



(12) 发明专利

(10) 授权公告号 CN 102394894 B

(45) 授权公告日 2014.01.15

(21) 申请号 201110383355.X

(22) 申请日 2011.11.28

(73) 专利权人 武汉大学

地址 430072 湖北省武汉市武昌区珞珈山武汉大学

(72) 发明人 陈晶 郑明辉 杜瑞颖 傅建明
李彤

(74) 专利代理机构 武汉科皓知识产权代理事务所(特殊普通合伙) 42222

代理人 薛玲

(51) Int. Cl.

H04L 29/06(2006.01)

H04L 29/08(2006.01)

(56) 对比文件

CN 101840346 A, 2010.09.22,

CN 102034046 A, 2011.04.27,

审查员 杨群

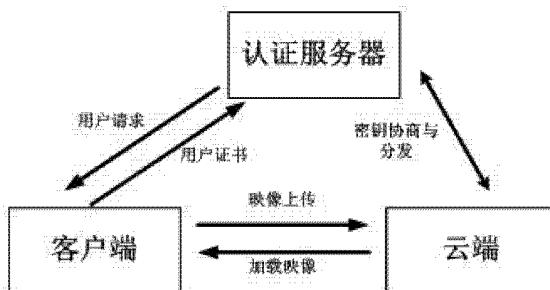
权利要求书3页 说明书13页 附图6页

(54) 发明名称

一种基于云计算的网络虚拟磁盘文件安全管理方法

(57) 摘要

本发明提出一种基于云计算的网络虚拟磁盘文件安全管理方法，采用云计算的思想，实现了自动负载均衡及透明扩容功能。为了克服公有云无法实现数据安全隔离和授权使用的弱点，本方法可以以 Hadoop 云平台为基础，构建无缝式的虚拟磁盘透明加密环境，实现分布存储、数据隔离、安全的数据共享等功能。



1. 一种基于云计算的网络虚拟磁盘文件安全管理方法,其特征在于:设置认证服务器,认证服务器负责客户端和云端交互过程中的身份认证与密钥分发,认证服务器为每个映像文件分配一个文件密钥M1并保存,认证服务器将映像文件的文件密钥M1传输到客户端时,用会话密钥M2对文件密钥M1进行加密后传输,客户端接收后采用会话密钥M2解密得到文件密钥M1;

在用户使用客户端过程中,实现网络虚拟磁盘文件的安全管理包括以下步骤,

步骤1,当用户从客户端输入用户名和密码时,客户端首先将用户名和密码采用SHA-2哈希函数进行处理,再将处理所得哈希值用认证服务器的公钥加密后发送至认证服务器,由认证服务器进行身份认证;当认证通过时进入步骤2,未通过时在客户端提示用户的用户名或密码不正确;

步骤2,进行系统初始化,首先初始化用户空间信息和用户权限信息,用户空间信息为用户在认证服务器存放的映像文件的相关信息,包括各映像文件的状态是否自动加载;然后,从云端下载状态为自动加载的映像文件并加载为虚拟磁盘,具体方式如下,

客户端向认证服务器发送加载该映像文件的请求,认证服务器接收到加载请求后,通知云端将映像文件传送到客户端,然后认证服务器将该映像文件的文件密钥M1传输到客户端;客户端用文件密钥M1对映像文件进行解密,解密完成后将该映像文件加载为虚拟磁盘;

步骤3,对用户在其用户映像空间的操作进行管理,用户在其用户映像空间的操作包括创建映像文件、加载映像文件、卸载虚拟磁盘、修改用户权限和浏览用户空间信息,

当用户创建映像文件时,输入待创建的映像文件保存路径、文件名以及文件大小后,客户端首先向认证服务器进行创建映像文件的申请,认证服务器接收到申请后,为待创建的映像文件分配一个随机的文件密钥M1并传输到客户端;

当用户加载映像文件时,对于该用户新创建的映像文件,客户端直接将该映像文件加载成虚拟磁盘;对于该用户先前创建的映像文件,客户端从云端下载加密的映像文件,并向认证服务器申请获取对应的文件密钥M1进行解密,解密完成后将该映像文件加载为虚拟磁盘;当用户加载其它用户的映像文件时,客户端从云端下载加密的映像文件,并向认证服务器申请对应的文件密钥M1,认证服务器查判断用户权限是否合法,若合法则将文件密钥M1传输到客户端,客户端用文件密钥M1对映像文件进行解密,解密完成后将该映像文件加载为虚拟磁盘,若不合法则认证服务器拒绝请求;

当用户卸载虚拟磁盘时,客户端将虚拟磁盘从资源管理器中卸载,将虚拟磁盘里面的内容更新到映像文件中去,并用对应的文件密钥M1对虚拟磁盘的映像文件进行加密,将加密后的映像文件上传到云端,同时将映像文件的相关信息上传到认证服务器;

当用户修改用户权限时,客户端将修改后的用户权限信息上传到认证服务器;

当用户浏览用户空间信息时,客户端向用户显示初始化用户空间信息的结果;

当用户退出时,客户端卸载所有加载的虚拟磁盘,将每个虚拟磁盘的映像文件分别用对应的文件密钥M1进行加密,将加密后的映像文件上传到云端,同时将所有映像文件的相关信息上传到认证服务器。

2. 如权利要求1所述基于云计算的网络虚拟磁盘文件安全管理方法,其特征在于:步骤1中身份认证具体实现方式包括以下步骤,

①客户端记为 A, 认证服务器记为 B ; 客户端 A 将自己的身份信息 ID_A 传递给认证服务器 B ;

②认证服务器 B 收到身份信息 ID_A 后, 产生一个随机的消息 R_B , 用客户端 A 的公钥 P_A 加密身份信息 ID_A 和消息 R_B 得到 $y_1 = \mathcal{E}_{P_A}(ID_A \parallel R_B)$, 其中 $\mathcal{E}_{P_A}(\cdot)$ 表示用公钥 P_A 进行的加密过程 ; 并用自己的私钥 S_B 计算得到签名 $y_2 = \mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))$, 其中 $\mathcal{D}_{S_B}(\cdot)$ 表示用私钥 S_B 进行的解密过程 ; 将加密结果 $\mathcal{E}_{P_A}(ID_A \parallel R_B)$ 和签名 $\mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))$ 传送给客户端 A ;

③客户端 A 收到消息后用认证服务器 B 的公钥 P_B 对签名 $\mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))$ 进行验证 , 验证方式为判断等式 $\mathcal{E}_{P_B}(\mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))) = \mathcal{E}_{P_A}(ID_A \parallel R_B)$ 是否成立, 其中 $\mathcal{E}_{P_B}(\cdot)$ 表示用公钥 P_B 进行的加密过程 ; 如果等式成立, 客户端 A 确认通信对方为认证服务器 B, 并对 $\mathcal{E}_{P_A}(ID_A \parallel R_B)$ 进行解密, 解密($\mathcal{E}_{P_A}(ID_A \parallel R_B)$)= $ID_A \parallel R_B$, 再分离出 ID_A 和 R_B ;

④客户端 A 将步骤③求得的 R_B 用认证服务器 B 的公钥 P_B 加密得到 $\mathcal{E}_{P_B}(R_B)$, 将 $\mathcal{E}_{P_B}(R_B)$ 传送给认证服务器 B, 认证服务器 B 用自己的私钥 S_B 解密 $\mathcal{E}_{P_B}(R_B)$ 即可得到 R_B , 将此 R_B 与原来在步骤②随机产生的 R_B 对比, 确认对方是否是意定的客户端 A。

3. 如权利要求 1 或 2 所述基于云计算的网络虚拟磁盘文件安全管理方法, 其特征在于 : 用户权限包括浏览、加载及加载受限 ;

(1) 浏览 : 如果用户 a 将该权限授予用户 b, 用户 b 用自己的账号登录后, 用户 b 的用户映像空间里提供用户 a 在认证服务器存放的映像文件相关信息, 但是不提供下载该映像文件 ;

(2) 加载 : 如果用户 a 将该权限授予用户 b, 则用户 b 用自己的账号登录后, 用户 b 的用户映像空间提供用户 a 在云端存放的映像文件相关信息, 并支持下载该映像文件和加载为虚拟磁盘, 但用户 b 对该虚拟磁盘的修改不更新到云端对应的映像文件中 ;

(3) 加载受限 : 如果用户 a 将该权限授予用户 b, 则用户 b 用自己的账号登录后, 用户 b 的用户映像空间提供用户 a 在云端存放的映像文件相关信息, 并支持下载该映像文件和加载为虚拟磁盘, 但用户 b 访问该虚拟磁盘的拷贝、截屏、另存为和打印操作都进行了限制。

4. 如权利要求 3 所述基于云计算的网络虚拟磁盘文件安全管理方法, 其特征在于 : 在用户对其他用户授予用户权限时, 由相关负责人登录客户端审批, 并提交到认证服务器进行记录。

5. 如权利要求 1 或 2 所述基于云计算的网络虚拟磁盘文件安全管理方法, 其特征在于 : 认证服务器保存文件密钥 M1 时, 采用管理密钥 M3 进行加密, 加密结果记为 M4 ; 将映像文件的文件密钥 M1 传输到客户端时, 先用管理密钥 M3 对加密结果 M4 进行解密得到文件密钥 M1, 然后用会话密钥 M2 对文件密钥 M1 进行加密后传输。

6. 如权利要求 3 所述基于云计算的网络虚拟磁盘文件安全管理方法, 其特征在于 : 认证服务器保存文件密钥 M1 时, 采用管理密钥 M3 进行加密, 加密结果记为 M4 ; 将映像文件的文件密钥 M1 传输到客户端时, 先用管理密钥 M3 对加密结果 M4 进行解密得到文件密钥 M1, 然后用会话密钥 M2 对文件密钥 M1 进行加密后传输。

7. 如权利要求 4 所述基于云计算的网络虚拟磁盘文件安全管理方法, 其特征在于 : 认

证服务器保存文件密钥 M1 时,采用管理密钥 M3 进行加密,加密结果记为 M4 ;将映像文件的文件密钥 M1 传输到客户端时,先用管理密钥 M3 对加密结果 M4 进行解密得到文件密钥 M1,然后用会话密钥 M2 对文件密钥 M1 进行加密后传输。

一种基于云计算的网络虚拟磁盘文件安全管理方法

技术领域

[0001] 本发明涉及计算机信息安全领域,尤其是涉及一种基于云计算的网络虚拟磁盘文件安全管理方法。

背景技术

[0002] 互联网的高速发展使我们能够浏览网上的海量资源,但与此同时也使我们的本地数据暴露在互联网上。在很多商业项目中,企业对数据的安全性要求非常高,尤其不希望企业开发的源代码、设计图纸、客户资料、研究成果和公司的销售记录等重要数据被内部人员泄露出去,或者流传到竞争对手手里。而据美国 FBI 统计,83% 的信息安全事故为内部人员和内外勾结所为,70% 的泄密犯罪来自于企业内部,数据安全存储与共享问题不容忽视。

[0003] 为此,很多公司会采用禁止员工携带可移动设备、禁止公司的电脑接入外网等各种各样的方法来防止机密资料外泄。但这并不够方便,也不十分有效,公司的机密文件泄露事件还是时有发生。

[0004] 为了保证企业内部信息数据的安全共享,又能提高工作效率,目前主要有以下五种手段。然而他们都存在一定的缺陷:

[0005] 1. 杀毒软件、防火墙、入侵检测等外网安全系统都是基于外部安全模型的,无法阻止内部人员的泄密,也无法防范新的病毒和漏洞。

[0006] 2. 主动型文件和文件夹加密系统。文件由用户主动加密,然而在密码传输过程中同样存在泄密问题,且文档创造者可在文件加密处理之前给自己留下拷贝,因而此方法还是防不住内部人员的主动泄密。

[0007] 3. 网络监控与审计系统。企业中计算机管理人员对每一台涉密计算机进行监控,其基本思想在于“堵漏洞”。然而用户常常为了必要的、正常的工作交流而留一些“口子”,从而大大降低系统的可靠性;否则只能堵死所有的“漏洞”,以牺牲方便性为代价来换取严格的安全性。

[0008] 4. 文件权限集中管理系统。该系统管理的直接是电子文档数据本身,可在一定程度上从源头上保证电子文档的安全,然而同样无法真正防止内部人员的主动泄密,需要更多的投资于服务器和周边子系统,需要庞大的数据库做后台支持。

[0009] 5. 强制文件加解密系统。该系统对涉密文件全部进行加密,但软件出错或者在停电等异常情况下,增大了文件损坏的几率,并且一般文件损坏就无法修复,对安全系统自身的稳定性和可靠性要求极高。

[0010] 针对上述问题,提出一种方便有效的网络虚拟磁盘文件安全管理方法,是急需解决的重大难题。

发明内容

[0011] 本发明提出了一种基于云计算的网络虚拟磁盘文件安全管理方法,其目的在于充分利用云平台分布式文件管理以及集群分发调度的功能特性,高效方便地实现数据的安全

共享。

[0012] 本发明的技术方案为一种基于云计算的网络虚拟磁盘文件安全管理方法，设置认证服务器，认证服务器负责客户端和云端交互过程中的身份认证与密钥分发，认证服务器为每个映像文件分配一个文件密钥 M1 并保存，认证服务器将映像文件的文件密钥 M1 传输到客户端时，用会话密钥 M2 对文件密钥 M1 进行加密后传输，客户端接收后采用会话密钥 M2 解密得到文件密钥 M1；

[0013] 在用户使用客户端过程中，实现网络虚拟磁盘文件的安全管理包括以下步骤，

[0014] 步骤 1，当用户从客户端输入用户名和密码时，客户端首先将用户名和密码采用 SHA-2 哈希函数进行处理，再将处理所得哈希值用认证服务器的公钥加密后发送至认证服务器，由认证服务器进行身份认证；当认证通过时进入步骤 2，未通过时在客户端提示用户的用户名或密码不正确；

[0015] 步骤 2，进行系统初始化，首先初始化用户空间信息和用户权限信息，用户空间信息为该用户在认证服务器存放的所有映像文件的相关信息，包括各映像文件的状态是否自动加载；然后，从云端下载状态为自动加载的映像文件并加载为虚拟磁盘，具体方式如下，

[0016] 客户端向认证服务器发送加载该映像文件的请求，认证服务器接收到加载请求后，通知云端将映像文件传送到客户端，然后认证服务器将该映像文件的文件密钥 M1 传输到客户端；客户端用文件密钥 M1 对映像文件进行解密，解密完成后将该映像文件加载为虚拟磁盘；

[0017] 步骤 3，对用户在其用户映像空间的操作进行管理，用户在其用户映像空间的操作包括创建映像文件、加载映像文件、卸载虚拟磁盘、修改用户权限和浏览用户空间信息，

[0018] 当用户创建映像文件时，输入待创建的映像文件的保存路径、文件名以及文件大小后，客户端首先向认证服务器进行创建映像文件的申请，认证服务器接收到申请后，为待创建的映像文件分配一个随机的文件密钥 M1 并传输到客户端；

[0019] 当用户加载映像文件时，对于该用户新创建的映像文件，客户端直接将该映像文件加载成虚拟磁盘；对于该用户先前创建的映像文件，客户端从云端下载加密的映像文件，并向认证服务器申请获取对应的文件密钥 M1 进行解密，解密完成后将该映像文件加载为虚拟磁盘；当用户加载其它用户的映像文件时，客户端从云端下载加密的映像文件，并向认证服务器申请对应的文件密钥 M1，认证服务器查判断用户权限是否合法，若合法则将文件密钥 M1 传输到客户端，客户端用文件密钥 M1 对映像文件进行解密，解密完成后将该映像文件加载为虚拟磁盘，若不合法则认证服务器拒绝请求；

[0020] 当用户卸载虚拟磁盘时，客户端将虚拟磁盘从资源管理器中卸载，将虚拟磁盘里面的内容更新到映像文件中去，并用对应的文件密钥 M1 对虚拟磁盘的映像文件进行加密，将加密后的映像文件上传到云端，同时将映像文件的相关信息上传到认证服务器；

[0021] 当用户修改用户权限时，客户端将修改后的用户权限信息上传到认证服务器；

[0022] 当用户浏览用户空间信息时，客户端向用户显示初始化用户空间信息的结果；

[0023] 当用户退出时，客户端卸载所有加载的虚拟磁盘，将每个虚拟磁盘的映像文件分别用对应的文件密钥 M1 进行加密，将加密后的映像文件上传到云端，同时将所有映像文件的相关信息上传到认证服务器。

[0024] 而且，步骤 1 中身份认证具体实现方式包括以下步骤，

[0025] ①客户端记为 A, 认证服务器记为 B; 客户端 A 将自己的身份信息 ID_A 传递给认证服务器 B;

[0026] ②认证服务器 B 收到身份信息 ID_A 后, 产生一个随机的消息 R_B , 用客户端 A 的公钥 P_A 加密身份信息 ID_A 和消息 R_B 得到 $y_1 = \mathcal{E}_{P_A}(ID_A \parallel R_B)$, 其中 $\mathcal{E}_{P_A}(\cdot)$ 表示用公钥 P_A 进行的加密过程; 并用自己的私钥 S_B 计算得到签名 $y_2 = \mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))$, 其中 $\mathcal{D}_{S_B}(\cdot)$ 表示用私钥 S_B 进行的解密过程; 将加密结果 $\mathcal{E}_{P_A}(ID_A \parallel R_B)$ 和签名 $\mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))$ 传送给客户端 A;

[0027] ③客户端 A 收到消息后用认证服务器 B 的公钥 P_B 对签名 $\mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))$ 进行验证, 验证方式为判断等式 $\mathcal{E}_{P_B}(\mathcal{D}_{S_B}(\mathcal{E}_{P_A}(ID_A \parallel R_B))) = \mathcal{E}_{P_A}(ID_A \parallel R_B)$ 是否成立, 其中 $\mathcal{E}_{P_B}(\cdot)$ 表示用公钥 P_B 进行的加密过程; 如果等式成立, 客户端 A 确认通信对方为认证服务器 B, 并对 $\mathcal{E}_{P_A}(ID_A \parallel R_B)$ 进行解密, 解密 $(\mathcal{E}_{P_A}(ID_A \parallel R_B)) = ID_A \parallel R_B$, 再分离出 ID_A 和 R_B ;

[0028] ④客户端 A 将步骤③求得的 R_B 用认证服务器 B 的公钥 P_B 加密得到 $\mathcal{E}_{P_B}(R_B)$, 将 $\mathcal{E}_{P_B}(R_B)$ 传送给认证服务器 B, 认证服务器 B 用自己的私钥 S_B 解密 $\mathcal{E}_{P_B}(R_B)$ 即可得到 R_B , 将此 R_B 与原来在步骤②随机产生的 R_B 对比, 确认对方是否是意定的客户端 A。

[0029] 而且, 用户权限包括浏览、加载及加载受限;

[0030] (1) 浏览: 如果用户 a 将该权限授予用户 b, 用户 b 用自己的账号登录后, 用户 b 的用户映像空间里提供用户 a 在认证服务器存放的映像文件相关信息, 但是不提供下载该映像文件;

[0031] (2) 加载: 如果用户 a 将该权限授予用户 b, 则用户 b 用自己的账号登录后, 用户 b 的用户映像空间提供用户 a 在云端存放的映像文件相关信息, 并支持下载该映像文件和加载为虚拟磁盘, 但用户 b 对该虚拟磁盘的修改不更新到云端对应的映像文件中;

[0032] (3) 加载受限: 如果用户 a 将该权限授予用户 b, 则用户 b 用自己的账号登录后, 用户 b 的用户映像空间提供用户 a 在云端存放的映像文件相关信息, 并支持下载该映像文件和加载为虚拟磁盘, 但用户 b 访问该虚拟磁盘的拷贝、截屏、另存为和打印操作都进行了限制。

[0033] 而且, 在用户对其他用户授予用户权限时, 由相关负责人登录客户端审批, 并由客户端进行记录。

[0034] 而且, 认证服务器保存文件密钥 M1 时, 采用管理密钥 M3 进行加密, 加密结果记为 M4; 将映像文件的文件密钥 M1 传输到客户端时, 先用管理密钥 M3 对加密结果 M4 进行解密得到文件密钥 M1, 然后用会话密钥 M2 对文件密钥 M1 进行加密后传输。

[0035] 本发明采用较为成熟的虚拟磁盘技术, 通过网络共享构建无缝式的虚拟磁盘环境, 实现数据多域独立存储和数据隔离, 并以在云端分布式存储的方式, 实现了数据便捷、快速、安全共享与权限审核; 独具特色的共享审批功能, 在保证用户数据安全、操作简单、使用方便的前提下能提升了信息共享行为的合法性和安全性。本方法为政府机关、公司企业、

开发团队等注重信息安全的机构营造一种信息传输安全、快速、方便的氛围。在机密信息传输过程中,永远不用担心信息泄露、窃取、破坏的问题,因为就算信息被泄露,脱离本方法后也无法使用;就算信息被窃取,窃取者看到的只是乱码;就算信息被破坏,云端仍有备份。同时,在各种行业里,对支持业务的数据和业务处理的需求出现了爆炸式的增长,这导致了能源消耗量的激增以及对数据中心容量需求的增长。云计算则提供了解决问题的另一种思路,使用云计算的模式,企业无需为数据中心扩容、基础架构硬件和软件的采购以及应用软件的采购投入大量资金。与以往工作相比,本发明技术方案有自己的独特之处,主要表现为:

- [0036] 1. 能在实现文件共享的前提下保障云平台数据安全,将内部泄密的可能性降低到最小。
- [0037] 2. 在系统异常时不会损坏文件。
- [0038] 3. 将文件分块传送至云端,通过分布式文件管理系统,实现负载均衡,防止海量数据造成服务器崩溃。
- [0039] 4. 进行透明式加解密,自动提醒用户加密,操作简单、使用方便。
- [0040] 5. 能有效防止非法拷贝和截屏,从而避免了意外途径的数据泄露。

附图说明

- [0041] 图 1 是本发明的系统架构图。
- [0042] 图 2 是本发明的实施例流程图。
- [0043] 图 3 是本发明的客户端与认证服务器交互示意图。
- [0044] 图 4 是本发明的客户端与云端交互示意图。
- [0045] 图 5 是本发明实施例的加载受限安全策略示意图。
- [0046] 图 6 是本发明实施例的共享审批示意图。
- [0047] 图 7 是本发明实施例的云平台示意图。
- [0048] 图 8 是本发明实施例的透明加解密过程示意图。
- [0049] 图 9 是本发明实施例的文件读取示意图。
- [0050] 图 10 是本发明实施例的文件写入示意图。
- [0051] 具体实施方式
- [0052] 以下结合附图和实施例详细说明本发明技术方案。
- [0053] 参见图 1,系统架构由客户端、云端和认证服务器三方组成。认证服务器负责客户端和云端交互过程中的身份认证与密钥分发,它与云端之间进行密钥协商与分发,响应客户端的用户请求,接收客户端的身份证书;云端负责对用户上传的映像文件进行分布式存储与管理;客户端可以供用户进行虚拟磁盘的基本操作,如创建映像文件、将映像上传至云端、从云端加载映像等等。
- [0054] 参见图 2,软件流程包括用户注册登录及身份验证,系统初始化,创建虚拟磁盘映像文件,加载虚拟磁盘映像文件,卸载虚拟磁盘,上传映像文件,权限共享与权限审批等。
- [0055] 其中,本方法基于 C/S 模式,通过接入身份认证技术与密钥协商技术相结合,实现严格的访问控制机制。
- [0056] 实施例的具体流程如下:

[0057] 步骤 1,当用户从客户端输入用户名和密码时,客户端首先将用户名和密码采用 SHA-2 哈希函数进行处理,再将处理所得哈希值用认证服务器的公钥加密后发送至认证服务器,由认证服务器进行身份认证;当认证通过时进入步骤 2,未通过时在客户端提示用户的用户名或密码不正确。

[0058] 步骤 2,进行系统初始化:首先初始化用户空间信息和用户权限信息,用户空间信息为用户在认证服务器存放的映像文件的相关信息,包括各映像文件的状态是否自动加载;然后,从云端下载状态为自动加载的映像文件并加载为虚拟磁盘,具体方式如下,

[0059] 客户端向认证服务器发送加载该映像文件的请求,认证服务器接收到加载请求后,通知云端将映像文件传送到客户端,然后认证服务器将该映像文件的文件密钥 M1 传输到客户端;客户端用文件密钥 M1 对映像文件进行解密,解密完成后将该映像文件加载为虚拟磁盘。

[0060] 认证服务器可以采用数据库表技术实现信息管理,实施例进行初始化用户空间信息和用户权限信息时,具体过程为:认证服务器依据该用户的用户名对数据库表执行一个 SQL 查询语句,查询结果包括该用户在服务器端存放的所有映像文件的相关信息,以及其他用户授予该用户浏览或加载权限的映像文件相关信息。查询到的信息包括文件所有者、映像文件名、映像文件大小、属性、是否自动加载,这些是用户空间信息;同时也查询到文件权限明细,即用户权限信息。接着,服务器端的程序将这些信息传给客户端,客户端成功接收到这些信息后,采用用户空间界面将这些信息显示给用户,用户即可浏览用户空间信息和相应权限信息。

[0061] 步骤 3,对用户在其用户映像空间的操作进行管理,用户在其用户映像空间的操作包括创建映像文件、加载映像文件、卸载虚拟磁盘、修改用户权限、浏览用户空间信息和退出:

[0062] 当用户创建映像文件时,输入待创建的映像文件的保存路径、文件名以及文件大小(实施例中映像文件的扩展名为 . vdk)后,客户端首先向认证服务器进行创建映像文件的申请,认证服务器接收到申请后,为待创建的映像文件分配一个随机的文件密钥 M1 并传输到客户端。

[0063] 当用户加载映像文件时,对于该用户新创建的映像文件,客户端直接将该映像文件加载成虚拟磁盘;对于该用户先前创建的映像文件,客户端从云端下载加密的映像文件,并向认证服务器申请获取对应的文件密钥 M1 进行解密,解密完成后将该映像文件加载为虚拟磁盘;当用户加载其它用户的映像文件时,客户端从云端下载加密的映像文件,并向认证服务器申请对应的文件密钥 M1,认证服务器查判断用户权限是否合法,若合法则将文件密钥 M1 传输到客户端,客户端用文件密钥 M1 对映像文件进行解密,解密完成后将该映像文件加载为虚拟磁盘,若不合法则认证服务器拒绝请求。用户新创建的映像文件为本地映像文件,客户端从云端下载的用户本身或其他用户的映像文件,可称为云端映像文件。

[0064] 当用户卸载虚拟磁盘时,客户端将虚拟磁盘从资源管理器中卸载,将虚拟磁盘里面的内容更新到映像文件中去,并用对应的文件密钥 M1 对虚拟磁盘的映像文件进行加密,将加密后的映像文件 (*. vdk) 上传到云端,同时将映像文件的相关信息上传到认证服务器。

[0065] 当用户修改用户权限时,客户端将修改后的用户权限信息上传到认证服务器。

[0066] 当用户浏览用户空间信息时,客户端向用户显示初始化用户空间信息的结果。用

户浏览后可以选择下载映像文件，客户端具体实现与前述用户加载映像文件时相同；其次，用户可以在用户空间上传已创建映像文件或更新后的映像文件，具体实现与从云端加载映像文件相类似。

[0067] 当用户退出时，客户端卸载所有加载的虚拟磁盘，将每个虚拟磁盘的映像文件分别用对应的文件密钥 M1 进行加密，将加密后的映像文件 (*. vdk) 上传到云端，同时将所有映像文件的相关信息上传到认证服务器。

[0068] 为了提高安全性，认证服务器在每个映像文件创建时分配一个文件密钥 M1 并保存，认证服务器将映像文件的文件密钥 M1 传输到客户端时，用会话密钥 M2 对文件密钥 M1 进行加密后传输，客户端接收后采用会话密钥 M2 解密得到文件密钥 M1，然后用文件密钥 M1 对从云端下载的映像文件解密，解密完成后加载。对于要加载的映像文件，客户端直接调用现有技术中的 Filedisk 驱动加载本地的虚拟磁盘即可。

[0069] 分配的文件密钥 M1 可以采用随机数发生器生成随机数后，结合用户信息散列值生成，随后存放在认证服务器中的数据库表中。认证服务器保存文件密钥 M1 时也可采用密文形式，即采用管理密钥 M3 对文件密钥 M1 进行加密，加密结果记为 M4；将映像文件的文件密钥 M1 传输到客户端时，先用管理密钥 M3 对加密结果 M4 进行解密得到文件密钥 M1，然后用会话密钥 M2 对文件密钥 M1 进行加密后传输。管理密钥 M3 可以采用管理员用户名和登录密码的哈希值，管理员用户名和登录密码在管理员登陆认证服务器的时候获取。

[0070] 参见图 3，客户端与认证服务器交互主要包括用户注册、身份认证以及客户端映像文件密钥分发。其中，认证服务器还可以对域中各成员进行管理，如黑名单管理、用户权限验证等，采用服务器端数据库表存放用户身份信息和映像文件信息。实施例可以采用计算机软件技术实现基于云计算的网络虚拟磁盘管理系统，设计用户使用过程为：用户首先访问系统，进行账号注册、用户登录、身份验证，通过客户端与认证服务器交互进行访问控制；身份合法的用户进入客户端主界面，进行磁盘管理、磁盘虚拟、用户空间、权限控制操作；同时，管理员还可以在认证服务器端进行黑名单管理和用户管理。客户端通过 Internet 与认证服务器之间采用 Socket 通信从而达到交互的目的。

[0071] 为便于实施参考起见，本发明提供了实施例的身份认证实现具体说明如下：

[0072] 客户端程序首先将用户名和密码采用美国国家标准局(ANSI)和国际标准化组织 ISO 推荐的 SHA-2 哈希函数进行处理，再将处理后的哈希值用服务器公钥加密发送至服务器。同时，利用下述技术实现用户身份的认证(假定 A 为客户端，B 为认证服务器)：

[0073] ①客户端 A 将自己的身份信息 ID_A 传递给认证服务器 B，但是认证服务器 B 不能确定此信息是来自客户端 A 还是窃密者 C；

[0074] ②认证服务器 B 收到身份信息 ID_A 后，产生一个随机的消息 R_B ，用客户端 A 的公钥 P_A 加密身份信息 ID_A 和消息 R_B 得到 $y_1 = E_{P_A}(ID_A \parallel R_B)$ ，其中 $E_{P_A}(\cdot)$ 表示用公钥 P_A 进行的加密过程；并用自己的私钥 S_B 计算得到签名 $y_2 = D_{S_B}(E_{P_A}(ID_A \parallel R_B))$ ，其中 $D_{S_B}(\cdot)$ 表示用私钥 S_B 进行的解密过程；将加密结果 $E_{P_A}(ID_A \parallel R_B)$ 和签名 $D_{S_B}(E_{P_A}(ID_A \parallel R_B))$ 传送给客户端 A。

[0075] ③客户端 A 收到消息后用认证服务器 B 的公钥 P_B 对签名 $D_{S_B}(E_{P_A}(ID_A \parallel R_B))$ 进行验

证，验证方式为判断等式 $E_{P_B}(D_{S_B}(E_{P_A}(ID_A \parallel R_B))) = E_{P_A}(ID_A \parallel R_B)$ 是否成立，其中 $E_{P_B}()$ 表示用公钥 P_B 进行的加密过程。由于只有合法的认证服务器 B 才拥有私钥 S_B ，因此客户端 A 就可以通过上述等式成立与否确认通信对方是否为认证服务器 B。如果等式成立，客户端 A 确认通信对方为认证服务器 B，并对 $E_{P_A}(ID_A \parallel R_B)$ 进行解密，解密 $(E_{P_A}(ID_A \parallel R_B)) = ID_A \parallel R_B$ ，再分离出 ID_A 和 R_B 。

[0076] ④客户端 A 将步骤③求得的 R_B 用认证服务器 B 的公钥 P_B 加密得到 $E_{P_B}(R_B)$ ，将 $E_{P_B}(R_B)$ 传送给认证服务器 B，因为只有合法的客户端 A 可以求得 R_B ，从而可以得到正确的 $E_{P_B}(R_B)$ ；认证服务器 B 用自己的私钥 S_B 解密 $E_{P_B}(R_B)$ 即可得到 R_B ，将此 R_B 与原来在步骤②随机产生的 R_B 对比，确认对方是否是意定的客户端 A。

[0077] 该身份认证技术具备以下特点：

[0078] (1) 实现了通信双方的交互认证；

[0079] (2) 防敌手的假冒攻击；

[0080] (3) 防重放攻击；

[0081] (4) 提供了消息的机密性和完整性保护。

[0082] 本发明实施例的会话密钥协商采用 PGKA 协议，该协议发表在《Computer Standards & Interfaces》上，它不仅具备抗主动攻击的能力，而且拥有仅需 2 轮通信的优点。为实施参考起见，提供如下：

[0083] 假设 p 、 q 分别为大素数，且满足 $p = 2q + 1$ ， Z_q 为 q 阶循环群， G_q 为 p 阶循环群 \mathbb{Z}_p^* 中的一个二次剩余子群，即 $G_q = \{i^2 \mid i \in \mathbb{Z}_q^*\}$ ， g 为 G_q 的生成元， H 为安全 hash 函数。假定 $P = \{P_1, P_2, \dots, P_n\}$ 表示初始的参与通信的成员集，且集合中成员的下标构成一个环，即 $P_{n+1} = P_1$ ， $P_0 = P_n$ ，依此类推。具体的密钥协商过程如下：

[0084] Step1. 每一个参与方 P_i ($1 \leq i \leq n$) 选择一个随机数 $x_i \in Z_q$ ，计算并广播消息 $y_i = g^{x_i} \bmod p$ ；

[0085] Step2. 接收到所有的 y_j ($1 \leq j \leq n, j \neq i$) 后，

[0086] P_1 随机选取子群元素 $R_1 \in G_q$ ，计算并广播消息 $z_1 = R_1 y_2^{x_1} \bmod p$ ，

[0087] ...

[0088] P_n 随机选取子群元素 $R_n \in G_q$ ，计算并广播消息 $z_n = R_n y_{n-1}^{x_n} \bmod p$ ，

[0089] P_i ($2 \leq i \leq n-1$) 选择一个随机数 $s_i \in Z_q$ ，计算并广播消息 $(z_i, \eta_i, \mu_i, \omega_i)$ ，其中

[0090] $z_i = (y_{i+1} y_{i-1})^{x_i} \bmod p$ ， $\eta_i = g^{s_i} \bmod p$ ， $\mu_i = (y_{i+1} y_{i-1})^{s_i} \bmod p$ ，

$\omega_i = s_i + x_i H(z_i \parallel \eta_i \parallel \mu_i) \bmod q$ 。

[0091] Step3. 接收到所有的 $(z_j, \eta_j, \mu_j, \varphi_j)$ ($1 \leq j \leq n, j \neq i$) 后, 成员 P_i 计算共享密钥:

$$[0092] k \equiv \begin{cases} (g_1^{-1})^2 \prod_{i=1}^{(n-1)/2} z_{2i} \bmod p & \text{当 } n \text{ 为奇数} \\ g_1^{-1} \prod_{i=1}^{n/2-1} z_{2i+1} \bmod p & \text{当 } n \text{ 为偶数} \end{cases},$$

[0093] 其中, g_1 表示生成元。

[0094] 最后可得: $k \equiv g_1^{-x_1x_2+x_2x_3+\dots+x_nx_1} \bmod p$, 从而获得共享密钥 k , 即实施例所用会话密钥 M2。

[0095] 以上 \equiv 表示 mod 运算的结果, 在数论里相当于等号。

[0096] 为便于实施参考起见, 以下提供实施例的客户端、认证服务器端和云端分别的具体实现说明:

[0097] 1 客户端实现

[0098] 1.1 磁盘管理

[0099] 用户登录后, 客户端向认证服务器发送加载状态为“主动加载”的映像文件的请求, 认证服务器接收到加载请求后, 通知云端将映像文件传送到客户端。

[0100] 然后认证服务器先用管理密钥 M3 对加密结果 M4 进行解密得到文件密钥 M1, 然后用会话密钥 M2 对文件密钥 M1 加密后传输给客户端。

[0101] 客户端接收后采用会话密钥 M2 解密得到文件密钥 M1, 然后用文件密钥 M1 对映像文件进行解密, 解密完成后调用 Filedisk 的 mount 命令, 将该映像文件加载为虚拟磁盘。

[0102] 该功能给用户的体验是, 本地凭空多出来了一个磁盘, 这个磁盘可以像其它本地磁盘一样进行格式化、添加文件、删除文件等操作。

[0103] 创建虚拟映像、加载、卸载虚拟磁盘

[0104] 创建映像文件: 用户选择待创建的映像文件所在的盘符、文件名以及文件大小(实施例映像文件的扩展名为 . vdk), 点击确定后, 客户端程序首先向服务器进行映像创建的申请, 服务器端程序接受到后, 为待创建的映像文件分配一个随机的文件密钥 M1, 并用会话密钥 M2 进行加密然后传给客户端的对应用户。磁盘卸载和客户端程序退出的时候所用的加密密钥就是文件密钥 M1。

[0105] 加载映像文件: 对于新创建的映像文件, 直接用 Filedisk 加载成本地磁盘。对于先前创建的映像文件, 由于它是加密了的, 所以先要向服务器申请获取对应的文件密钥 M1 进行解密, 然后才能用 Filedisk 进行加载。

[0106] 卸载虚拟磁盘: 将创建的虚拟磁盘从资源管理器中卸载, 系统将磁盘里面的内容更新到映像文件中去, 并用对应的文件密钥 M1 进行加密, 并自动将其上传到服务器上。

[0107] 用户空间

[0108] 客户端的用户在通过身份验证后, 认证服务器会自动将用户在云端保存的所有映像文件的信息传送给用户, 映像文件内容将实时显示在用户空间中。

[0109] 用户可以查看映像文件里面保存的内容, 并可将映像文件下载到本地并加载为本地磁盘。加载磁盘后, 用户对该磁盘的修改会在用户客户端程序退出时, 自动更新到服务器端的映像文件中。

[0110] 用户除了查看自己在云端的映像文件的内容之外, 还可以查看其他用户授予该用

户浏览或加载权限的映像文件里面的内容。同时该用户还可以下载该用户有加载权限的映像文件，并可在本地进行加载。

[0111] 用户对本地加载的映像文件进行操作后，可以手动上传到云端或者在退出系统后自动上传。

[0112] 权限控制

[0113] 用户对其它用户的映像文件默认的权限是不可加载且不可浏览。当用户由于某些需要将该用户的数据和其他某用户进行共享时，可以通过权限控制来实现其他用户浏览或下载该用户在服务器端的映像文件，从而达到数据共享的目的。

[0114] 本方法定义的权限有三种：

[0115] (1) 浏览：如果用户 A 将该权限授予用户 B，那么用户 B 用自己的账号登录后，可以在用户空间里面查看到用户 A 在服务器端存放的映像文件相关信息，但是不可以下载该映像文件。

[0116] (2) 加载：如果用户 A 将该权限授予用户 B，则用户 B 不仅可以在他的用户空间里面看到用户 A 在服务器端存放的映像文件相关信息，还可以下载该映像文件，并可将下载后的映像文件 I 在用户 B 的本地加载为本地磁盘，从而使用里面的文件，但用户 B 对该本地磁盘的修改不会更新到服务器端对应的映像文件中。

[0117] (3) 加载受限：如果用户 a 将该权限授予用户 b，则用户 b 用自己的账号登录后，用户 b 的用户映像空间提供用户 a 在云端存放的映像文件相关信息，并支持下载该映像文件和加载为虚拟磁盘，但用户 b 访问该虚拟磁盘的拷贝、截屏、另存为和打印操作都进行了限制。

[0118] 如图 5，用户 1 和用户 2 所使用的云终端之间进行数据共享，安全策略有三项：保存与打印控制、文字拷贝控制、屏幕拷贝控制。

[0119] 实施例的页面保存与打印控制实现方式：

[0120] 对于业务上的文档，大部分为 word 文档和 pdf 文档，因此对这两种主流文档的保存和打印控制是非常必要的。控制的前提是捕获事件 spplcstion. SaveAs，这个事件对应着 windows 一个内核函数，该过程主要通过系统钩子模块完成。spplcstion. SaveAs 为 Window 提供的现有技术。

[0121] 钩子实际上是一段用来处理系统消息的程序，通过系统调用，将其挂入到系统，它是 Windows 的消息处理机制中的一个监视点。在设置钩子的情况下，Windows 的消息传递过程会发生改变，钩子可以在系统中的消息流到达目的窗口过程前监控它们。钩子函数可以监视指定窗口的某种消息，而且所监视的窗口可以是其他进程所创建的。当消息到达后，钩子机制允许应用程序截获处理窗口消息或特定事件。这时钩子函数既可以加工处理(改变)该消息，也可以不进行处理而继续传递该消息，还可以强制结束消息的传递。

[0122] 实施例的文字拷贝控制实现方式：

[0123] 文字拷贝主要是实现当用户在操作敏感业务系统时候，需要禁止用户通过拷贝方式将重要数据保存下来。其关键技术是对剪贴板的监控。如果发现剪贴板在用户操作过程中发现变化，则说明用户已经实施了拷贝行为。

[0124] Windows 剪贴板是一种开销比较小的 IPC(InterProcess Communication, 进程间通讯) 机制。Windows 系统支持剪贴板 IPC 的基本机制是由系统预留一块全局共享内存，用

来暂存在各进程间需要交换的数据：提供数据的进程创建一个全局内存块，并将要传送的数据移到或复制到该内存块；接受数据的进程（也可以是提供数据的进程本身）获取此内存块的句柄，并完成对该内存块数据的读取。为了实现上述功能，Windows 提供了存放于系统文件 USER32.dll 中的一组 API 函数、消息和预定义数据格式等，并通过对这些函数、消息的使用来管理在进程间进行的剪贴板数据交换。系统文件 USER32.dll 为 Windows 提供的现有技术。

[0125] 系统调用剪贴板的简化步骤为：首先通过调用 OpenClipboard 函数打开剪贴板，如果是获取剪贴板的内容则调用 GetClipboardData 函数，如果是设置剪贴板内容则先通过调用 EmptyClipboard 函数清空剪贴板，然后调用 SetClipboardData 函数设置剪贴板内容（在获取和设置剪贴板内容的函数的参数中都要有相应的数据格式）。这些函数为 Windows 提供的现有技术。

[0126] 实施例的屏幕拷贝控制实现方式：

[0127] 屏幕拷贝主要通过对用户键盘操作进行控制完成，一方面禁用屏幕拷贝按钮，另外一方面禁止具有屏幕拷贝功能的进程出现。键盘控制主要也通过系统钩子模块完成。

[0128] Win32 系统会为钩子建立一个钩子链（HookChain），一个钩子链实际上是一个指针列表，其指针指向钩子的各个处理函数，这些函数是一种特殊的回调函数。钩子链的运作方式类似于栈，在钩子链中最后安装的钩子放在钩子链的最前面，最先安装的钩子则放在钩子链的最底层，所以最后加入的钩子优先获得控制权。

[0129] 通过挂接系统键盘钩子，即可完成对屏幕打印键的控制。

[0130] 权限审批

[0131] 在客户端与云端交互中，对于企业，每个部门作为一个相对独立的存储域。对于部门每一个员工来说，他可以通过云终端（客户端）创建自己的用户空间。企业内部，成员之间可以进行方便快捷的文件共享，但对于部门之间的共享行为，需要通过相关部门的负责人进行审批。存储映像文件信息存放在认证服务器上，而映像文件却存放在云端。映像文件的上传和下载则通过客户端与云端进行，其中的管理者便是认证服务器。参见图 4，某公司有部门一、部门二、部门三、部门四，部门一的用户之间可以进行部门内部资源共享，当部门一和部门二之间要求部门间共享时，需要由部门一的负责人登录客户端进行审批。

[0132] 参见图 6，对于一般的企业文件管理系统，内部文件共享的机制并不完善。一个文件所有者或负责人可以将其具有访问权限的文件共享给任何人，包括部门外业务人员以及潜在的泄密者，并且这种共享行为没有日志记录，因而无法确保文件共享行为的合法性，容易导致机密信息的扩散和泄露。

[0133] 本发明所提供基于云计算的网络虚拟磁盘管理系统，采用严格的共享审批机制，当文件所有者需要将其映像文件权限共享给其它用户时，该共享行为需要通过相关负责人审批。相关负责人发现部门外业务人员或潜在泄密者时，则拒绝此次共享，从而防止机密文件泄露。这种共享审批与日志审计相结合的访问控制技术，确保了文件共享行为的合法性和安全性。

[0134] 对于公司来说，一个部门可以称作一个域。现在从上述例子中将部门抽象成域。例如，当 1 号域普通成员 a1 将其空间的浏览权限赋予 2 号域普通成员 b1，登录 b1 用户，在 b1 用户空间的浏览器里，可以看到 a1 用户空间文件信息。此时，b1 若尝试加载，系统会提

示仅有浏览权限，无法加载。

[0135] 当 1 号域普通成员 a1 将 2 号域普通成员 b1 的权限改为加载，由于是域间的共享行为，需要 a1 所在的 1 号域负责人 admina 来进行审批。

[0136] 登录负责人 admina，负责人界面与普通成员的多出一个权限审批页面，在共享审批列表中，允许此次共享行为。此时，在 2 号域普通成员 b1 的用户空间中，此时可以加载 1 号域 a1 的映像，普通加载可以对文件内容进行随意拷贝、截屏、另存为等操作。

[0137] 当负责人 admina 拒绝此次共享行为时，a1 会收到拒绝相关消息。因而 b1 无法对 a1 映像进行加载等进一步操作。

[0138] 共享行为和审批操作，都会提交到认证服务器的日志系统里，这样由认证服务器记录每一次用户之间的共享行为，包括时间、共享内容、对象及审批结果等。

[0139] 认证服务器端实现

[0140] 2.1 用户管理

[0141] 服务器端程序在初始化的时候，通过访问数据库表将所有注册了的用户信息显示出来，方便管理员进行管理。管理员可以将这些用户进行删除、加入黑名单的操作，删除后的用户的该账户从此无法使用，只能重新注册，而被加入黑名单的用户账号会被屏蔽，无法登录和注册。这些操作会引起服务器端程序对数据库表的相应修改。除此之外，管理员还可以设置每个用户在服务器端可上传的映像文件总大小的上限。

[0142] 黑名单管理

[0143] 加入黑名单的账号从此无法再被使用，管理员拥有将黑名单中的某账号从黑名单中移除从而恢复为可用账号的权限，该操作会引起服务器端程序对数据库表的相应修改。

[0144] 磁盘管理

[0145] 在服务器端可以设一个公共映像文件，该映像文件是可以被所有客户端用户浏览和加载的。主要是为了方便管理员通过客户端程序向所有用户发布某些文件，只要将该文件放到有映像文件加载后的磁盘中即可，而不需要逐一拷贝给所有用户，当用户登录的时候自动加载该映像文件为本地磁盘从而浏览使用服务器端程序发布的这些文件。极大地减轻了管理员发布文件的工作。

[0146] 云端实现

[0147] 3.1 云平台搭建

[0148] 近年来，云存储技术发展也日益成熟。本发明正是基于此技术，通过开源的 Apache Hadoop 搭建起云平台，解决数据的可访问性、安全性、移动性，明确定义与数据所有权、归档、发现和搜索相关的角色和职责并实现了自动负载均衡及透明扩容缩容。参见图 7，实施例通过开源软件 Apache Hadoop 搭建云平台。Apache Hadoop 构建在虚拟主机上，作为云计算平台。其设计核心是 MapReduce 实现和 HDFS (Hadoop Distributed File System)，它们源自 MapReduce(一种编程模型，用于大规模数据集的并行运算)和谷歌文件系统 (Google File System)。Hadoop 云平台中，MapReduce 和 HDFS 是两大最重要的组成部分。MapReduce 包括 map(映射) 和 reduce(化简) 两个过程，HDFS 包括名称节点 NameNode 和数据节点 DataNode。

[0149] 实施例采用全分布模式，即 Hadoop 配置在不同的主机上，作为集群运行。Hadoop 基本组成结构为，一台机器作为 Master 主机(Hadoop-A)、两台机器分别为 Slave 机

(Hadoop-B\Hadoop-C)。其中, Hadoop-A 为名称节点, 也是数据节点, Hadoop-B\Hadoop-C 分别为数据节点, 由于整个平台搭建在私有域中, 因此构成了私有云存储与管理平台。

[0150] 其中, 云平台设计中, 独具特色的容崩自检机制和容灾备份与负载均衡机制, 为云平台数据可用性、安全性提供了保证。

[0151] 实施例的容崩自检机制实现说明如下:

[0152] 在云平台中, 没有名称节点, 文件系统将无法使用。因此, 名称节点能够经受故障是非常重要的, 本方法通过在 Hadoop 上运行一个二级名称节点的机制来保证系统的可靠性与可用性。

[0153] 对于二级名称节点, 其不能作为名称节点使用。这个二级名称节点的重要作用就是定期的通过编辑日志合并命名空间镜像, 以防止编辑日志过大。该节点一般在其他单独的物理计算机上运行, 因为它也需要占用大量 CPU 和内存来执行合并操作。它会保存合并后的命名空间镜像的副本, 在名称节点失效后就可以使用。

[0154] 实施例的容灾备份与负载均衡机制实现说明如下:

[0155] HDFS 文件块的概念与普通的文件系统中分块的概念基本相同, 不过是更大的单元, 默认为 64 MB, 这样的一个好处是减少寻址时间开销。与单一磁盘上的文件系统不同的是, HDFS 中小于一个块大小的文件不会占据整个块的空间。

[0156] 为了恢复损坏的块以及应对磁盘或机器的故障, 每个块都将自己的另外两个副本放在其它数据结点上, 并保证副本数目的恒定。这个工作由设立在名称结点上的监控程序完成。如果一个块发生损坏或机器故障, 系统会在其他地方读取另一个副本, 并将此副本复制一份, 通过查看文件块分配索引表, 根据负载均衡原则, 将新的副本存放在选定的数据结点上, 以保证副本的数量回到正常水平, 这个过程对用户是完全透明的。

[0157] 在云端, 设置统计云端映像文件加载次数等信息反映文件热门程度。对于非常热门的文件块, 可以给其设置更高的副本数量以提高集群的读取负载量, 从而达到负载均衡。

[0158] 在不改变用户使用习惯、计算机文件格式和应用程序的情况下, 对虚拟映像文件采取“驱动级 AES 透明动态加解密技术”, 对指定类型的文件进行实时、强制、透明的加解密。即在正常使用时, 计算机内存中的文件是以受保护的明文形式存放, 但硬盘上保存的却是加密状态的数据。由于加解密是在驱动级上实现的, 在每次读写文件时自动使用特定的文件密钥进行加解密, 对用户完全透明。对于非管理员用户而言, 他们甚至完全不需要知道密钥的存在。

[0159] 驱动级透明加密技术基于 windows 的文件系统(过滤)驱动(IFS)技术, 由于工作在受 windows 保护的内核层, 运行速度更快, 加解密操作更稳定。本方法在驱动级上实现文件的加解密操作如下图 8 所示。当用户双击打开文件或者修改后保存文件等读写操作后, IO 管理器根据用户操作生成文件读写操作指针 IRP(IRP 指针包括读操作指针 IRP_MJ_READ 或写操作指针 IRP_MJ_WRITE), IRP 传递到驱动程序, 执行指定的派遣函数。在派遣函数中执行加解密操作, 加解密操作结束后, 得到文件加解密之后的结果, 并将结果返回给 IO 管理器, 结束本次读写请求。

[0160] 参见图 9, HDFS 客户端中文件读取流程: 映像文件以分块的形式存储在云端数据结点上, 数据节点中存放文件块的索引通过名称节点获得。云终端与云端进行通信时, 客户端节点通过 JAVA 虚拟机(客户端 JVM), 运用 HDFS 开始分布式计算, 向名称节点 NameNode 请

求并获得数据块位置，客户端通过读取 `FSDataInputStream` (hadoop 中数据流类) 数据流，从各个数据节点 `DataNode` 上分别读取数据块并在终端进行合并，最后关闭管道，结束本次读取。

[0161] 参见图 10，HDFS 客户端中文件写入流程：文件写入与读取基本类似。云终端通过调用创建函数 `Create` 向名称节点 `NameNode` 发送请求，然后通过 `FSDataOutputStream` 数据流进行文件写入，数据块写入存储到各个数据节点 `DataNode` 上，各个数据节点 `DataNode` 进行回应，最后关闭管道。名称节点 `NameNode` 负责维持整个文件系统的负载均衡以及分块副本数目，创建完成后通知分布式文件系统。

[0162] 通信模块接口实现

[0163] Hadoop 中的 RPC 是 Hadoop 系统内部的通信机制，RPC(Remote Procedure Call Protocol) 远程过程调用协议，它是一种通过网络从远程计算机程序上请求服务，而不需要了解底层网络技术的协议。

[0164] PC 采用客户机 / 服务器模式。请求程序就是一个客户机，而服务提供程序就是一个服务器。当我们讨论 HDFS 的时候，通信可能发生在：

[0165] • 客户端 Client 与名称节点 `NameNode` 之间，其中名称节点 `NameNode` 是服务器

[0166] • 客户端 Client 与数据节点 `DataNode` 之间，其中数据节点 `DataNode` 是服务器

[0167] • 数据节点 `DataNode` 与名称节点 `NameNode` 之间，其中名称节点 `NameNode` 是服务器

[0168] • 数据节点 `DataNode` 与数据节点 `DateNode` 之间，其中某一个数据节点 `DateNode` 是服务器，另一个是客户端。

[0169] 本文中所描述的具体实施例仅仅是对本发明精神作举例说明。本发明所属技术领域的技术人员可以对所描述的具体实施例做各种各样的修改或补充或采用类似的方式替代，但并不会偏离本发明的精神或者超越所附权利要求书所定义的范围。

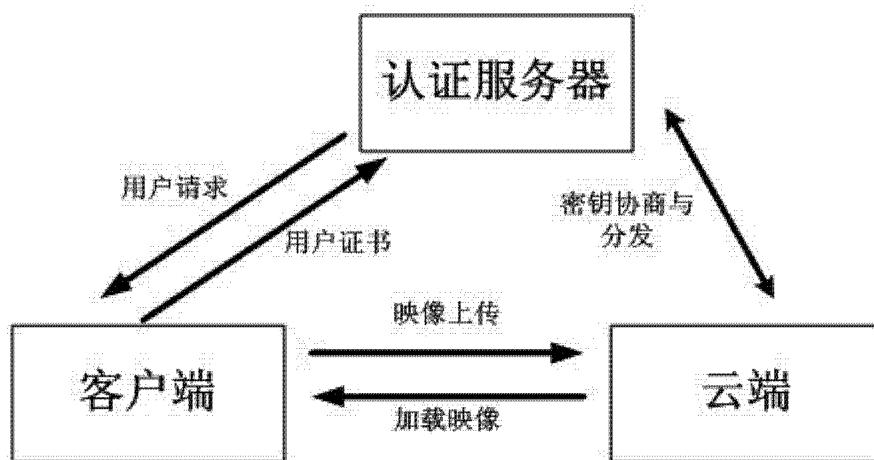


图 1

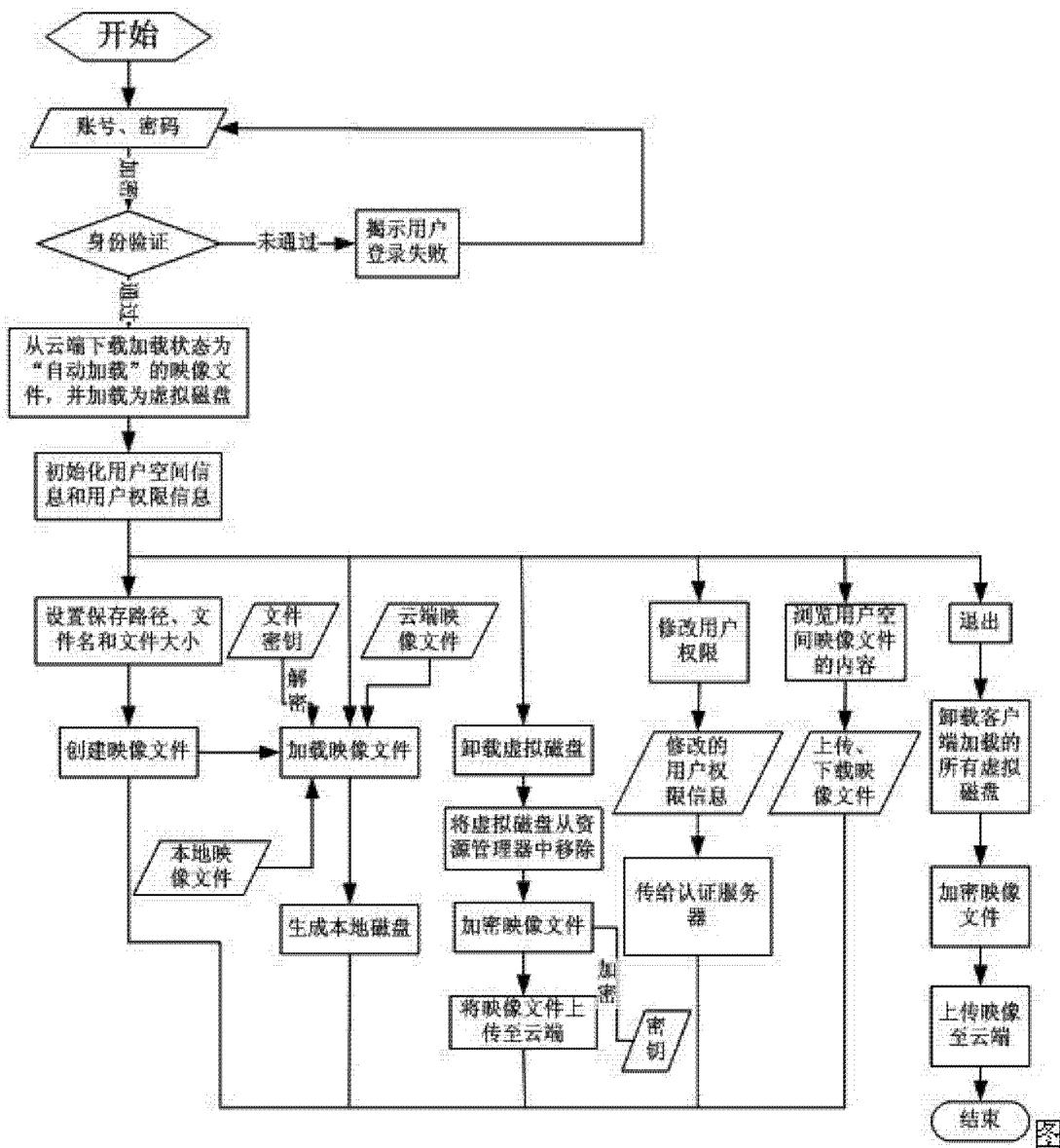


图 2

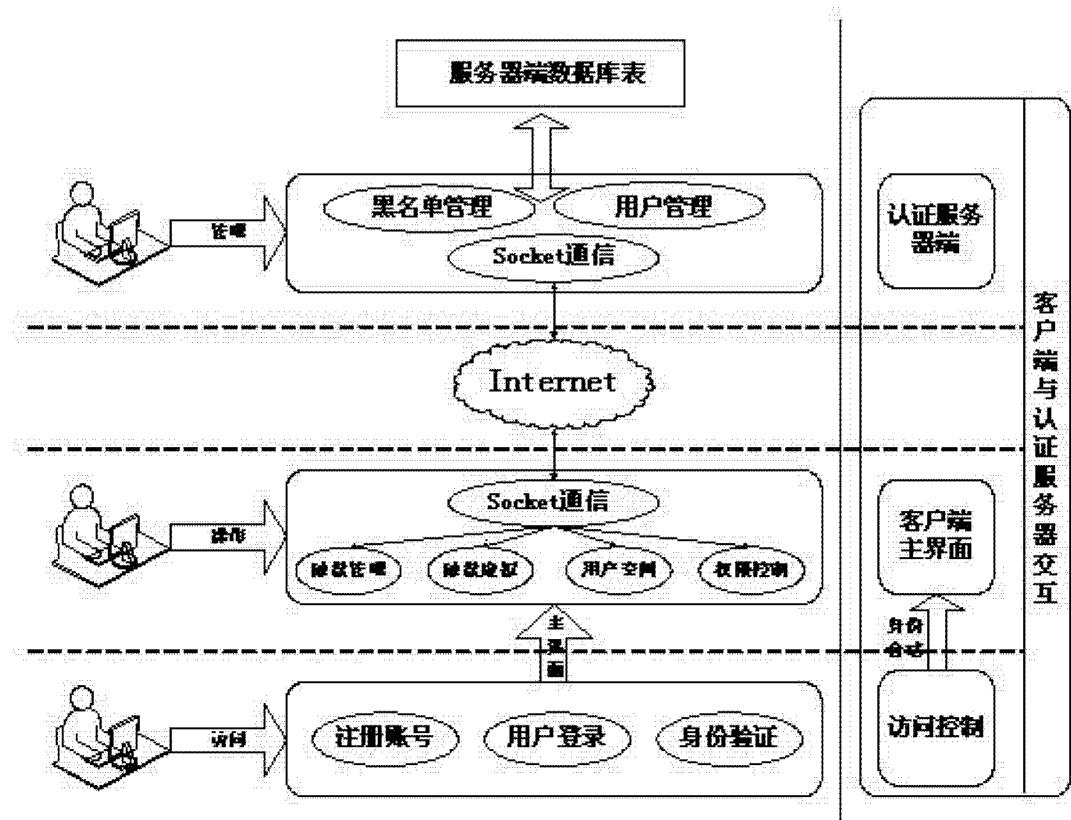


图 3

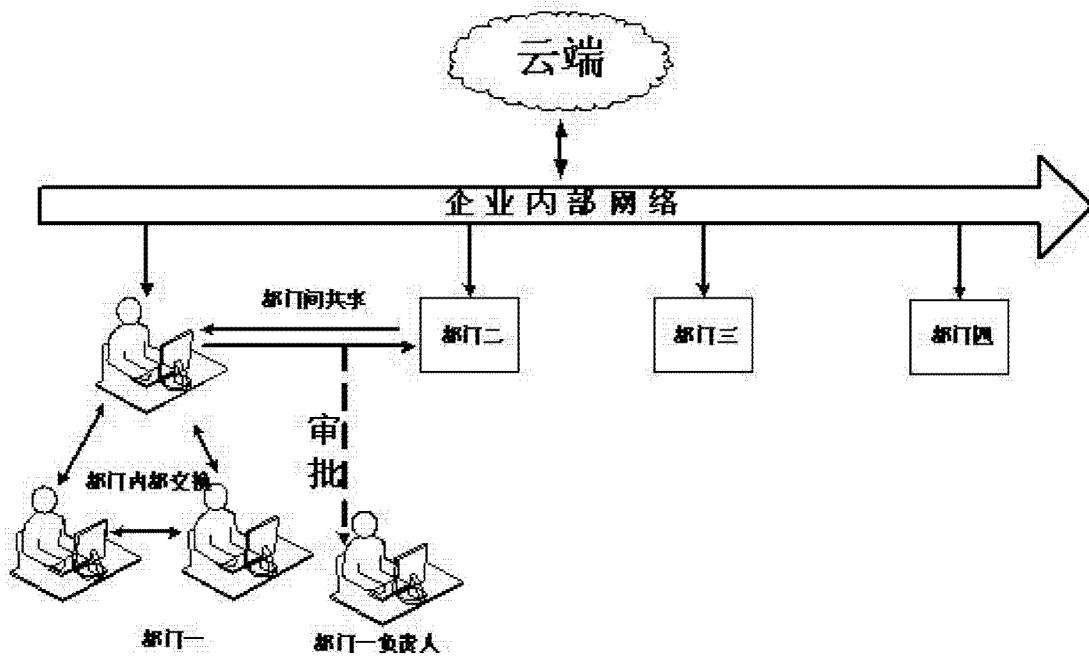


图 4

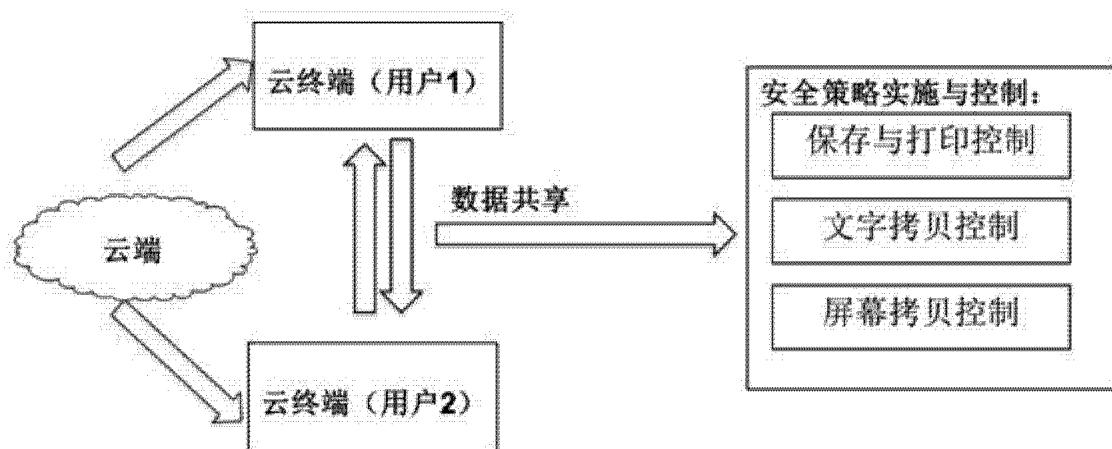


图 5

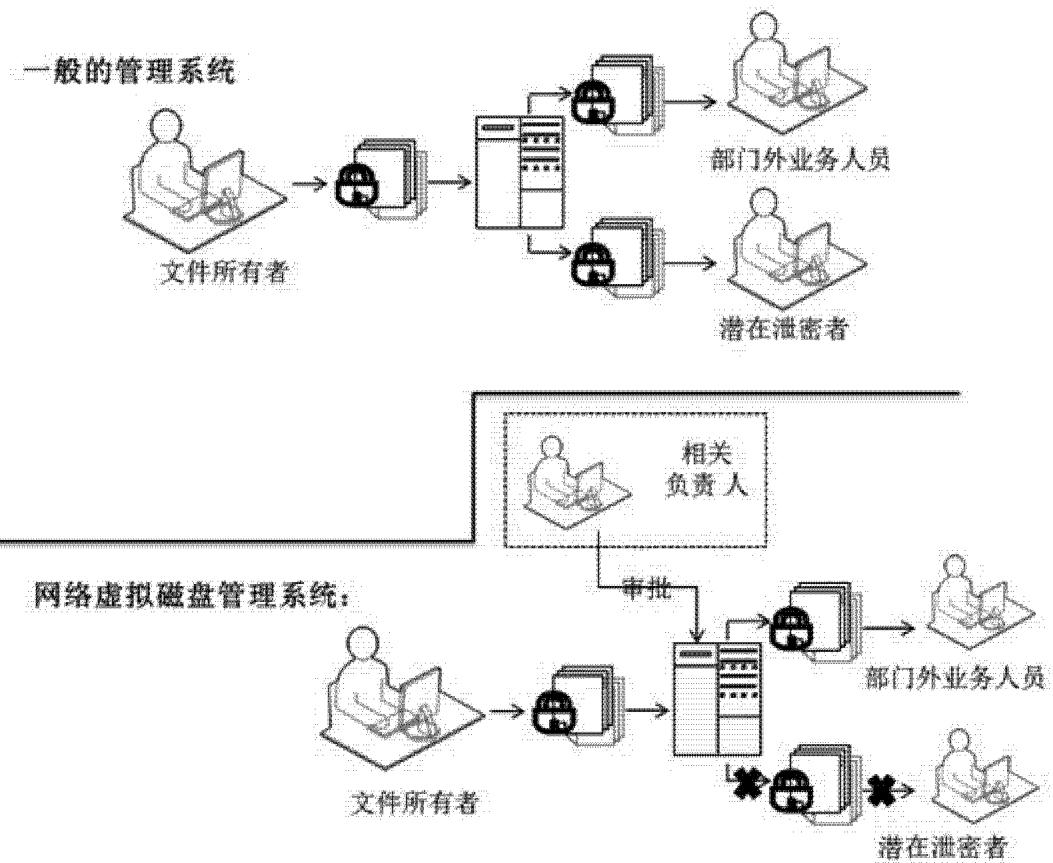


图 6

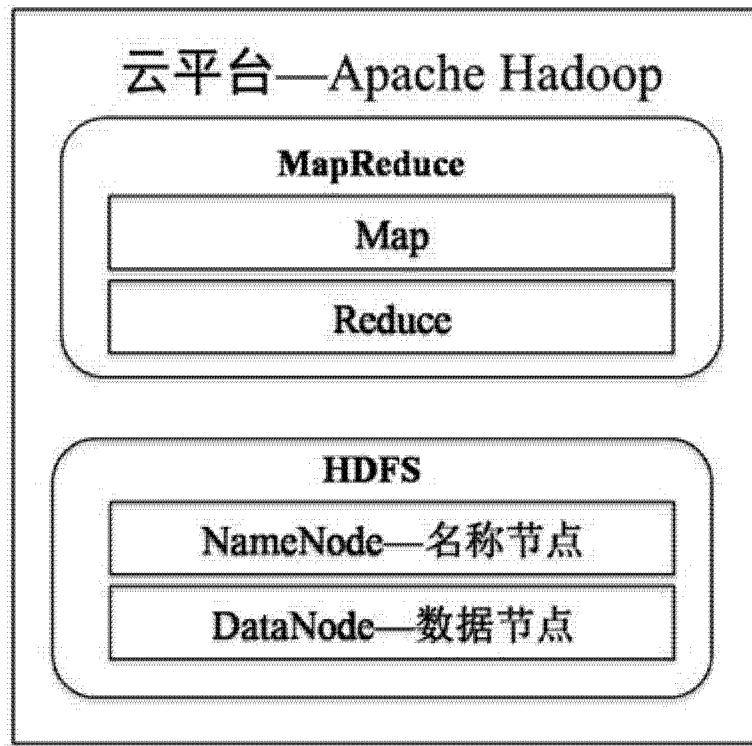


图 7

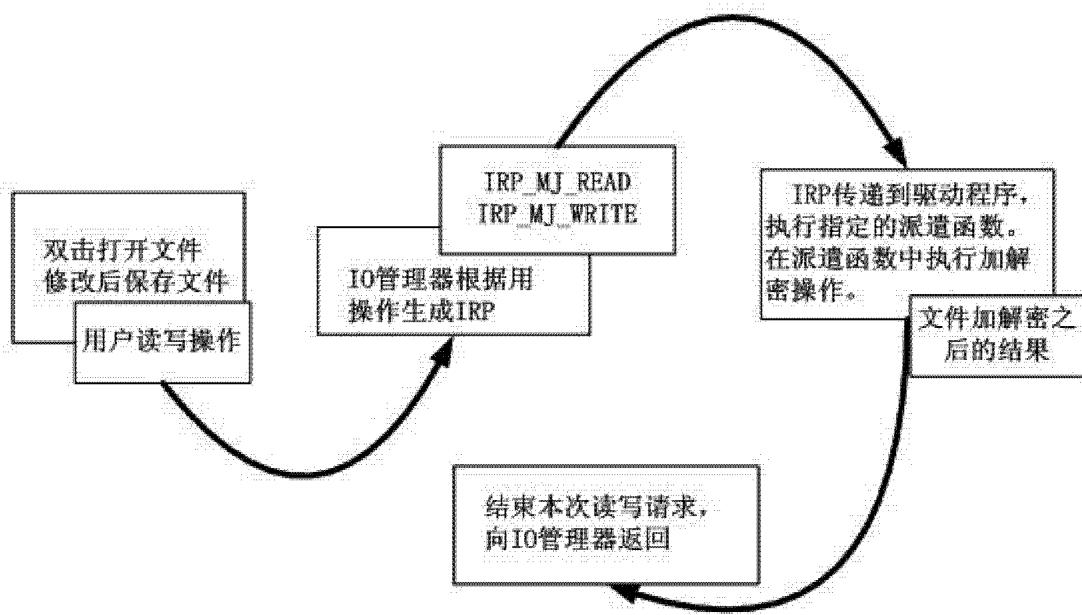


图 8

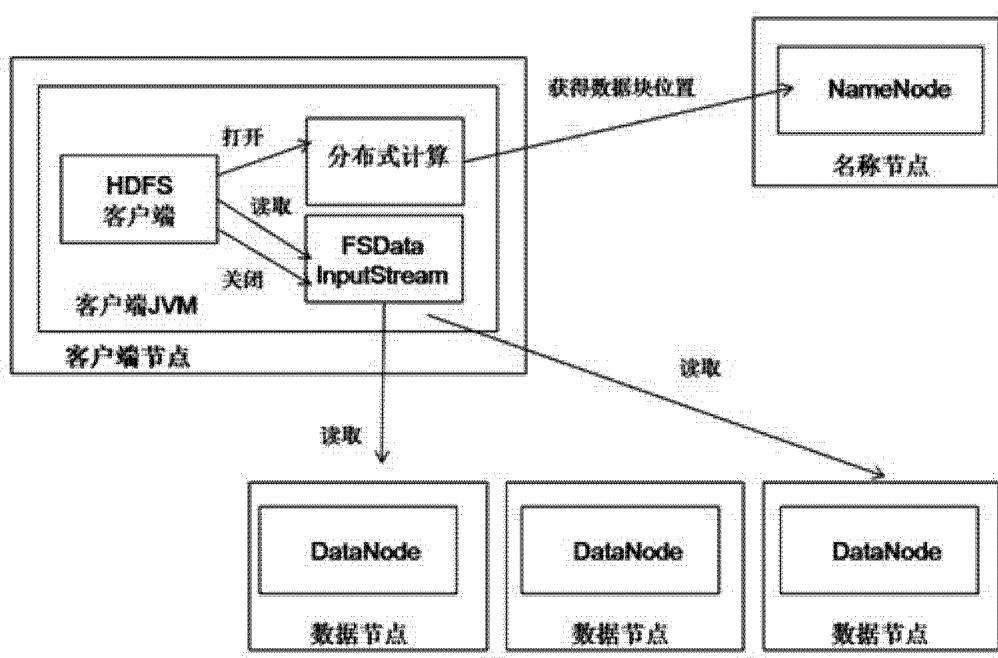


图 9

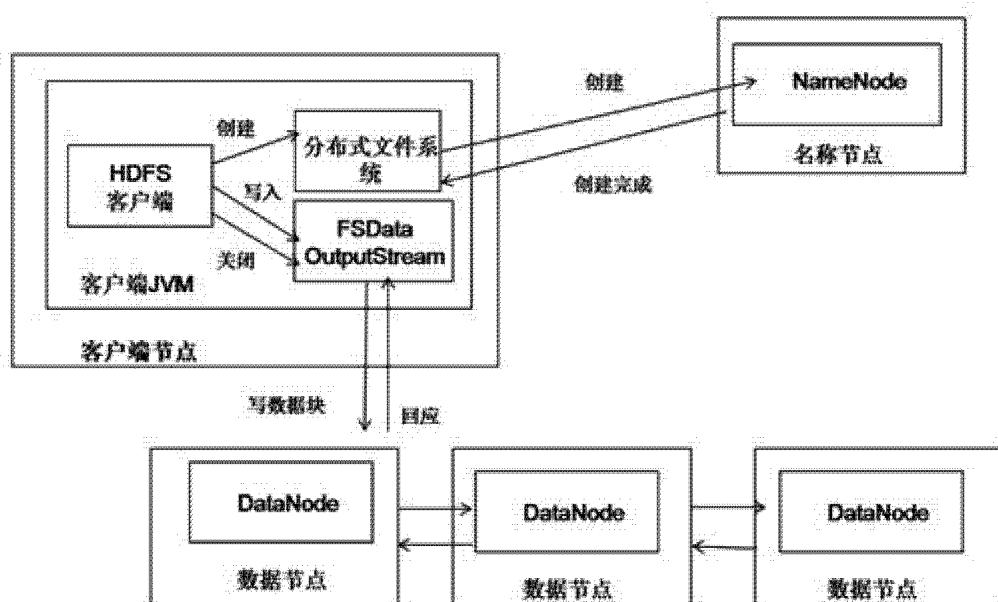


图 10