文章编号：2095-2163(2019)04-0234-03 中图分类号：TP311 文献标志码：A

一种 MD5 双重校验模型研究及应用

张明礼，袁佳峰
（上海浦东发展银行股份有限公司，上海 200001）

摘 要：本文提出了基于 MD5 算法的双重校验模型，实现了跨平台的版本一致性校验，可完成任意形式文件集合的一致性比对。介绍了该模型在投产版本交付、测试过程版本管理和测试案例集内容比对等领域的实际应用情况。

关键词：版本文件；MD5；一致性；散列

A dual MD5 checking model and its application

ZHANG Mingli, YUAN Jiafeng
(Shanghai Pudong Development Bank Co.,Ltd., Shanghai 200001, China)

【Abstract】 A dual checking model based on MD5 algorithm is proposed, which could realize cross-platform version consistency checking and complete consistency matching of arbitrary file sets. The application of the model in production version delivery, test version management and test case set content comparison are respectively introduced.

【Key words】 version file; MD5; consistency; hash

0 引 言

随着金融科技的快速发展，金融产品推陈出新日益加快，软件版本发布频率越来越高。如何加强开发、测试、运维团队之间的协作，有效管控版本文件在整个软件生命周期中的交付风险[1]，如何防范客观上的差错风险和主观上的篡改风险，是软件行业特别是金融行业需关注的问题。

MD5[2]是一种散列（Hash）技术，广泛用于加密、解密、数据签名和数据完整性校验等方面。本文主要关注数据完整性校验问题，对于任何一个文件，无论代码文件、可执行程序、或者其它类型的文件，不管文件字节数的多少，都可以计算出一个 MD5 值，可以通过对比同一文件的 MD5 值来判断文件是否被篡改。目前通行的做法是使用 MD5 算法解决单一文件的比对，本文基于标准 MD5 算法，提出了一种 MD5 双重校验模型，可以处理任意形式文件集合的一致性比对，在不改变交付物目录结构的情况下，有效防范投产版本交付、测试版本过程管控[3]中的操作风险。

1 MD5 校验模型介绍

1.1 MD5 双重校验模型原理

模型的技术原理为，使用标准 MD5 散列算法获得每个文件的 MD 值（第一次），遍历并记录所有文件的文件路径、文件名和 MD5 值，形成中间过程文件（文件不落地的情况下为字节流），针对该文件（字节流）生成 MD5 值（第二次），如图 1 所示。

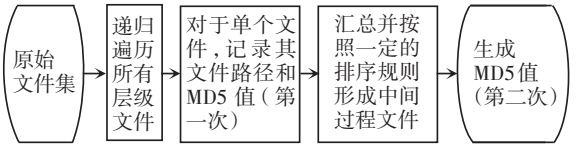

图 1 MD5 双重校验模型

Fig. 1 Dual MD5 checking model

1.2 模型设计目标

（1）对于任意形式（单一文件、复杂层级文件夹）的版本文件，最终生成唯一的 MD5 值。

（2）兼容主流操作系统，在不改变任何系统设置情况下，实现跨平台的一致性校验（UNIX 和 WINDOWS），即对于同一文件集合，在 WINDOWS 和 UNIX 平台下生成的最终 MD5 值应相同。

1.3 算法生成步骤

（1）递归方式遍历所有层级文件，以版本文件及其路径构建版本信息二叉排序树，并生成每个文件的 MD5 值。

（2）遍历二叉树，生成包含文件名称、文件路径、MD5 值信息的中间过程文件（字节流）。

（3）对中间过程文件（字节流）再次生成 MD5 值，该 MD5 值可记录至平台或通过邮件告知相关方，并作为一致性比对依据。

作者简介：张明礼（1978-），男，硕士，工程师，主要研究方向：自动化测试、软件配置管理等。

收稿日期：2019-04-11

2　实例验证

以个人网银系统日常变更的版本文件作为实例，说明该校验模型的应用情况。

（1）在个人网银系统 20190327 版本包中，包含一个 20190327 文件夹，该文件夹内有 menusNew.xml 等 4 个文件。

（2）中间过程文件，每一行对应一个文件叶子节点的 MD5 值，由文件路径，文件名组成，如图 2 所示。该文件可以根据需要写到本地备查，也可以不落地直接生成 MD5。针对此中间过程文件，2 次生成 MD5 值 32F1BE00B64BAC55D73EF02DB0E494C0，该 MD5 值可以通过平台或邮件，通知不同的操作人员，作为版本文件一致性的比对依据。

```
<ProductionVersion>
  <FileInfo FileMD5="9707A47E2BE191CE43C7FB9392C83581"
    FilePath="\20190327\menusNew.xml"
    FileName="menusNew.xml"/>
  <FileInfo FileMD5="C1B6F1419466973528A3ABDDD6F90A7A"
    FilePath="\20190327\nbper.ear" FileName="nbper.ear"/>
  <FileInfo FileMD5="7823D6D6EC2245838D39DCE38B397783"
    FilePath="\20190327\per_common.properties"
    FileName="per_common.properties"/>
  <FileInfo FileMD5="C84EB307E5BB5A91479716B45545ED8E"
    FilePath="\20190327\个人网银_发布说明（补充）.txt"
    FileName="个人网银_发布说明（补充）.txt"/>
</ProductionVersion>
```

图 2　中间过程文件

Fig. 2　Intermediate process file

（3）Windows 客户端工具。客户端工具支持 Windows 下的文件比对，文件路径指版本文件在本地的存放路径。平台（远程）MD5 值是基于中间文件生成的最终 MD5 值。本地 MD5 值是根据本地文件生成的 MD5 值。在 D:\MD5 目录下，存放了网银 20190327版本包，Windows 客户端工具如图 3 所示。

图 3　Windows 客户端工具

Fig. 3　Client tool for Windows

（4）UNIX 平台客户端比对工具。UNIX 类平台虽然具有多样性，但一般都预装有 perl 解释器，并且无需安装运行环境，普通用户可以直接执行。经过调研比较，本研究选择 perl 语言开发比对工具，兼容 AIX、HP-UX、LINUX 等 UNIX 类工作平台，要求 perl 环境为 5.6 及以上版本。工具支持文件和目录的校验，支持相对路径和绝对路径，通过如下格式调用 perl 脚本进行 MD5 一致性校验：

perl md5check.pl - d directory（filename）-m md5Value。

Perl 环境版本号检查方法如下：

[host1013_cbs]/cbs/md5 >perl -v

This is perl，v5.10.1（*）built for aix-thread-multi

在/cbs/md5/nb 目录下，存放了网银 20190327 版本包，下面给出了 3 种情况的比对示例：

①输入正确的 MD5 值，提示 MD5 比对通过：

[host1013_cbs]/cbs/md5 >perl md5check.pl-d nb -m 32F1BE00B64BAC55D73EF02DB0E494C0

MD5 Verified OK

②输入错误的 MD5 值，提示 MD5 比对不通过：

[host1013_cbs]/cbs/md5 >perl md5check.pl-d nb -m 32F1BE00B64BAC55D73EF02DB0E494C1

MD5 Verified FAILURE，please recheck the md5 value you input!!

③输入正确的 MD5 值，但修改了版本文件 menusNew.xml（例如在文件最后增加一个字符），提示 MD5 比对不通过：

[host1013_cbs]/cbs/md5 >perl md5check.pl-d nb -m 32F1BE00B64BAC55D73EF02DB0E494C0

MD5 Verified FAILURE，please recheck the md5 value you input!!

（5）性能表现。对于 100 M 以下的版本文件，在 1 s 内即可获得文件集的 MD5 结果；800 M 以上版本文件，其处理时间在 10 s 内，属于可接受的等待时间。但测试发现，较大文件的 CPU 消耗略高，建议不要在生产环境的业务高峰期使用该工具。

为了得到 MD5 客户端工具的使用效率和性能开销情况，按照版本文件大小区分，分别选取大中小 3 种类型的版本文件，进行执行时间、系统资源消耗的监控，测试结果见表 1。

MD5 客户端工具的运行时间和版本大小成正比，CPU 资源的消耗主要和版本文件的大小、CPU 主频与数量、存储设备性能等有关。

表 1 耗时及 CPU 占用情况对比

Tab. 1 Time & CPU cost

平台版本与配置	版本文件 1(文件集大小为 7 M)		版本文件 2(文件集大小为 116 M)		版本文件 3(文件集大小为 875M)	
	耗时/s	CPU/%	耗时/s	CPU/%	耗时/s	CPU/%
AIX7.1 , 4C64G	<1 s,极短	1.0	1.0	2.7	3.0	16.1
HP11.31,4C16G	<1 s,极短	2.3	1.0	6.3	5.0	12.1
Windows 7,2C8G	<1 s,极短	2.0	1.0	8	9.0	15.0

3 应用场景

该模型在投产版本交付、测试过程版本管理和测试案例集内容比对等领域得到了实际应用,具体应用情况如下:

3.1 投产版本交付管理领域

投产版本是指最终发布在生产环境中的一组特定软件包及其各类附属文件的集合,一般包括目标代码、脚本、配置文件和安装说明文档等。投产版本交付的主要环节为:版本文件在项目开发方制作完成后,经由测试方测试通过,最终交付给运维方在生产环境变更实施。对于任何一个版本文件集,开发方在制作完成后提交至版本平台,并检验本地和平台 MD5 值是否保持一致;测试方在 MD5 一致性校验通过后开展测试验证工作;实施方在 MD5 一致性校验通过后方可进行生产环境发布。可以看到,模型的应用贯穿了投产版本从开发、测试到投产的全过程。通过提供跨平台 MD5 一致性校验,满足了应用系统平台多样性需求,实现了浦发银行应用系统投产版本交付的全覆盖,月均交付投产版本超过 300 个。该模型的成功应用,替代了传统的基于文件名、字节数和时间戳等内容为主的手工比对,实现了针对版本内容的一致性校验。不仅有效防范了客观上的差错风险(例如网络传输过程中的部分文件意外丢失)和主观上的篡改风险(例如上传未经授权的恶意代码),同时提高了版本比对效率,为应用系统的安全投产与稳定运行提供了重要保障。

3.2 测试版本过程管控领域

生产环境的变更往往有严格的变更管理流程来管控,测试环境一般由项目组自行管理,应用版本更新相对来说具有灵活性和随意性。假定在某一轮测试过程中发生了版本更新,就会引发测试结果与测试版本之间无法有效对应,进而影响测试结果的可信度。使用了基于此校验模型的版本变更检测工具,可随时监控被测环境的应用版本更新情况。当测试版本变更时,项目开发人员和测试人员都会收到版本变化的邮件通知,及时掌握每次版本更新的情况,掌控协调版本更新节奏,减少了对功能、性能差异的探究以及频繁更新导致的重复测试。测试版本从首次提交到测试结果发布,整个测试过程纳入管控,有力推进了测试版本交付和变更过程规范化。

3.3 测试案例自动化生成项目

基于业务流程图生成的测试案例集中,测试案例数量往往达到数百个。如果业务流程图因业务变化需要更新,更新前后生成的有差别的案例可能只有几个,通过人工识别这些差别,效率非常低。通过基于该模型的中间过程文件比对,可以快速的定位哪些案例发生了变化,在系统层面予以标记,节省了大量的人工成本。

4 结束语

本文提出了一种改进的 MD5 校验模型,该模型成功应用于投产版本文件交付、测试过程版本管控、测试案例集内容比对等领域。模型的提出及其应用,提升了投产版本交付的效率和安全性,实现了测试版本的过程管控,大幅提升了测试案例集比对的效率。基于该模型的研究和应用,对于软件行业和金融行业具有一定的参考价值。

参考文献

[1] 董昕,郭勇,王杰. 基于 DevOps 能力模型的持续集成方法[J]. 计算机工程与设计,2018,39(7):1930-1937.

[2] 崔永辉,贾连兴,张江. MD5 算法研究[C]//第 17 届中国系统仿真技术及其应用学术年会论文集(17th CCSSTA 2016).安徽:中国自动化学会系统仿真专业委员会,2016,17:216-218.

[3] 冯文亮,陈俊,成洁. 测试版本管理平台设计与应用[J]. 中国金融电脑, 2018(5):57-62.

(上接第 233 页)

[8] MUJUMDAR A, TAMHANE B, KURODS S R. Observer-based sliding mode control for a class of noncommensurate fractional-order systems[J]. IEEE/ASME Transactions on Mechatronics, 2015, 20(5):2504-2512.

[9] ATEN Q T, ZIRBEL S A, JENSEN B D, et al. A numerical method for position analysis of compliant mechanisms with more degrees of freedom than inputs[J]. Journal of Mechanical Design, 2010, 133(6):491-502.

[10] 杨延勇. 汽车车身结构件设计与性能计算分析[J]. 机械工程与自动化,2018(2):117-119.