ISSN 1004-9037,CODEN SCYCE4
Journal of Data Acquisition and Processing Vol. 35,No. 5,Sep. 2020,pp. 834－841
DOI:10. 16337/j. 1004-9037. 2020. 05. 004

http://sjcj. nuaa. edu. cn
E-mail:sjcj@nuaa. edu. cn
Tel/Fax:＋86-025-84892742

基于分层哈希编号的智能制造产线数据同步方法

燕雪峰，丁　叶

（南京航空航天大学计算机科学与技术学院/人工智能学院，南京，211106）

摘　要：海量数据存储和同步是智能制造产线中的重要问题。当前数据同步的主流方法是远程同步（Remote synchronization，RSYNC）算法，采用同步增量数据的方法减少数据传输量。智能制造产线产生的数据层次结构深、目录结构复杂，导致同步时评估时间长。为此提出分层哈希编号算法进行同步，基于数据分层对数据文件编号，使用散列表记录层次信息，快速比对差异数据，并对不同类型的差异数据采用不同备份策略。实验结果表明，与标准RSYNC相比，该方法有效减少了RSYNC评估的数据量，有效降低了同步时间，提高了同步备份效率。

关键词：远程备份；文件同步；RSYNC；分层哈希编号表；数据灾备

中图分类号：TP309　　**文献标志码：**A

Data Synchronization Method of Intelligent Manufacturing Production Line Based on Hierarchical Hash Number

YAN Xuefeng，DING Ye

（College of Computer Science and Technology / College of Artificial Intelligence，Nanjing University of Aeronautics and Astronautics，Nanjing，211106，China）

Abstract：Mass data storage and synchronization are important issues in intelligent manufacturing production lines. The current mainstream method of data synchronization is remote synchronization（RSYNC）algorithm. It uses the method of synchronizing incremental data to reduce the amount of data transmission. The data generated by the intelligent manufacturing line has a deep hierarchical structure and a complex directory structure，thus leading to a long evaluation time during synchronization. Here，a hierarchical Hash numbering algorithm is proposed for synchronization based on data hierarchical numbering of data files，by using a Hash table to record hierarchical information，quickly comparing differential data，and adopting different backup strategies for different types of differential data. Experimental results show that compared with standard RSYNC，the proposed method effectively reduces the amount of data evaluated by RSYNC and synchronization time，which also improves synchronization backup efficiency.

Key words: remote backup; file synchronization; remote synchronization (RSYNC); hierarchical hash numbering table; data disaster recovery

---

基金项目：国家重点研究计划（2018YFB1702700）资助项目。
收稿日期：2020-07-31；**修订日期：**2020-08-18

引　言

随着大数据、云计算等技术的不断发展,数据成为了贯穿整个智能制造过程的关键因素[1]。以数据为核心,从智能车间生产过程产生的海量数据中挖掘有价值的信息来指导车间运行优化,已经引起学术界和工业界的极大关注[2]。在智能制造领域需要进行数据采集及存储,数据包括装配数据、建模数据、产线数据等,这些数据对于智能制造生产发挥重要作用,为防止这些重要数据的丢失造成损失就需要对生产数据进行保护。远程文件同步[3-5]作为其中的一个重要的数据保护手段受到了广泛研究,其主要原理是通过将生产机上的文件数据同步备份到异地灾备机来实现容灾。远程文件同步的核心是数据同步,高效的同步方法需要在数据正确的情况下,尽可能地降低时间、存储等资源的消耗,所以研究高效的同步算法十分必要。

在过去几十年的研究中,研究者提出了远程同步(Remote synchronization, RSYNC)方法,它是数据同步的常用工具[6-8],该方法只对差异数据进行传输,提高了同步效率,是目前通用的文件同步系统[9],适合低带宽下的同步[10]。但是该方法需要对所有同步数据进行分块,对比校验值,计算文件差异块,在文件变化数量少或新增文件较多时,消耗了大量时间。RSYNC算法如图1所示,服务器与主机相连,在服务器端对数据进行固定分块,并计算各数据块的强、弱校验码,形成以弱校验码为关键字的哈希表发送给主机。主机接受哈希表后,按照同样的分块长度对主机数据进行分块,并计算强、弱校验码,通过比对哈希值找到差异数据并将差异数据块发送给服务器。

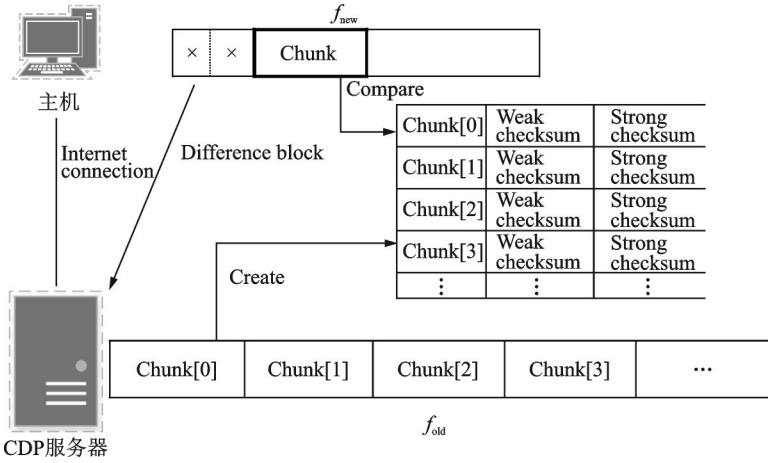


图1　RSYNC算法

Fig.1　RSYNC algorithm

目前的研究主要对RSYNC进行预处理。张静等[11]提出的两轮差异数据同步算法,根据差异数据块的比例,在比例较低,即差异数据块数量较少时,直接用内容可变长度分块(Content-defined chunking, CDC)[12]算法计算差异数据块内容得到差异数据,在比例较大时仍采用RSYNC精确查询进行同步,这种方法只有在数据差异不大的情况下效果较好,在数据差异较大时还是使用了RSYNC算法进行同步,效果不明显。针对上述问题,王青松等[13]提出了Winnowing指纹串匹配的重复数据删除算法,此方法在数据分块前引入分块大小预测模型,解决CDC中分块大小难以控制的问题,并使用ASCII/Unicode编码方式作为数据块,减少指纹计算开销,重删率提升了10%左右,在指纹计算和对比开销方面减少了18%左右。Ghobadi等[14]提出指纹预处理目录结构方法,对原目录的层次结构采用层次信息、文件名以及文件大小进行编号的方式,快速查找出需要采用RSYNC进行评估的文件,这种方法可以减少评

估的文件数量,但由于编号使用了文件大小作为其中的对比依据,造成数据发生变化而大小未变情况下的对比错误,同时该算法因为编码函数的限制只能识别修改文件,无法识别新增文件。李帅等[15]提出了基于目录哈希树(Directory hash tree, DHT)的数据同步方法,该方法在保持与原磁盘目录树拓扑结构一致的条件下,通过利用DHT能够快速确定文件的异同,并对差异文件使用RSYNC同步算法,从而实现数据同步。但该方法中父节点的Hash值采用将所有子节点Hash值相加,增大碰撞概率,并且由于DHT结构复杂,造成构建时间较长,影响同步效率。Hu等[16]提出使用多因素身份验证的增强型安全备份方案,来加密同步数据,防止某些敏感数据被拦截,提高数据传输的安全性。Zhang等[17]设计并实现了一个安全性得到增强的数据备份和恢复系统,此系统在原有备份恢复系统中加入基于数字证书的身份验证,提高安全性能。任燕博等[18]提出借助文件系统监控机制同步数据的方法,通过Inotify机制监控文件目录下的文件变动情况,提高同步效率。Yang等[19]设计基于文件系统的增量备份,对文件内容采用Diff算法进行数据库增量备份,降低数据的传输量。

本文针对上述问题,采用MD5信息摘要算法(Message digest algorithm, MDA)[20]作为编号中的识别码同时使用更为简单的层次结构来构建对比表。通过比较文件编号可以快速识别变化文件,并根据文件的变化类型,选择合适的备份策略。对于修改文件采取增量同步备份策略,而对于新增文件采用完全同步策略。

1 基于分层哈希编号算法的数据同步模型

基于分层哈希编号算法的数据同步模型由2部分组成:(1)服务器根据目录的层次结构,构建分层哈希编号表(Hierarchical hash numbering algorithm, HHNT),并传输该文件;(2)客户端接收服务器发送的HHNT,根据目录层次结构分层对比差异文件。

1.1 HHNT的构建

根据目录层次结构,通过编号函数构建每个文件的唯一编号,编号函数为

$$C = \text{FilePath}*F*(\text{Identifier}) \tag{1}$$

式中,FilePath表示父文件的路径,F为文件或文件夹名称,Identifier为文件或文件夹的识别码。如果是文件夹,则设为-1,如果是文件,则设为此文件的MD5值。由于使用了MD5值作为文件的识别码,所以文件编号会随着文件变化而改变,解决了文件内容变化而文件大小不变造成的文件对比错误问题。HHNT构建算法如下。

输入:文件夹路径
输出:分层目录哈希表
Begin
  for QueIsNotEmpty() do
    获取队列头结点,N_i
      if N_i is 文件
        计算N_i的MD5值作为N_i的识别码
      end if
      if N_i is 文件夹
        使用“-1”作为N_i的识别码
        将N_i增加到队列的队尾
      end if
  end for

end

构建分层的哈希编号表,其具体步骤如下:

步骤1 根据输入的目录,对根节点进行编号并加入到队列中,队列用来记录还未编号节点;

步骤2 检测队列是否为空,若为空即没有需要编号的节点,返回HHNT,若不为空执行步骤3;

步骤3 取出队头元素,计算此节点的识别码,如果此节点是文件,计算其MD5值作为其识别码,若为文件夹,“-1”作为其识别码,并将其子文件加入队尾,执行步骤2。

1.2 对比同步

当目录发生改变时,根据层次结构,对比相应层的节点编号,快速区分出变化文件,并加入不同的同步队列中,差异文件对比算法如下。

输入:改变后的文件夹及上个版本的HHNT

输出:完全同步队列以及增量同步队列

```
Begin
    for ComQueIsNotEmpty() do
      获取队列头结点,Ni
      获取对应层的编号集合,N[level]
      if Ni is 文件
        if 在N[level]找到Ni的前缀码
          计算 Ni 的哈希值, Hashi=MD5(path of Ni)
          if Hashi 匹配
            continue
          else
            将 Ni 加入增量备份队列
          end if
        else
            if在N[level]未找到Ni的前缀码
            计算 Ni 的哈希值并修改编号, Hashi=MD5(path of Ni)
            将 Ni 加入完全备份队列
          end if
      else if Ni is 文件夹
        if 在N[level]找到Ni的前缀码
            continue
        else
            计算 Ni 以及 Ni 子文件的目录编号
            将Ni 以及 Ni 子文件加入完全备份队列
        end if
      end if
      end for
        end
```

对比同步步骤如下。

步骤1 将目录根节点加入队列中,队列用来记录待对比的节点。

步骤2 若队列不为空,取出队头节点,根据层次信息,从HHNT中取出对应层的编号集合,若为空则输出RSYNC以及完全同步(Full synchronization, FSYNC)队列。

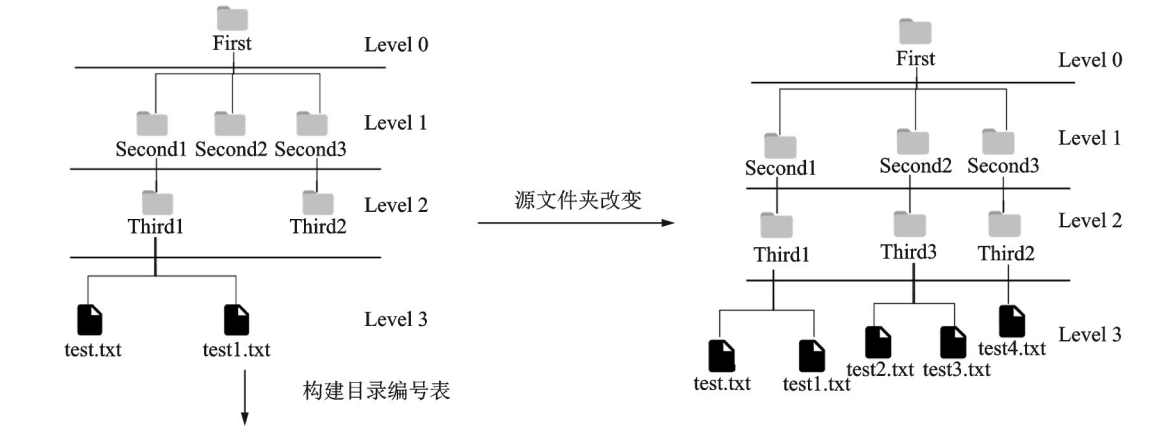
步骤3 如果节点是文件夹,执行步骤4,节点是文件执行步骤5。

步骤4 遍历集合,根据节点的路径和编号的前缀码对比结果,判断是否为新增文件夹,若为新增文件夹,执行步骤6;若不是则将此文件的子文件加入到队列中,执行步骤2。

步骤5 遍历集合,将节点的路径和编号的前缀码对比,若匹配,表示此文件存在,计算节点MD5值和编号中的识别码对比,若对比失败,则表示此文件为修改文件,将此文件加入RSYNC队列,若前缀码对比失败,将其加入FSYNC队列,执行步骤2。

步骤6 通过编号算法,将此文件夹进行编号,添加到HHNT中,并将此文件夹加入FSYNC队列。

根据分层哈希编号算法,可以快速分离出变化文件,如图2所示。与DHT算法相比,在构建过程中不需要使用复杂的树结构以及计算文件夹节点的哈希值,减少时间消耗,同时降低了文件夹节点的碰撞概率。



Level	Parent path	Folder or file
0	../root	first*first*(-1)
1	../first	first/second1*second1*(-1)
		first/second2*second2*(-1)
		first/second*second*(-1)
2	../first/second1	first/second1/third1*third1*(-1)
	../first/second3	first/second1/third2*third2*(-1)
3	../first/second1/third1	first/second1/third1/test.txt*test.txt*(202cb962ac5975b964b7152d234b70)
		first/second1/third1/test1.txt*test1.txt*(90150983cd24fb0d6963f7d28e17f72)

更新编号表

Level	Parent path	Folder or file
0	../root	first*first*(-1)
1	../first	first/second1* second1*(-1)
		first/second2* second2*(-1)
		first/second* second*(-1)
2	../first/second1	first/second1/third1* third1*(-1)
	../first/second3	first/second1/third2* third2*(-1)
	../first/second2	first/second1/third3*third3*(-1)
3	../first/second1/third1	first/second1/third1/test.txt*test.txt*(202cb962ac5975b964b7152d234b70)
		first/second1/third1/test1.txt*test1.txt*(90150983cd24fb0d6963f7d28e17f72)
	../first/second1/third3	first/second1/third3/test2.txt*test2.txt*(0744d493e0c08f8d17596cc3ab96ee0c)
		first/second1/third3/test3.txt*test3.txt*(c3cfaee35155cf0540761b2045ea1e7e)
	../first/second1/third2	first/second1/third2/test4.txt*test4.txt*(28214b307f7969b211a00be0a870e51f)

图2 分层文件编号示例

Fig.2 Example of hierarchical file numbering

2 分层哈希编号算法在智能制造中的应用

基于分层哈希编号算法应用在RSYNC数据同步之前,根据智能制造产线数据的层次目录结构,构建HHNT。当产线数据发生变化时,对比上个版本的HHNT即可完成文件同步。图3展示了整体框架流程图,服务器对智能制造产线数据构建HHNT并发送给客户端,客户端根据此表分层对比文件编号,分离出变化文件,并对变化文件使用不同的备份策略进行同步。

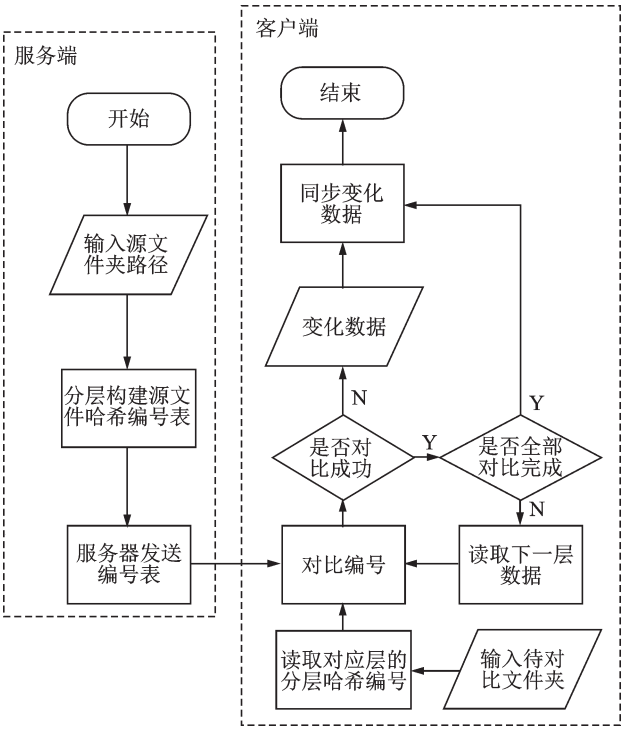


图3 框架流程图

Fig.3 Framework flow chart

由上述可知,本文方法根据比对HHNT判断结果进行文件同步。若文件是未变化文件,则无需对该文件进行分块、校验值计算、校验值查找验证等操作,解决了RSYNC在文件未变情况下还需进行分块、校验值计算对比的资源浪费问题。当文件是改变文件时,根据HHNT中的编号信息进一步判断是否是新增文件,如果是新增文件,同样跳过RSYNC评估,进一步减少RSYNC评估的文件数量,降低资源以及时间消耗。该方法作为RSYNC同步的一个预处理过程,可以有效减少使用RSYNC评估的文件夹和文件的数量。如果不使用预处理方法,标准的RSYNC将对整个文件结构进行遍历评估,在有限数据发生改变的情况下,这将是不必要的开销。

3 仿真与实验分析

为了验证上述分层哈希编号算法的有效性,本文对磁盘在不同变化程度情况下的同步时间进行对比。同步时间是同步算法最重要的评价标准,本文中的同步时间采用系统时间函数,计算同步进程所用时间。

本文使用1 GB的智能制造产线数据进行实验,该环境包含19个层次级别,共10 000个文件。分别记录无差同步、全同步、差异同步3种情况下的同步时间。实验结果如图4所示。

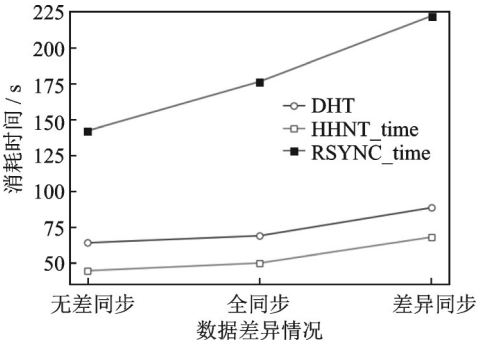


图4 同步时间比较

Fig.4 Comparison of synchronization time

图4中,在无差同步情况下,本文方法不像标准RSYNC那样使用精准查询找到差异数据块,只对目录构建HHNT记录文件编号便可完成同步,所以在时间消耗上效果明显,特别是当数据量越大时,所用时间相差越明显。在全同步情况下,即包含大量新增文件时,从图4中可以看出与无差同步相比,本文方法只增加了很短时间便完成同步,而标准方法却直线上升,这是由于本文方法不需要对这些

新增文件再使用RSYNC算法，只需花费很短的时间对比分层目录哈希表并更新即可。在差异同步情况下，本文方法只对修改文件使用了RSYNC进行精确查询，所以时间消耗变大，但是这与标准RSYNC对全目录进行精确查询相比，在时间消耗上还是有很大改善的。从整体上来看，标准RSYNC算法消耗的时间上升趋势较快，而本文方法只有当差异数据较多情况下，时间消耗才会有较明显的上升。

综上，在智能制造产线数据变化有限的情况下，本文的预处理方法与标准RSYNC相比可极大降低同步时间；与目录哈希表方法相比，本文方法可以避免哈希碰撞引起的文件同步错误问题，且在同步时间上提高了30%左右。

4 结束语

本文提出的分层哈希编号方法，作为RSYNC的预处理方法，结合目录层次结构进行编号，通过比对编号找出修改文件，对修改文件使用RSYNC进行同步，其他变化文件使用完全同步，有效降低评估文件数量，在时间消耗上获得较大提升。当然本方法还存在一定的不足，由于该方法仍需遍历对比整个文件夹的编号，所以未来需要设计一个识别文件层次的算法，快速定位到文件变化的目录层级，优化对比算法，进一步提高同步效率。同时，在大量新增文件的情况下，完全同步中的网络资源消耗仍是一个需要优化的问题，需要寻找合适的压缩算法解决该问题。

参考文献：

[1] 郭磊，陈兴玉，张燕龙，等．面向智能制造终端的车间生产数据采集与传输方法[J]．机械与电子，2019，37(8)：21-24.
GUO Lei, CHEN Xingyu, ZHANG Yanlong, et al. Workshop production data collection and transmission method for intelligent manufacturing terminals[J]. Machinery and Electronics, 2019, 37(8): 21-24.

[2] 张洁，高亮，秦威，等．大数据驱动的智能车间运行分析与决策方法体系[J]．计算机集成制造系统，2016，22(5)：1220-1228.
ZHANG Jie, GAO Liang, QIN Wei, et al. Big data driven intelligent workshop operation analysis and decision method system [J]. Computer Integrated Manufacturing Systems, 2016, 22(5): 1220-1228.

[3] 周丽丽，赵时轮．远程文件同步技术应用分析[J]．中国新通信，2018，20(24)：57-58.
ZHOU Lili, ZHAO Shilun. Application analysis of remote file synchronization technology[J]. China New Telecommunications, 2018, 20(24): 57-58.

[4] 何骞，卓碧华．一种远程文件同步方法[J]．计算机应用，2012，32(2)：566-568.
HE Qian, ZHUO Bihua. A remote file synchronization method[J]. Computer Applications, 2012, 32(2): 566-568.

[5] 胡晓勤，卢正添，刘晓洁，等．远程文件快速同步方法[J]．电子科技大学学报，2008(4)：594-597.
HU Xiaoqin, LU Zhengtian, LIU Xiaojie, et al. Fast synchronization method of remote files[J]. Journal of University of Electronic Science and Technology of China, 2008(4): 594-597.

[6] FRIDRICH J, GOLJAN M. Robust Hash functions for digital watermarking[C]// Proceedings of IEEE Int Conf Information Technology: Coding Computing. [S.l.]: IEEE, 2000: 178-183.

[7] COSKUN B, SANKUR B. Robust video hash extraction[C]// Proceedings of the IEEE 12th Signal Processing and Communications Applications Conference. Kusadasi, Turkey: IEEE, 2004: 292-295.

[8] HAN Y J, PARK M W, KIM J M, et al. Design and implementation of end-to-end network performance measurement and diagnosis system for high-speed networks [J]. Communications in Computer and Information Science, 2009, 56: 201.

[9] 杨小龙，李涛，刘晓洁，等．基于差异的文件同步系统的设计和实现[J]．微计算机信息，2009，25(9)：67-69.
YANG Xiaolong, LI Tao, LIU Xiaojie, et al. Design and implementation of file synchronization system based on differences [J]. Microcomputer Information, 2009, 25(9): 67-69.

[10] 梁丽木，刘晓洁，胡晓勤，等．一种低带宽网络文件同步方法的设计与实现[J]．四川大学学报(自然科学版)，2011，48(1)：55-60.
LIANG Limu, LIU Xiaojie, HU Xiaoqin, et al. Design and implementation of a low-bandwidth network file synchronization method[J]. Journal of Sichuan University (Natural Science Edition), 2011, 48(1): 55-60.

[11] 张静，王少奎，郝希亮，等．一种基于RSYNC算法的改进型两轮同步算法[J]．数据通信，2017，28(1)：52-54.

ZHANG Jing, WANG Shaokui, HAO Xiliang, et al. An improved two-wheel snchronization algorithm based on RSYNC algorithm[J]. Data Communications, 2017, 28(1): 52-54.

[12] ZHANG C, QI D, LI W, et al. Function of content defined chunking algorithms in incremental synchronization[J]. IEEE Access, 2020, 8: 5316-5330.

[13] 王青松,葛慧.Winnowing指纹串匹配的重复数据删除算法[J].计算机应用,2018, 38(3): 677-681, 714.
WANG Qingsong, GE Hui. Data deduplication algorithm for winnowing fingerprint string matching[J]. Computer Applications, 2018, 38(3): 677-681, 714.

[14] GHOBADI A, MAHDIZADEH E H, KEE Y L, et al. Pre-processing directory structure for improved RSYNC transfer performance[C]//Proceedings of IEEE 2011 13th International Conference on Advanced Communication Technology (ICACT). Seoul, South Korea: IEEE, 2011: 1043-1048.

[15] 李帅,刘晓洁,徐兵.一种基于目录哈希树的磁盘数据同步方法研究[J].信息网络安全,2019(2): 53-59.
LI Shuai, LIU Xiaojie, XU Bing. Research on a disk data synchronization method based on directory hash tree[J]. Information Network Security, 2019(2): 53-59.

[16] HU H, LIN C, CHANG C C, et al. Enhanced secure data backup scheme using multi-factor authentication[J]. IET Information Security, 2019, 13(6): 649-658.

[17] ZHANG J , LI H. Research and implementation of a data backup and recovery system for important business areas[C]//Proceedings of 2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC). Hangzhou: [s.n.], 2017: 432-437.

[18] 任燕博,刘钊远.基于RSYNC远程同步系统的优化[J].计算机与数字工程,2014,42(6): 1007-1010.
REN Yanbo, LIU Zhaoyuan. Optimization based on RSYNC remote synchronization system[J]. Computer and Digital Engineering, 2014, 42(6): 1007-1010.

[19] YANG Chao, XU Wen , WU Guohui , et al. Incremental local data backup system based on bacula[C]//Proceedings of 2018 IEEE International Conference of Safety Produce Informatization (IICSPI). Chongqing, China: [s.n.], 2018: 429-432.

[20] 靳燕.基于MD5算法的文件完整性检测系统分析及设计[J].网络安全技术与应用,2019(11): 36-38.
JIN Yan. Analysis and design of file integrity detection system based on MD5 algorithm[J]. Network Security Technology and Application, 2019(11): 36-38.

作者简介:

燕雪峰(1975-),男,教授,博士生导师,研究方向:智能建模、大数据、MBSE、复杂系统建模与仿真理论和方法,E-mail:yxf@nuaa.edu.cn。

丁叶(1995-),男,硕士研究生,研究方向:数据保护。

(编辑:张彤)