KESS元数据处理一致性协议

邓科峰[1] 何连跃[1] 王晓川[1] 周先奉[2]
(国防科学技术大学计算机学院 长沙 410073)[1] (国防科学技术大学继续教育学院 长沙 410073)[2]

摘 要 在麒麟分布式加密存储系统中,分布式元数据处理在系统发生异常时会出现不一致的情况。为了解决这一问题,提出了元数据处理一致性协议2PC-MP。该协议引入事务序号,保证日志记录和消息交互的一致性;增加悬挂队列,避免参与者进程因网络异常而阻塞;增加回退队列,解决用户登录session失效后无法回退的问题;通过分布式日志保证系统故障后的快速恢复。结果表明,2PC-MP协议能够保证元数据处理的一致性和提高系统的性能。

关键词 麒麟,分布式加密存储系统,元数据处理,一致性
中图法分类号 TP309 **文献标识码** A

Consistency Protocol for Metadata Processing in KESS

DENG Ke-feng[1] HE Lian-yue[1] WANG Xiao-chuan[1] ZHOU Xian-feng[2]
(School of Computer Science, National University of Defense Technology, Changsha 410073, China)[1]
(Graduate School, National University of Defense Technology, Changsha 410073, China)[2]

Abstract In Kylin Distributed Encrypting Storage System, data inconsistency may occur during distributed metadata processing under system exception. To solve this problem, the consistency protocol 2PC-MP was presented. In this protocol, transaction number is introduced to ensure the consistency of log records and message interaction, append queue is added to prevent process blocking under network exception, redeem queue is added to allow failed operation to roll back when the user's session gets invalid, and distributed logging technology brings a fast recovery after system failure. Results show that 2PC-MP can ensure the consistency of metadata processing and also enhance its performance.

Keywords Kylin, Distributed encrypting storage system, Metadata processing, Consistency

1 引言

随着信息技术的深入发展,计算机应用逐渐由计算密集型向存储密集型转变,越来越多的应用对存储系统提出了更高性能和更大容量的要求。麒麟分布式加密存储系统(Kylin Distributed Encrypting Storage System, KESS)通过将用户管理请求和数据访问请求相分离,用户管理请求通过元数据服务器集中处理,数据访问请求直接由多个存储服务器并行处理,有效地解决了集中式单服务器系统容量有限和性能瓶颈问题。在KESS中,元数据服务器(Metadata Server, MS)集中管理数据的元信息,包括数据索引、加密密钥、共享者、访问权限等,存储服务器(Storage Server, SS)为了减少数据访问延迟,存储了部分元信息。

用户管理请求通过MS集中进行控制。在元数据的处理过程中,请求经过MS决策后可能分解为子请求,分发给一个或多个SS进行处理。处理结束后,MS根据返回的结果决定是否对相应元数据进行最终的修改。在请求的处理过程中,系统可能出现各种各样的异常情况:不同的SS可能返回不同的操作结果,数据包在松散耦合的网络环境中可能丢失,服务器可能因为某种原因崩溃。当这些情况发生时,就产生了元数据的一致性问题。

目前,有多种机制保证数据的实时一致性,包括数据同步复制、两阶段提交协议2PC(Two Phase Commit)[1,2]等。但上述方法都没有考虑加密存储系统中数据加密的特殊性。本文提出针对分布式加密存储系统元数据处理的两阶段提交协议2PC-MP(Two Phase Commit for Metadata Processing),协议引入事务序号保证日志记录和消息交互的一致性,增加悬挂队列(append queue)避免参与者进程因网络异常而阻塞,增加回退队列(redeem queue)解决用户登录session失效后无法回退的问题,通过分布式日志[3,4]保证系统发生故障后的快速恢复。

2 元数据处理一致性协议2PC-MP

2PC-MP协议采用以MS为主导的元数据处理策略。MS接收到元数据处理请求后,分析该请求需要哪些SS参与,然后依次向它们发送子请求。SS收到请求后,对请求进行前瞻执行,操作完成后向MS返回结果。如果MS收到了所有参与者的Success消息,那么它向用户返回Success消息,否则返回Failed消息。

2.1 元数据处理流程

到稿日期:2009-03-09 返修日期:2009-05-28 本文受国家"八六三"高技术研究发展计划基金项目(2007AA012408)资助。
邓科峰(1985—),男,硕士生,研究方向为信息安全,E-mail:nudtdk@163.com;何连跃 副研究员,硕士生导师,研究方向为操作系统、信息安全;王晓川 博士,助理研究员;周先奉 硕士生,工程师。

MS为涉及到多个节点的元数据处理启动协议。协议启动后,MS首先生成用于本次请求处理的事务序号,然后向每个参与协议的SS发送子请求。请求发出后,MS和SS开始进行元数据处理的消息交互。其流程如图1所示。

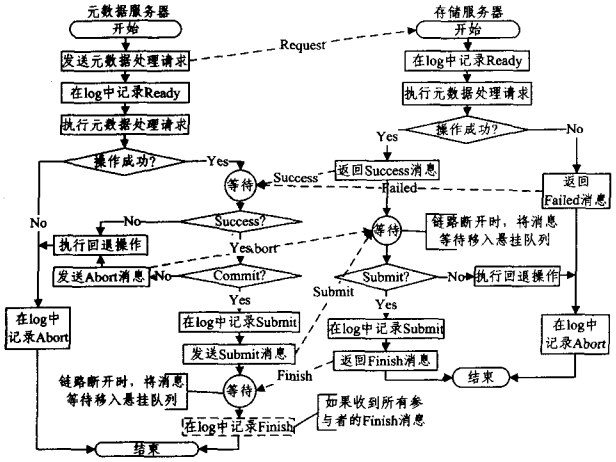

图1 2PC-MP协议元数据处理流程

交互过程分为两个阶段:从MS发送元数据处理请求到等待接收参与者的投票消息为第1阶段;从MS接收到所有参与者的投票消息到在log中强制写入Finish或者Abort记录为第2阶段。具体步骤如下:

1)MS向所有参与此协议的SS发送元数据处理请求;发送完成后,MS在log中强制写入Ready记录,然后开始处理本地元数据。如果处理失败,MS在log中强制写入Abort记录后退出协议;否则,等待SS返回执行结果。

2)SS接收到请求后,首先在log中强制写入Ready记录,然后执行元数据处理请求。如果处理成功,SS向MS返回Success消息;否则,向MS返回Failed消息,并在log中强制写入Abort记录后退出协议。

3)MS等待接收所有参与者的投票消息。如果至少收到一条Failed消息,MS将决定异常中止:向所有返回Success消息的SS发送Abort消息;对本地元数据的修改进行回退;在log中强制写入Abort消息后退出协议。如果所有的投票消息都是Success消息,MS将决定提交:在log中强制写入Submit记录,向每一个参与协议的SS发送Submit消息。

4)每个返回Success消息的SS都在等待接收MS发出的消息。如果SS接收到的是Abort消息,它将回退自己对元数据做出的更改,并在log中强制写入Abort记录,最后终止参与此协议。如果SS收到的是Submit消息,它将在log中强制写入Submit记录,然后向MS返回Finish消息,最后终止参与此协议。

5)MS向所有的参与者发送完Submit消息后,等待SS返回Finish消息。在接收到所有参与此协议的SS的Finish消息后,MS在log中强制写入Finish记录,最后终止参与此协议。

2.2 分布式日志

为了能够在节点崩溃后快速重建崩溃发生时正在执行的任务,系统采用分布式日志技术。在元数据处理的过程中,MS和SS会在本地日志文件中记录请求处理相关的每个重要事件。由于请求处理的日志记录分布在MS以及参与此请求处理的SS的日志文件中,因此恢复时需要相关的节点根据各自的日志协商请求的完成情况。由于节点可能在某一时刻同时执行多个元数据处理请求,因此日志文件中不同请求的记录可能相互交错。为了快速区分不同请求的日志记录,分布式日志在记录中加入了事务序号。

日志记录采用变长格式,如图2所示。记录的类型有4种:Ready记录、Submit记录、Abort记录、Finish记录。其中,Abort记录、Finish记录以及SS的Submit记录的附加信息域为空;Ready记录的附加信息域为请求处理的类型以及参数;MS的Submit记录的附加信息域为参与协议的SS的列表。

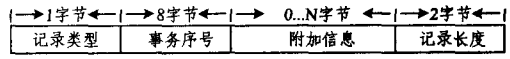

图2 日志记录格式

为了保证日志记录写入后及时地刷新到磁盘上,日志文件使用了同步写入方式。为了避免日志文件无限制地增长,需要定期地对日志文件进行整理。在整理过程中,系统使用一个临时的日志文件暂存正在处理的请求的日志记录,整理完成后再将两个文件进行合并。

2.3 悬挂队列

在链路断开的情况下(节点失效或者链路过于拥塞),活动节点(MS或者SS)消息等待的实现是非阻塞的。消息等待的地方有3处:MS发送元数据处理请求后等待SS返回Success消息;SS返回Success消息后等待MS发送Submit或者Abort消息;MS发送Submit消息后会等待SS返回Finish消息。如果在第1种情况下连接断开,那么MS直接进行Abort处理;其他情况下,节点将未完成的消息交互转到悬挂队列中。

MS悬挂队列元素由事务序号以及SS地址构成,如图3所示。当MS等待SS的Finish消息时连接断开,它将用此事务序号和SS的地址初始化一个结构元素并将其加入队列尾部。当MS和某个SS之间的链路恢复正常时,MS会向队列中所有涉及此SS的成员依次发送由事务序号标示的请求的Submit消息,继续协议的执行。

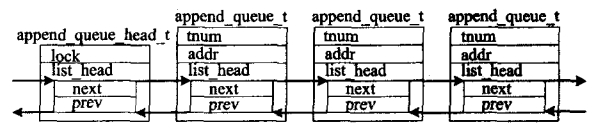

图3 MS悬挂队列结构

SS悬挂队列元素由事务序号、请求信息以及用户名组成,如图4所示。当SS等待MS发送Submit或者Abort消息时连接断开,SS将使用相应的信息生成一个结构元素并将其插入队列尾部。连接恢复时,SS向MS查询悬挂队列中事务序号所标示的请求的处理状态,如果它收到的是Submit消息,则在日志中写入Submit记录,否则将队列元素移动到回退队列中。

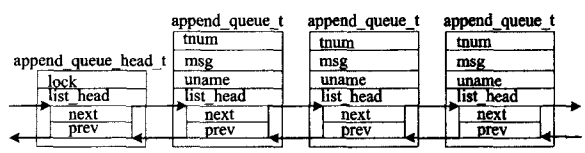

图4 SS悬挂队列结构

2.4 回退队列

用户只能对自己的元数据进行修改,因此要对元数据进

行操作首先需要登录系统。然而，节点崩溃、网络断开的时间太久或者网络断开时用户退出系统，都将导致用户登录 session 失效。当这些情况发生时，无论是 MS 还是 SS 都无法对操作进行回退。为了解决这一问题，系统引入了回退队列来暂存这些需要回退的操作，队列结构如图 5 所示。

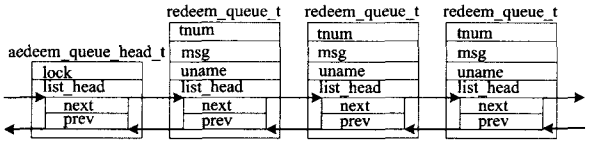

图 5　回退队列结构

用户重新登录系统时，认证模块在用户身份认证完成后激活对回退队列的处理。回退队列处理例程从队列头开始遍历队列，查找与当前登录用户名匹配的项。一旦找到匹配项，就依据 msg 域记录的操作类型、参数等信息进行回退处理。队列先进后出特性保证了用户的后一个操作先于前一个操作被回退，因此从队列头开始的回退处理保证了元数据能够依次回退到原始的状态。

3　2PC-MP 故障恢复

在恢复中假设故障只有节点失效和网络异常的情况，日志记录写入的原子性由同步日志方式保证，元数据修改操作的原子性由节点本身的事务机制保证。由于采用 undo 日志记录的设计方式，回退操作复原的始终是元数据的原始属性，因此回退操作是幂等的，在恢复过程中允许系统再次异常。

1）MS 发送元数据处理请求时超时

MS 关闭和 SS 之间的连接，终止协议的执行。收到元数据处理请求的 SS 在执行完请求发送处理结果后将发现连接已经断开，于是它对操作进行回退，同时在 log 中写入 Abort 记录，最后终止参与此协议。由于 MS 没有对元数据做任何处理，也没有记录任何日志，收到请求的 SS 会对元数据的修改自动进行回退，同时写入 Abort 日志记录。因此，这一情况下日志记录是完整的，元数据是一致的。

2）MS 等待投票消息时超时

MS 对操作进行回退，在 log 中写入 Abort 记录，向返回 Success 消息的 SS 发送 Abort 消息，关闭和 SS 之间的连接，终止协议的执行。发送 Success 消息的 SS 将收到 Abort 消息，然后对操作进行回退，同时在 log 中写入 Abort 记录，最后终止参与此协议。其他的 SS 在执行完请求发送处理结果后将发现连接已经断开，于是将对操作进行回退，同时在 log 中写入 Abort 记录，最后终止参与此协议。由于 MS 和 SS 均会对元数据的修改进行回退并写入 Abort 日志记录，因此这一情况下日志记录是完整的，元数据也是一致的。

3）SS 等待 Submit 或者 Abort 消息时超时

由于 SS 不知道 MS 是决定 Submit 还是 Abort，因此当这种情况发生时，SS 进入不确定的状态。此时，SS 将消息等待转移到悬挂队列中，然后退出协议的执行。

4）MS 等待 Finish 消息时超时

MS 将消息等待转移到悬挂队列中，继续等待接受其他 SS 发出的 Finish 消息。

5）MS 崩溃

当 MS 崩溃后重新启动时，它将由后向前搜索其日志。

（1）搜索到的是 Abort 或者 Finish 记录，则在匹配数组中添加其事务序号。

（2）搜索到的是 Submit 记录。如果事务序号不在匹配数组中，那么它必定处于协议的第 2 阶段。此时，MS 根据 Submit 记录中的 SS 列表生成 Finish 消息等待，并将它们加入悬挂队列中，然后在匹配数组中添加此事务序号。其他情况不进行处理。

（3）找到的是 Ready 记录。如果事务序号不在匹配数组中，那么它必定处于协议的第 1 阶段。此时，MS 根据 Ready 记录中的操作类型和参数生成对应的回退操作，将其加入到回退队列中。其他情况不进行处理。

6）SS 崩溃

当 SS 崩溃后重新启动时，它将由后向前搜索其日志。

（1）搜索到的是 Abort 或者 Submit 记录，则在匹配数组中添加其事务序号。

（2）找到的是 Ready 记录。如果事务序号不在匹配数组中，那么 SS 在崩溃发生时可能在等待 MS 的 Submit 或者 Abort 消息，于是 SS 生成 Ready 消息等待，将其加入到悬挂队列中。其他情况不进行处理。

结束语　分布式加密存储技术集成了加解密技术和分布存储技术，是当前信息安全领域新的研究方向和热点。KESS 采用元数据集中管理、文件数据分散存储的带外管理机制，满足了系统对高性能、大容量的要求。由于加解密的原因，部分元数据同时需要存储在 SS 上。为了解决这些元数据处理时引发的一致性问题，提出了 2PC-MP 协议。另外，通过模拟各种网络异常和节点故障对协议进行了测试。结果表明，协议在节点失效或网络异常恢复后，能够保证元数据的一致性，提高分布式元数据处理的性能。

参 考 文 献

[1] Tanenbaum A S, van Steen M. Distribute Systems Principles and Paradiams[M]. Beijing: TsingHua University Press, 2004: 307-312

[2] Garcia-Molina H, Ullman J D, Widom J. Database System Implementation[M]. Beijing: China Machine Press, 2001: 310-318

[3] Lewis P M, Bernstein A, Kifer M. Database and Transaction Processing[M]. Beijing: China Machine Press, 2005: 655-661

[4] Gray J, Reuter A. Transaction Processing : Concepts and Techniques[M]. Beijing: China Machine Press, 2004: 357-378

（上接第 52 页）

[12] Ma L, Barner K E, Arce G R. Satistical Analysis of TCP's Retransmission Timeout Algorithm[J]. IEEE/ACM Transactions on Networking, 2006, 14(2): 383-396

[13] 田东，陈蜀宇，陈峰. 一种网格环境下的动态故障检测算法[J]. 计算机研究与发展，2006，43(11)：1870-1875

[14] Hayashibara N, D'efago X, Yared R, et al. The f accrual failure detector[C]// SRDS IEEE Computer Society. 2004: 66-78

[15] Satzger B, Pietzowski A, Trumler W, et al. A Lazy Monitoring Approach for Heartbeat-Style Failure Detectors[C]// Proceedings of the Third International Conference on Availability, Reliability and Security. 2008: 404-409

[16] 邓聚龙. 灰色预测与决策[M]. 武汉：华中理工大学出版社，1988

[17] 吉培荣，胡翔勇，熊冬青. 对灰色预测模型的分析与评价[J]. 水电能源科学，1999(2)：42-441