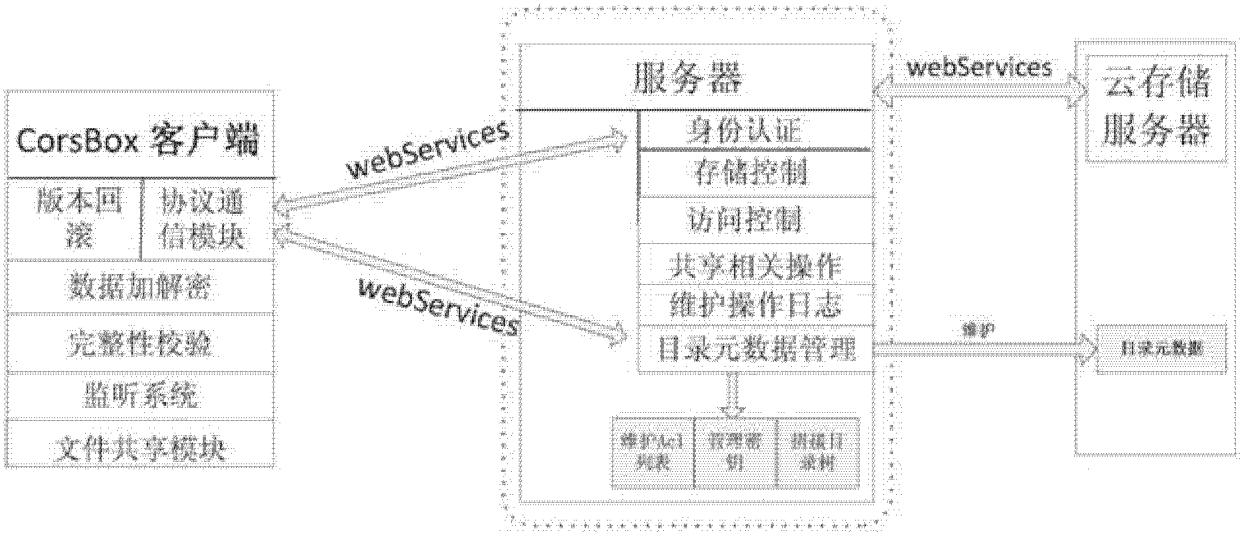

CorsBox 客户端
版本回滚
协议通信模块
数据加解密
完整性校验
监听系统
文件共享模块
webServices
webServices
服务器
身份认证
存储控制
访问控制
共享相关操作
维护操作日志
目录元数据管理
webServices
云存储服务器
维护
目录元数据

图 1

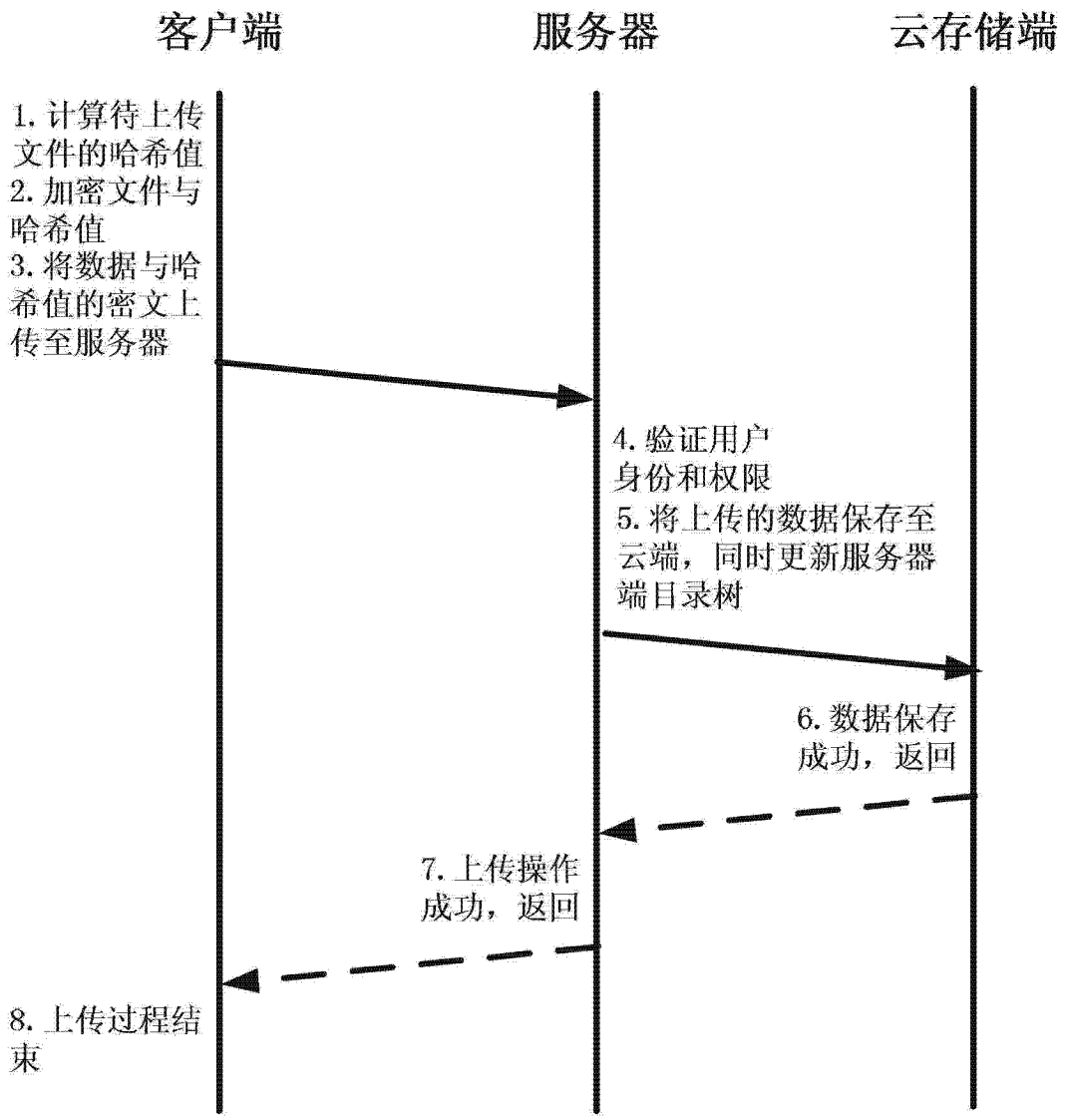

客户端
服务器
云存储端
1. 计算待上传文件的哈希值
2. 加密文件与哈希值
3. 将数据与哈希值的密文上传至服务器
4. 验证用户身份和权限
5. 将上传的数据保存至云端，同时更新服务器端目录树
6. 数据保存成功，返回
7. 上传操作成功，返回
8. 上传过程结束

图 2

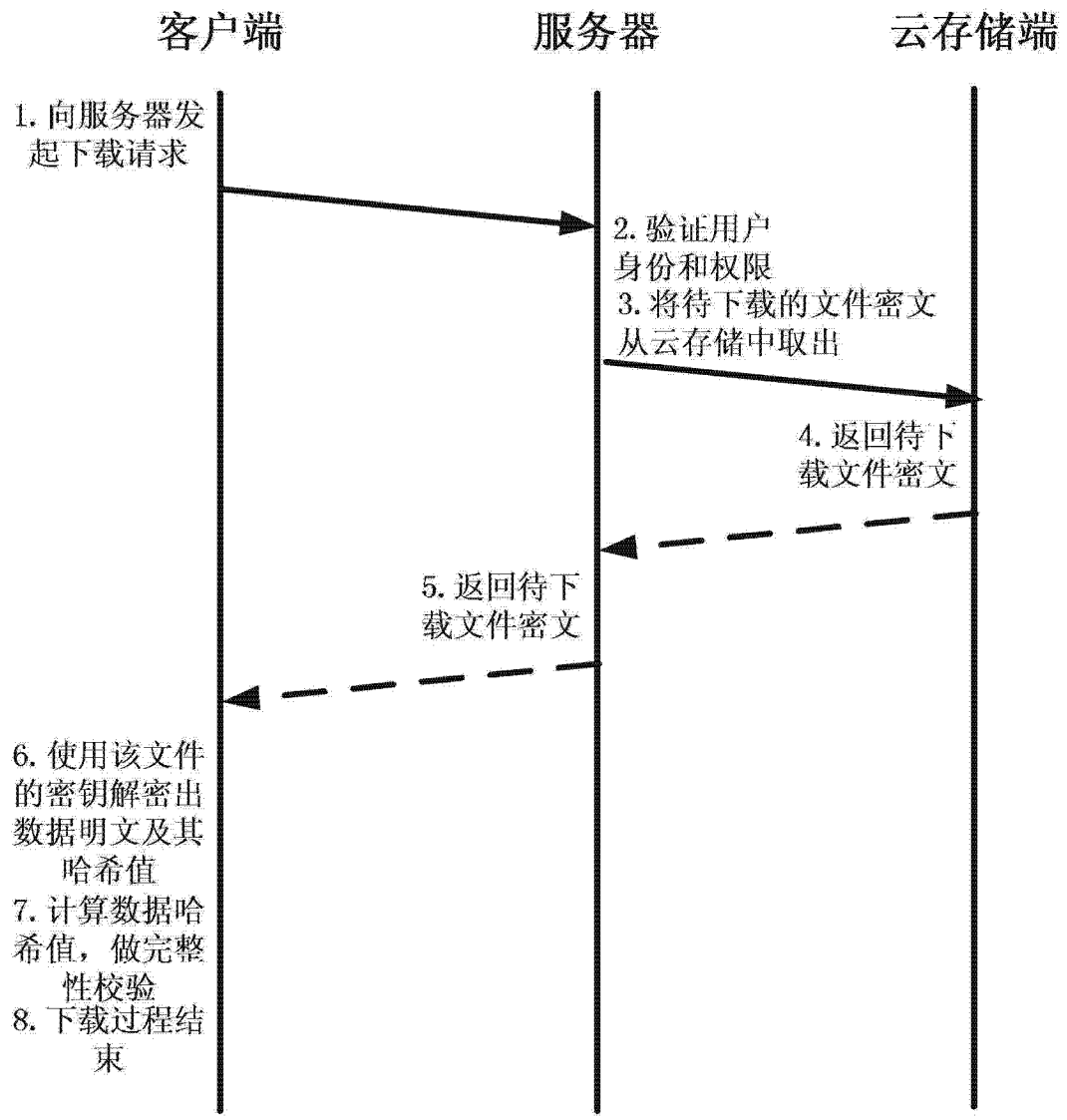

客户端
服务器
云存储端
1. 向服务器发起下载请求
2. 验证用户身份和权限
3. 将待下载的文件密文从云存储中取出
4. 返回待下载文件密文
5. 返回待下载文件密文
6. 使用该文件的密钥解密出数据明文及其哈希值
7. 计算数据哈希值，做完整性校验
8. 下载过程结束

图 3

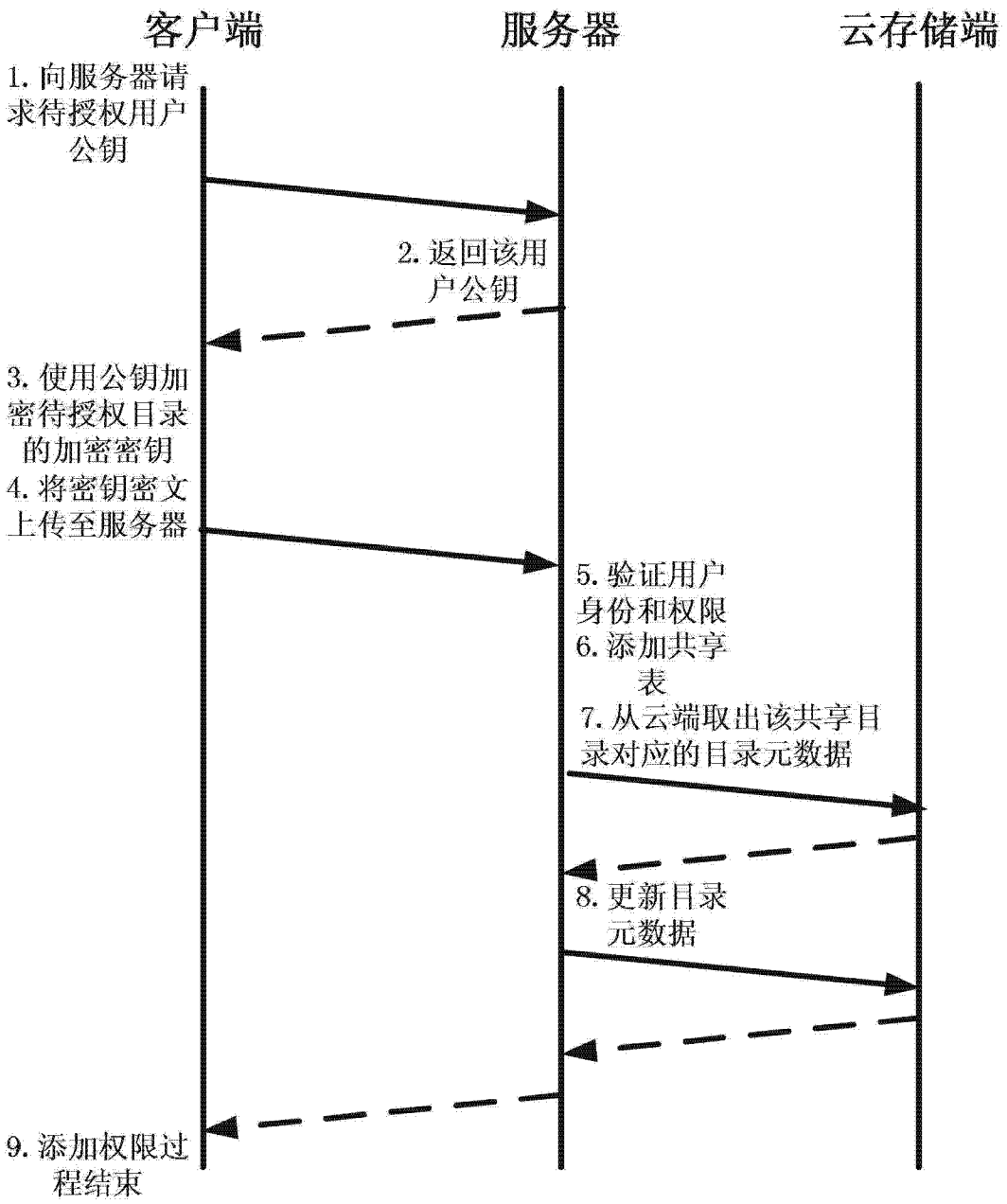

客户端
服务器
云存储端
1. 向服务器请求待授权用户公钥
2. 返回该用户公钥
3. 使用公钥加密待授权目录的加密密钥
4. 将密钥密文上传至服务器
5. 验证用户身份和权限
6. 添加共享表
7. 从云端取出该共享目录对应的目录元数据
8. 更新目录元数据
9. 添加权限过程结束

图 4