doi:10.3969/j.issn.1007-855x.2010.01.016

一种基于 P2P的网络存储系统

杨 超
(合肥学院 基础部,安徽 合肥 230601)

摘要: 探讨了一种基于 P2P的网络存储系统的设计与实现方法,针对系统的拓扑结构、资源检索策略、可靠性、安全性问题,提出了解决方案,并对系统的通讯方式进行了详细设计. 初步的实验结果表明,该系统在现实条件下具有较高的可靠性.

关键词: P2P网络;存储系统;门限密码
中图分类号: TP311 文献标识码: A 文章编号: 1007-855X(2010)01-0079-04

A Network Storage System Based on P2P

YANG Chao
(Department of Basic Courses, Hefei University, Hefei 230601, China)

Abstract A way to design and implement a network storage system based on P2P is discussed. Aiming at the problems of the system such as topology, resource retrieval strategy, reliability and security, a solution is put forward. The complete design of communication modes for the system is then given in detail. Its reliability in real setting is proved through preliminary experiments.

Key words P2P; storage system; threshold cryptography

0 引 言

在公用的机房中存储个人文件,往往采用建立网络存储服务器的方法,这种方法重点要解决的是提高中央服务器的存储容量,以此来缓解日益增加的网络资源数据量对网络存储的压力. 随着网络规模的扩大,对中央服务器进行维护和更新的费用将急剧增加,所需成本过高. 论文提出一种基于 P2P的个人文件存储系统的设计与实现方法. 该系统为了充分利用了网络中各台机器的空间,首先对文件进行分割,然后加密签名,再分散存储,具有安全性和机密性的特点,解决了公用机房中建立私有空间的问题或个人存储空间不够的问题.

1 系统功能概述

系统主要应用于同构的局域网络中,将局域网络中可以用来存储文件的机器组成一个组,在任意一台组中机器上可以实现个人文件的存储和获取,系统要充分考虑到个人文件的安全性,保存后的可重获性.

1) 存储文件: 当其中一个成员要存储文件的时侯,系统将首先对文件进行分割,在对分割过的文件块进行加密和签名后,分散存储到每个组成员机器的空间中.

2) 获取文件: 当使用者要求获取自己保存在网络中的文件时,可以取回并且解密验证后合并自己保存的文件块,对于别人保存的文件块将无法解密合并.

2 系统实现的关键问题解决

2.1 系统拓扑结构和资源查找和定位方式

系统采用集中式拓扑结构[1],也就是存在一个中央服务器,在服务器上存储所有系统中当前在线节

收稿日期: 2009-03-18 基金项目: 合肥学院科学研究发展基金项目资助(项目编号: 06KY023ZR)
作者简介: 杨超(1979~),男,硕士,讲师,主要研究方向: 计算机网络,人工智能. E-mail: c_yang5102@hfuu.edu.cn

点的IP地址信息. 当其中一个成员节点要存储或获取文件时, 首先从中央服务器中获取在线节点地址信息, 接下来采用广播方式逐一和每个节点进行连接, 完成存储或获取文件. 图1表示了系统拓扑结构和资源的检索方式.

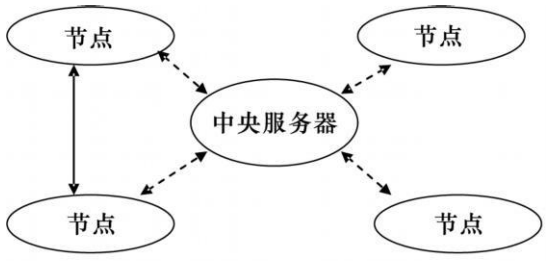

图1 系统拓扑结构和资源检索方式
Fig.1 Topology and resource retrieval strategy of the system

2.2 系统可靠性

系统可靠性是指出现因网络故障、机器故障、磁盘故障等因素和某主机退出系统而导致某个节点失效的情况时, 整个系统仍然能正常工作. 对要存储的数据做一定的冗余, 是在节点失效情况下保证数据持久性的最基本和必要的手段. 副本和纠删码是当前做冗余的主要方式[1]. 这里尝试使用Shamir的(t, n)门限秘密共享方案来实现.

Shamir秘密分享[2,3]是基于多项式插值的门限方案, 它有一个分发者D把秘密s分发给n个参与者, 至少$t \leqslant n$个参与者合作才能重构秘密s. 应用于本系统也就是在存储文件时将文件s分割为n个文件块, 则需要n个节点$p_1, p_2, \cdots, p_n$参与存储, 在获取文件时, 只需要$t \leqslant n$个节点就可以恢复文件s, 这样系统就可以容忍$n-t$台机器不在线的情况下正确使用. 下面将具体介绍该方案的实现.

2.2.1 文件的分割过程

1) 要求存储文件的节点首先在$GF(p)$域选取$t-1$次多项式

$$f(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1} \tag{1}$$

其中$a_0, a_1 \cdots, a_{t-1}$是$GF(p)$域中随机选取的, p是大于$a_0, a_1 \cdots, a_{t-1}$, 也就是分割后的文件块大小最大不能超过p, 该多项式中, 常数项a_0是需要存储的文件s, 也就是将文件转为二进制的整型即可, 并且$a_0, a_1 \cdots, a_{t-1}$保密, $GF(p)$公开.

2) 要求存储文件的节点在$GF(p)$域中随机选取n个不同的点x_i, 其中$1 \leqslant i \leqslant n$. 计算$\text{share}_i(s) = (X_i, f(X_i))$作为文件块信息, 分发给第$i$个参与者$P_i$, 其中$x_i$是公开的值, 为简便起见, 这里选取$x_i = i$, $1 \leqslant i \leqslant n$, 这样得到分割的文件块$f(1), \cdots, f(i), \cdots, f(n)$.

2.2.2 文件的还原过程

文件还原的目标是从n个分割的文件块信息中的合格子集(即任意t个文件块)来恢复目标文件. 设分割的文件块为$f(1), \cdots, f(i), \cdots, f(n)$, 根据Lagrange插值性质, 采用Lagrange插值法重构$t-1$次多项式:

$$f(x) = \sum_{i=1}^{t} f(x_i) \prod_{j=1, j \neq i}^{t} \frac{x - x_j}{x_i - x_j} \tag{2}$$

而恢复的目标文件数据为:

$$s = a_0 = f(0) = \sum_{i=1}^{t} f(x_i) \prod_{j=1, j \neq i}^{t} \frac{-x_j}{x_i - x_j} \tag{3}$$

2.3 安全问题

系统中文件块分散存储, 系统安全面临着巨大的挑战. 为了保证保存的文件只有保存者才能获取, 并且确保文件的正确性和完整性, 系统使用AES算法和DSA算法[2~4]相结合的方法对文件块进行加密和签名处理. 有关上面2个算法原理和实现参见文献[2~4]. 值得注意的是系统可能需要保存2个密钥, 一个AES的密钥用于加密及还原, 一个DSA的密钥对用于签名和验证, 为了减少管理密钥的工作, 从AES和DSA 2个公式的特点出发, 将AES的密钥隐藏在DSA签名之中, 使用时只需保存DSA的密钥对即可, 这样使密钥合2为1, 增加了方便性.

在DSA公式$r = (g^k \bmod p) \pmod q$和$s = k^{-1}(M + xr) \pmod q$中$x$为私钥, 因此当使用者获得签名, 取出其$r, s$2数时, 也可顺便倒推出随机数$k = s^{-1}(M + xr) \pmod q$. 如果在加密过程中, 将AES中的随机数以DSA所产生的随机数带入产生AES密钥, 则在解密时, 可以由DSA中倒推出的k来获得AES密钥, 进行文件的解密. 实际上, 因为DSA的k最大不能超过160位, 而AES却可以支持到256位的加密, 所以上面的做法是可行的.

3 系统通讯过程设计

系统中的每个节点在中央服务器的支持下完成文件的存取. 系统整体通讯架构如图 2所示:

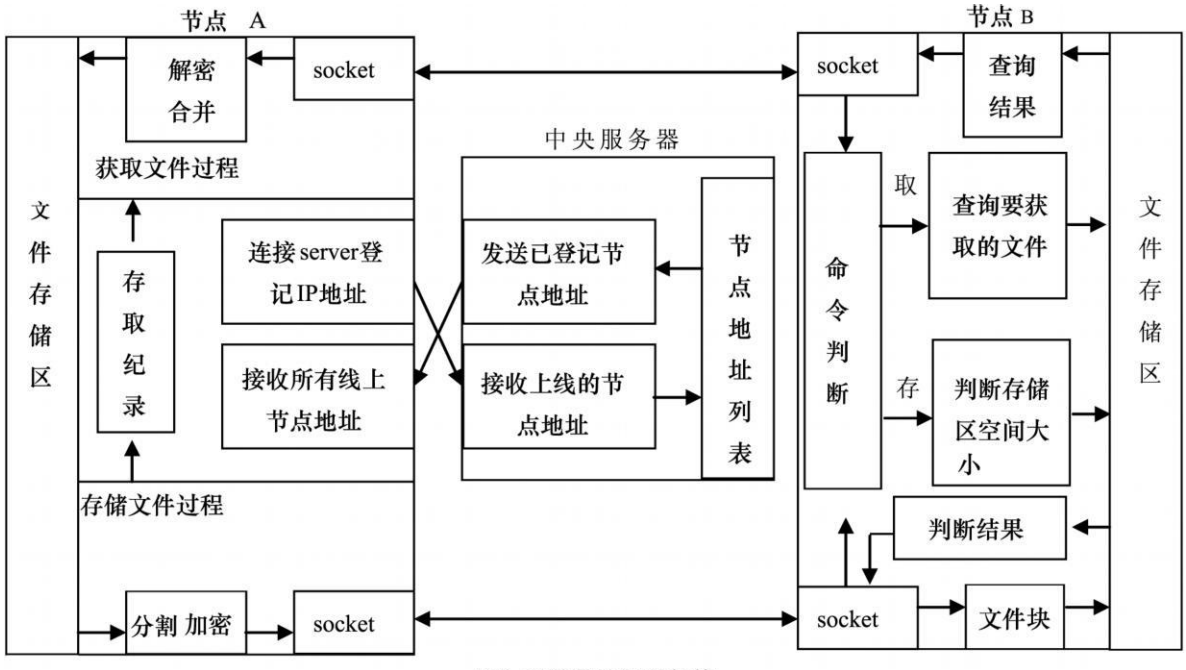

图2 系统整体通讯架构
Fig.2 Communication structure of the system

3.1 中央服务器支持节点过程

中央服务器有一个一直在等待的线程, 用来接收各节点传来的命令, 依照所收到的不同命令, 进行以下 4种不同的操作[5].

1) 节点上线时, 则获取节点 IP地址, 判断地址是否在 IP地址列表中, 如不存在就将该节点的 IP地址存入到列表中.

2) 节点要求存储文件时, 中央服务器从地址列表中选取 IP地址传送给要求节点.

3) 节点要求获取文件时, 中央服务器从地址列表中选取 IP地址以及在线的节点数目传送给要求节点.

4) 节点离开系统时, 中央服务器从地址列表中删除该节点.

3.2 系统中节点之间的通讯过程

节点从中央服务器获得目标节点的地址信息后, 便会和目标节点连线, 形成点对点的数据传输. 每个节点主要有 main, send和 back 3个线程相互合作完成文件的存取[6].

1) main线程 在系统启动时自动执行, 负责等待其他节点的请求和提供其它节点存取文件的服务[7]. 基本的处理流程如图 3所示.

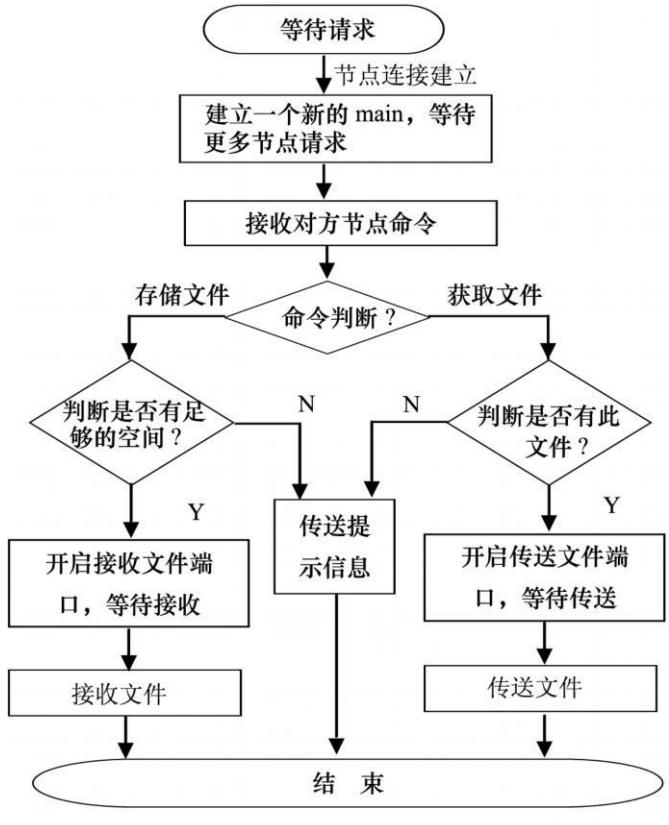

图3 main线程流程图
Fig.3 Flow chart of main thread

2) send线程 节点在进行存储文件时,首先进行文件的分割、加密、签名,然后启动 send线程将分割的文件块分散存储到网络上,其基本的过程如下:

步骤 1 节点连接中央服务器,发送存储文件请求,获取可用存储的节点 IP.

步骤 2 和目标节点进行连接,发送存储请求.

步骤 3 如果目标节点有足够的存储空间,则传送文件块,否则返回步骤 1.

步骤 4 判断是否所有的分割文件块已经存储完毕,如是则结束,否则返回步骤 1.

3) back线程 当使用者要获取自己的文件时,便会启动此线程,负责抓回各个文件块,然后解密、确认、合并获得自己的文件. 其基本的流程如下:

步骤 1 节点连接中央服务器,发送获取文件请求,取得在线节点 IP和总在线节点数.

步骤 2 判断是否访问完所有在线节点,如是则结束,否则和目标节点连接,发送获取文件请求.

步骤 3 如果目标节点中有要求获取的文件,如有则获取文件块,否则换一个节点地址,返回步骤 2

步骤 4 判断获取的文件块数是否满足合并要求,如满足则调用合并程序并结束线程. 否则换一个节点地址,返回步骤 2

4 结果分析

这里利用 java平台初步实现了系统,由于 java本身性能的限制,在执行加密、签名等操作时速度较慢,这里的结果只是对系统的可实现性和具有一定可靠性的证明,并没有对系统速度性能进行分析. 图 4是系统在获取文件过程中进行某个文件块的数据截取图,测试用的文件大小是 1M,图中的 k是随机数.

图4 某个文件块的数据截取图
Fig.4 Data clip picture of a file block

为了测试系统的可靠性,这里先把测试用的文件分发到 20个节点上,然后依次测试当在线节点从 1到 20变化时系统出错的比率,如图 5所示. 结果表明,当在线节点大于 9的时候,系统就具有了相当高的可靠性.

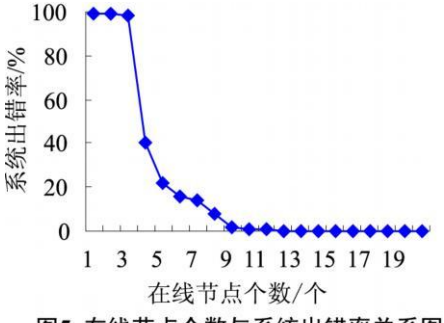

图5 在线节点个数与系统出错率关系图
Fig.5 Diagram of online node number and system error rate

5 结束语

利用 P2P技术实现了一个安全的个人文件网络存储系统,并在解决 P2P存储技术的安全性、可靠性问题上进行了创新,由于采用广播在线节点的方式查找资源,同时设计时没有考虑异构平台上的文件存储,所以本系统只适用于小型的同构局域网. 另外如何很好的管理系统中的冗余文件块[8],是有待今后进一步探讨和研究的问题.

参考文献:

[1] 田敬,代亚非. P2P持久存储研究[J]. 软件学报, 2007, 18(6): 1379-1399.
[2] 商建伟. 门限密码及相关安全应用的研究[D]. 济南: 山东大学, 2007.
[3] 刘蓬涛. 门限密码体制及其在 DSP上的应用研究[D]. 济南: 山东大学, 2007.
[4] 段笑言. 动态(t, n)门限秘密共享的研究与实现[D]. 合肥: 合肥工业大学, 2007.
[5] 张晓, 李战怀, 陈建全. 基于 P2P技术的虚拟文件夹设计与实现[J]. 计算机应用, 2004, 24(12): 138-139.
[6] RichardW S. UNIX环境高级编程[M]. 尤晋元, 刘炀, 俞茂元, 等, 译. 北京: 机械工业出版社, 2000.
[7] 陈明, 杨广文, 刘学铮, 等. 面向点对点的安全可靠存储系统[J]. 软件学报, 2005, 16(10): 1790-1797.
[8] 徐非, 杨广文, 鞠大鹏. 基于 Peer-to-Peer的分布式存储系统设计[J]. 软件学报, 2004, 15(2): 268-277.