

**毕业设计（论文）**

题 目 企业服务器文件完整性检查

系统的设计与实现

专 业 网络空间安全

学　　 号 160700226

学 生 陈锦前

指 导 教 师 董开坤

答 辩 日 期 2020.06.09

摘 要

计算机网络技术的不断发展，使得人们的工作和生活发生了巨大的变革。网络已经渗透到生活的各个领域，随之而来的是个人隐私的泄露、重要文件的篡改等安全问题。文件完整性检查系统作为入侵检测系统必不可少的一个子模块，可以保障系统运行安全。

本论文研究了基于MD5算法与DES算法的文件完整性检查技术，设计并实现了针对企业服务器的文件完整性检查系统，详细阐述了系统构建过程，并通过实验的方式对系统进行评估。

该系统的文件数据均来自于Windows环境下的文件。该系统首先使用操作系统的接口获取文件的属性值以及文件内容，再使用MD5算法生成文件内容的摘要值，最终将文件的属性值及文件内容摘要值存储到数据库从而得到文件的初始消息。该系统分为检查端与服务器端，使用基于TCP协议的通信技术，对企业服务器文件完整性进行检查，并将检查结果可视化输出。

通过对检查结果的分析，该系统可以实现对多台企业服务器中的不同文件进行完整性检查，并将检查出被篡改的文件进行记录及展示。系统在文件完整性检查上整体较为有效。

**关键词**：入侵检测；文件完整性检查；摘要值计算；MD5算法；DES算法；

**（内容及关键词用小4号字）**

**Abstract**

With the continuous development of computer network technology, great changes have taken place in people's work and life. The network has penetrated into all areas of life, followed by personal privacy disclosure, important documents tampering and other security issues. As an essential sub module of IDS, file integrity detection system can ensure the security of the system.

This paper studies the file integrity detection technology based on MD5 algorithm and DES algorithm, designs and implements the file integrity detection system for enterprise server, describes the system construction process in detail, and evaluates the system through experiments.

The file data of this system are all from files under Windows environment. The system first uses the interface of the operating system to obtain the attribute value and the content of the file, then uses the MD5 algorithm to generate the summary value of the file content, finally stores the attribute value and the summary value of the file content to the database to get the initial message of the file.The system is divided into detection end and server end.The system is divided into detection end and server end. It uses the communication technology based on TCP protocol to detect the integrity of enterprise server files, and visualizes the detection results.

Through the analysis of the detection results, the system can detect the integrity of different files in multiple enterprise servers, and record and display the tampered files. The system is more effective in document integrity detection.

***空一行***

**Keywords:** intrusion detection，integrity detection，summary value calculation，MD5 algorithm，DES algorithm

**（内容及关键词用Times New Roman**

**注意事项***中各章题序及标题用黑体小4号字，其余用宋体小4号字，数字及符号用新罗马字体*

**目 录**

[摘 要 I](#_Toc42259475)

[Abstract II](#_Toc42259476)

[第1章 绪 论 1](#_Toc42259477)

[1.1 课题背景及研究的目的和意义 1](#_Toc42259478)

[1.1.1 课题背景 1](#_Toc42259479)

[1.1.2 研究目的和意义 1](#_Toc42259480)

[1.2 国内外在该方向的研究现状及分析 2](#_Toc42259481)

[1.2.1 国外的研究现状 2](#_Toc42259482)

[1.2.2 国内的研究现状 3](#_Toc42259483)

[1.3 论文主要内容与结构安排 4](#_Toc42259484)

[第2章 系统需求分析 6](#_Toc42259485)

[2.1 系统概述 6](#_Toc42259486)

[2.1.1文件完整性检查技术 6](#_Toc42259487)

[2.1.2 基于TCP协议的通信技术 9](#_Toc42259488)

[2.2 系统功能性需求分析 12](#_Toc42259489)

[2.3 系统可行性分析 14](#_Toc42259490)

[2.4 本章小结 15](#_Toc42259491)

[第3章 企业服务器文件完整性检查系统设计 16](#_Toc42259492)

[3.1 系统整体设计 16](#_Toc42259493)

[3.1.1 系统架构设计 16](#_Toc42259494)

[3.1.2 系统功能设计 17](#_Toc42259495)

[3.2 功能模块概要设计 18](#_Toc42259496)

[3.2.1 文件信息获取和存储模块设计 18](#_Toc42259497)

[3.2.2 文件完整性检查模块设计 20](#_Toc42259498)

[3.2.3 检查结果可视化模块设计 21](#_Toc42259499)

[3.3 系统数据架构设计 21](#_Toc42259500)

[3.4 服务器端设计 21](#_Toc42259501)

[3.4.1 文件信息对比与检测结果发送 22](#_Toc42259502)

[3.5 检查端设计 23](#_Toc42259503)

[3.5.1 检查请求发送 23](#_Toc42259504)

[3.5.2 检测结果接收与发出警告 25](#_Toc42259505)

[3.6 本章小结 25](#_Toc42259507)

[第4章 企业服务器文件完整性检查系统实现 26](#_Toc42259508)

[4.1 文件信息获取和存储模块实现 26](#_Toc42259509)

[4.1.1 文件属性与内容获取 26](#_Toc42259510)

[4.1.2 文件信息获取与存储 26](#_Toc42259511)

[4.2 文件完整性检查模块实现 28](#_Toc42259512)

[4.2.1 文件信息对比 28](#_Toc42259513)

[4.2.2 检查信息交互 30](#_Toc42259514)

[4.3 检查结果可视化模块实现 33](#_Toc42259515)

[4.3.1 检查结果展示 33](#_Toc42259516)

[4.3.2 发出弹窗警告 34](#_Toc42259517)

[4.4 本章小结 35](#_Toc42259518)

[第5章 系统测试与结果分析 36](#_Toc42259519)

[5.1 系统功能测试 36](#_Toc42259520)

[5.1.1 测试说明 36](#_Toc42259521)

[5.1.2 测试用例 36](#_Toc42259522)

[5.2 系统性能测试 - 38](#_Toc42259523)

[5.3 测试结果与分析](#_Toc42259524) 39

[5.4 本章小结](#_Toc42259525) 39

[结 论 40](#_Toc42259526)

[参考文献 41](#_Toc42259527)

[致 谢 43](#_Toc42259528)

第1章 绪论

## 1.1 课题背景及研究的目的和意义

### 1.1.1 课题背景

随着现代网络技术的快速发展和网络规模的不断扩大，计算机网络已渗透到了社会的各个领域，人们在享受网络带来的资源共享及信息交流方便快捷的同时，也不得不面对越来越多的来自网上的恶意攻击和各种黑客入侵等，也存在个人隐私泄露、重要文件篡改等安全问题，而且日新月异，人们对于网络安全的需求与日俱增[1]。

互联网与生俱有的开放性、交互性和分散性特征，使人类所憧憬的信息共享、开放、灵活和快速等需求得到满足。网络环境为信息共享、信息交流、信息服务创造了理想空间，为人类社会的进步提供了巨大推动力。然而也正是由于互联网的上述特性，产生了许多安全问题[2]。尤其当今网上支付、数字货币技术不断成熟，保护企业的文件至关重要。

攻击者越来越多,攻击工具和手法日趋复杂多样,单防火墙技术已经无法完全阻止网络入侵这一日益严重的网络安全问题。所以需要在防火墙后提供新的安全保障，网络的防卫必须采用一种纵深的、多样的手段[3]。入侵检侧系统(IDS)是一种不同于防火瑞的主动保护网络资派的网络安全系统,是防范网络攻击的最后一道防城,是作为防火堵有益补充而出现的网络第二道防线产品,在网络安全技术中起着不可替代的作用[4]。

### 1.1.2 研究目的和意义

按照美国国防部1983年制定的可信计算机系统评测标准，信息安全应当确保信息的机密性(Confidentiality)、完整性(Integrity)和可用性(Availability)这三个基本标准要求。其中,完整性指防止信息遭受以非授权方式所作的篡改或破坏,它作为信息安全三个基本标准要求之一,占有举足轻重的作用[5]。

系统的安全是一个体系,建立在各种安全机制集成的基础上,入侵检测系统(IntrusionDeteetionSystem,简称IDS)作为系统安全的一道重要防线,主要是通过多种手段监控系统来发现入侵。文件完整性检查是IDS中的关键部分之一,它通过检查系统关键资源的变化情况来为入侵的发现提供依据。文件完整性检查系统的目的在于:保证被保护主机的关键资源不被恶意更改。所谓关键资源是指系统中不应也不能被经常更改的关键成分,如:内核、配置文件、可执行程序、库等,以及那些由第三方提供的可能影响到主机运行的资源[6]。

文件完整性检查系统的优点：①从数学上分析，攻克文件完整性检查系统，无论是时间上还是空间上都是不可能的。文件完整性检查系统是非常强劲的检测文件被修改的工具。②文件完整性检查系统具有相当的灵活性，可以配置成为监测系统中所有文件或某些重要文件。③当一个入侵者攻击系统时，他会干两件事：首先，他要掩盖他的踪迹，即他要通过更改系统中的可执行文件、库文件或日志文件来隐藏他的活动；其次，他要作一些改动保证下次能够继续入侵。这两种活动都能够被文件完整性检查系统检测出[7]。

通过研究企业服务器文件完整性检查系统的设计与实现，可以有效地检测攻击，可以更有效地保护系统关键文件不被篡改，防止木马、病毒等对系统的破坏，保障企业正常运行[8]。

## 1.2 国内外在该方向的研究现状及分析

### 1.2.1 国外的研究现状

2014年Mark Kedgley通过分析当前黑客的攻击行为以及网络环境，指出了当前杀毒软件的不足之处，例如无法有效抵御基于0-day漏洞的攻击行为，同时阐明了文件完整性检查系统的优点及可靠性[9]。

2014年,AlienVault的Patrick Bedwell研究了SIEM（安全信息和事件管理）和日志管理的技术基础，强调了传统传统的SIEM的复杂性会使许多中小型企业遭受到黑客攻击，因此入侵检测系统、文件完整性检查系统也是必须的[10]。

2014年，日本学者Watanabe,H根据在日本使用广泛的Gfarm分布式文件管理系统，在已有的基于时间戳的文件完整性检查系统基础上，设计了一套改进系统，大幅度提升了该系统的效率[11]。

当前，许多安全研究集中在可生存或者入侵容忍系统上，这些系统不是被认为不可攻击的，而是能够抵御攻击的影响，并在持续攻击的情况下继续提供服务。Vinı́cius da Silveira Serafim,Raul Fernando Weber提出了一套文件完整性控制系统，该系统包括文件备份模块，文件完整性检测模块。当系统检查到文件受到篡改后，能及时从文件备份数据库中将文件进行恢复，以保持系统的完整性、可用性和机密性[12]。

有许多私人或机密文件存储在计算机中或在Internet上传输。人们担心甚至害怕自己的安全问题，如盗窃、破坏、伪造等，迫切需要一个既安全又高效（为了获得更好的体验）的文件保护系统（SEEPS）。为此，Xiongwei Fei,Kenli Li,Wangdong Yang,Keqin Li提出并实现了一个安全高效的文件保护系统（SEFPS）。基于高级SHA3（Secure Hash Algorithm 3）和并行AES（Advance Encryption Standard）的SEFPS可以提供保密性和完整性的保护，并通过GPU（Graphics Processing Unit）并行或/和CPU（Central Processing Unit）并行产生高性能[13]。

云存储系统中数据的所有权和管理权分离,导致用户数据隐私泄露、完整性破坏等一系列安全问题。2018年Pinheiro Alexandre等人设计了一套基于实时监控云端数据及服务器行为的系统用于保护用户隐私[14]。

### 1.2.2 国内的研究现状

近年来中国学者在文件完整性检查研究方向取得了不断进步，尤其在云计算以及云存储方向，根据云计算环境下平台系统的特点,刘海永提出了一套新型的文件完整性检测系统的设计方案，采用了新型的SHA-3安全算法生成文件摘要，将安全系统与被检测主机相互分离，将安全检测负载与被检测主机适当分离。实现了检测系统的运行的大部分负载从被监控主机转移,从而适用于云计算环境[15]。

随着云端存储技术的不断成熟，目前已有的数据完整性证明机制还存在不支持全动态操作,通信和计算开销大等各种问题。因此贾倍提出了支持数据隐私的数据完整性证明机制、一个带秩的多分支路径结构(MBT,Multi-branch Tree)和数据完整性检测算法、以及基于MBT的数据完整性证明机制，该方案不仅保护了数据对第三方验证者的隐私性,而且支持全动态操作与批量审计操作,简化了数据完整性证明信息的获取过程[16]。

同时云服务商可以向数据用户提供数据存储和计算等服务。但是,在这种服务中必须有一个代表数据用户利益的第三方审计员定期验证用户外包数据的完整性。而且,在外包和审计过程中必须保护数据用户的身份和数据文件内容的安全。姜红,亢保元,李春青针对云共享数据完整性检测问题,利用数字签名技术和双线性对的性质,对两个保护用户身份的云共享数据完整性公开审计方案进行了改进[17]。

刘广沛利用移动Agent技术在云端部署分布式虚拟机代理模型,同时通过虚拟机代理模型构建基于区块链的完整性保护框架,使用默克尔哈希树生成文件对应的唯一散列哈希值,借助区块链上的智能合约监控数据变化,及时向数据拥有者发出数据篡改的警告信息，另外,利用“挑战-应答”模式构造基于区块链的云数据完整性验证方案[18]。

移动可信平台需要经常进行大量完整性校验,然而其各种资源却十分有限。张永棠教授提出了一种可以配合各种成熟的HASH算法使用的完整性校验方法——RMAC(Random-MAC)。从不同版本的Linux系统中,搜集了不同格式的ELF文件,随机抽取粒度样本,进行完整性校验分析。由于其引入随机性,每次计算产生的摘要都不同,使目前已有的病毒都无法做到每次都能通过RMAC校验。RMAC提供的障碍有效地降低了病毒的繁殖速度,可以阻止病毒的大规模爆发[19]。

针对现有支持隐私保护的批量检测方案均未考虑重要数据安全性的问题,邵必林,吴书强,刘江,胡家发提出了一种重要数据完整性分布式检测系统,该系统通过AES-128加密算法对数据进行加密,保证了数据的安全性。在重要数据完整性检测过程中利用双线性映射性质,采用将多个重要数据证据和标签证据加密后聚合,保护用户隐私。基于多个分布式检测代理节点,采用加权最小连接数调度算法动态调整分布式检测代理节点的检测任务,使得分布式检测代理节点的性能与待检测任务数量高度匹配,避免检测堵塞的情况,减少检测任务整体响应时间,增强检测系统的高可用性[20]。

## 1.3 论文主要内容与结构安排

本次课题的主要研究内容是设计一个企业服务器文件完整性检查系统，对企业服务器中的文件信息进行提取、分析、检查，以实现在复杂的网络环境下，根据文件属性或者文件摘要进行文件完整性检查，判断文件是否受到恶意篡改。具体研究内容如下：

（1）对现有的文件完整性检查技术进行调研并完成综述

（2）设计针对企业服务器的文件完整性检查系统，使用改进后的MD5算法进行文件初始信息获取，并基于MySQL数据库进行信息存储。

（3）基于TCP协议，使用多进程/多线程编程技术，实现通信系统的构建，完成检查请求发送与检查结果接收的并发处理。

（4）利用Python编程实现所设计的企业服务器文件完整性检查系统。

（5）使用该系统对多台服务器的不同文件进行完整性检查，通过对检查结果进行分析得出结论。

本文结构及各章节所述信息具体如下：

第1章是绪论，本章主要介绍了本课题研究的背景、课题研究需要达到的目的，以及国内外从大数据、云存储、可信计算等不同方向对文件完整性检测系统的研究现状、详细说明了本文要研究的具体内容。

第2章是相关理论概述以及系统需求分析，本章对企业服务器文件完整性检查技术的相关的理论进行介绍，主要由基于TCP协议的通信技术和文件完整性检查技术这两方面构成，其中文件完整性检查技术作为本章的重点内容，详细介绍了基于MD5算法的文件完整性检查技术以及MD5算法的优化技术。对系统进行了功能性需求分析以及可行化分析。

第3章是企业服务器文件完整性检查系统设计，本章对此系统进行了总体设计和功能化模块概要设计。本章共设计了三个功能模块，并对各个模块的具体功能进行了阐述。

第4章是企业服务器文件完整性检查系统的系统实现，本章描述上一章所设计内容的实现过程，包括文件初始信息获取与存储模块、文件完整性检查模块、检查结果可视化模块。

第5章是系统测试与分析，本章设计了对企业服务器文件完整性检查系统的功能测试与性能测试，根据测试结果进行分析得出结论。

结论总结了本次课题的全部研究工作、各项工作的完成状态，对接下来的研究工作进行了展望。

第2章 系统需求分析

本章首先对论文研究的课题进行概述，主要对课题中的通信技术与文件完整性检查技术进行解释说明。根据系统概述中所描述的内容对系统的功能性需求和可行性进行分析，最后总结本章内容。

## 2.1 系统概述

本论文研究了文件完整性检查技术和基于TCP协议的通信技术，根据研究内容设计和实现了企业服务器文件完整性检查系统。那么首先对该系统所使用的相关技术进行概述，之后再分析得出系统需求。

### 2.1.1文件完整性检查技术

完整性指网络信息的真实可信性，即网络上的信息不会被偶然或者蓄意地进行删除、修改、伪造、插入等破坏，保证授权用户得到的信息是真实的。只有具有修改权限的实体才能修改信息，如果信息被未经授权的实体修改了或者在传输过程中出现了错误，信息的使用者应该能够通过一定的方式判断出信息是否真实可靠[21]。目前主要使用密码学中的各种算法对文件完整性进行检查，本节主要介绍两种常见的文件完整性检查技术。

一、基于MD5算法的文件完整性检测技术

美国密码学学家罗纳德（Ronald Linn Rivest）在MD5算法的基础上，于1992年设计出更为趋于成熟的MD5算法。使用该算法可以将任意长度的文本转换为一个固定长度的字符串。由于同一文件产生出相同字符串的概率极小，可将该字符串视为该文件的摘要信息。基于MD5算法的文件的完整性检查是比较文件前后的摘要值是否相同，如果被篡改，摘要值将发生变化，因此可以有效检查出文件是否完整，其检查模型图见图2-1所示。

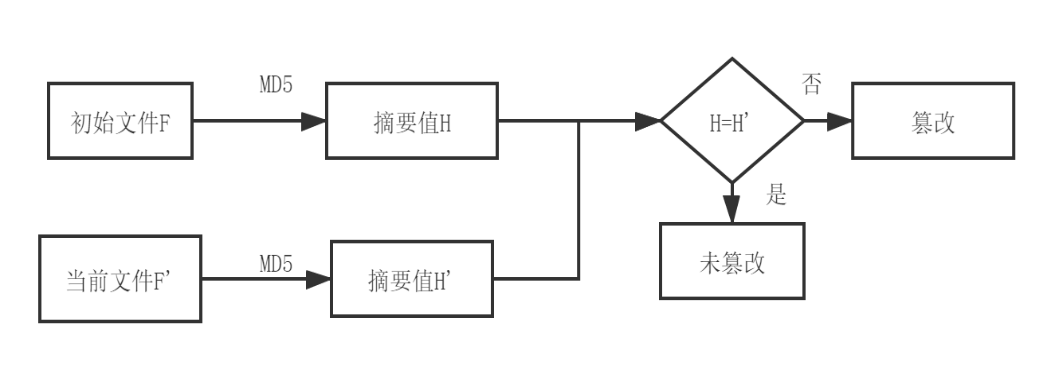


图2-1 基于MD5算法的文件完整性检查模型图

自2004年8月，王小云在美国加州圣芭芭拉召开的国际密码大会上首次宣布了她及她的研究小组破译了MD5算法后用MD5做数字签证、电子签名已经不安全了，但是在普通的应用开发中MD5还是被开发者广为应用，虽然随着海量存储技术和分布式计算的发展，在如今的大数据时代，可以通过创建名文库组建反向查询库对MD5值进行反向查询，只要拥有足够大的反向查询库，就有可能直接查询出器明文值。但是通过用户和系统对MD5算法作部分修改或辅助改进则能有效加大反向查询的难度，从而避免攻击者从数据库中的文件摘要值直接获取文件内容导致文件泄露。例如以下几种方案：

（1）多次加密

进行一次MD5加密后，可对生成的MD5密码的全部或部分再进行一次或多次加密，增加在字典中查找的难度。如md5(md5(“cjqhit”))是对MD5码的全部进行再一次加密；md5（right（md5（“cjqhit”），10））是对MD5码的右10位进行再加密。经实验证明多次MD5加密是不会影响其时间性能的。

（2）加盐法

若想解决短明文摘要加密安全性不高的问题，可以增加原始明文长度, 该方法又称为加盐法。这个“盐”常为字符串数据，将盐与密码一起合并计算摘要，可增加查询破解的难度，从而提升安全性。破解网站的暴力查询破解只能查询常见的短明文的摘要，如果加长原文数据长度，摘要结果则很难查到[22]。

（3）对MD5算法的步函数和消息分量计算方法进行改进

由于HASH函数本身的限制，使用HASH函数加密会出现碰撞的现象，其抗碰撞性不足的重要原因之一是高位不能充分混淆，为了使得MD5抗碰撞的能力得以提升，我们可以对明文进行消息预处理，根据多次实验探索，发现对Mi的复杂化是十分有效的改进方式，改进方法为把原先只有一个参数的32位Mi改成有三个参数取三个提取出来的32位明文消息段相乘取后32位的合成参数[23]。

（4）散列值的长度增加

虽然128位散列值空间搜索的复杂度已经很大了，但是随着各类算法的不断优化，寻找碰撞变得越来越简单。增加散列值的长度虽然会对运算速度造成影响，但对于增加安全性考虑，这些改进是很有意义的[24]。

该系统存在大量、同时的使用MD5算法对文件内容进行摘要值的计算，MD5算法过程繁琐、算法复杂，在具体实现时必须考虑其计算性能，如果加密时间过长将会为该系统带来非常差的用户体验。所以对于MD5算法的计算性能的优化分析具有非常大的现实意义[25]。这里介绍4种优化MD5算法计算性能的方法。

（1）展开MD5算法的循环过程

由于MD5算法需要很多循环，而且很多时候采用多层循环嵌套来实现。对于计算机体系来说, 多次的循环与多层次的循环嵌套，加上变量的地址寻址，自身在CPU执行时，会浪费大量的时间，加上多次循环和多层循环嵌套，与变量地址寻址使得CPU指令流水线的预取与阻断的机制失效，从而增加了大量的计算时间。对于大数量级和多层循环嵌套的MD5算法来说，可以通过展开循环过程来提升其计算时间。所以在编程实现MD5算法时，尽量将每个循环体采用5到10次的循环次数，通过多个循环程序块来完成整个MD5算法。同时，为了减少内存寻址的时间浪费，能够采用常量的变量，尽量在预定义时采用常量的形式定义。

（2）避免指令跳转

在当前CPU的计算机制中，执行固定内存块的指令速度是最快的，如果发生程序地址指针改变，则CPU将会浪费一部分时间去内存寻址或从虚拟交换空间或磁盘中读取数据，而这将会浪费大量的计算时间。对于常见的编程语言, if…else或者for、while循环，甚至是goto语句，都是常见的指令跳转语句，在实现MD5算法时，尽量避免使用产生指令跳转的语句，从而减少CPU执行代码时的寻址时间。

（3）变量长度CPU寄存器匹配

不同计算机的CPU寄存器大小是不一样的。对于长度大于CPU寄存器长度的变量，CPU将会分多次进行计算，最后将结果进行整合来完成计算。如果变量长度小于CPU寄存器长度，那么CPU将会在寄存器后附加其他的数据或者执行来完成计算。对于一个32位寄存器的CPU来说，对一个32位变量的计算时间将比对一个16位变量的计算时间块近一倍，所以在复杂繁琐的MD5算法实现过程中, 定义变量或常量时，尽量与当前执行算法的CPU寄存器长度一致，从而增加整个代码中数值的计算速度。

（4）减少变量个数

由于操作系统对于变量的存放机制，使得CPU在对某个变量进行计算时，需要按照变量指针从内存中寻址读取，存放到寄存器中进行计算。整个计算机框架CPU从内存读取和写入的时间对于CPU计算来说慢很多，所以在MD5算法实现时, 尽量减少变量的个数。此外，常量与变量的存放机制也有所不同，所以对于固定不变的数据尽量使用常量类型代替。

二、基于DES算法的文件完整性检查技术

DES是一种典型的对称加密算法，将CBC模式的反馈机制与传统的加密算法DES结合起来，产生文件相应的密文校验码，这样不仅可以验证了数据的有效性，也可以防止信息被恶意篡改、伪造或假冒。

DES也是一种分组密码加密算法，它以64位分组对数据加密。它的密钥长度是56位（因为每个第8位都用作奇偶校验），密码可以是任意的56位的数，而且可以任意时候改变。其中有极少量的数被认为是弱密钥，但是很容易避开它们，所以保密性依赖于密钥。DES算法的加密过程大致可分成四步:初始置换、迭代过程、子密钥生成和逆置换。该加密算法就是将两种基本的加密方法—替换和移位精心且完美的结合起来。通过有意识的将一些位置替换为其他位的方法使得原明文结构混乱，再通过重新安排达到扩散的目的，安全强度也通过重复应用这种技术来保证。分组密码工作模式是利用分组密码加密消息的技术，当消息的长度大于分组的长度时，首先把消息按分组的长度进行分块，然后利用分组密码进行加密。CBC是一种密码分组链接模式，即是将一种反馈机制引入分组密码中，前一个分组的加密结果被反馈机制引入到当前分组的加密中，换而言之，每一个分组被用于修改下一个分组的加密。每一个密文不仅依赖于产生它的明文分组，而且依赖于所有前面的明文分组。改变任一明文组中一个字节的内容，都会影响明文的产生。

根据以上理论。首先将8字节的文件校验码附加于文件末尾，当加密后，会产生相对应的密文校验码，然后将其与原始密文校验码进行比较，就可以判断文件是否遭受篡改，保证文件的完整性。其检查模型图见图2-2所示。

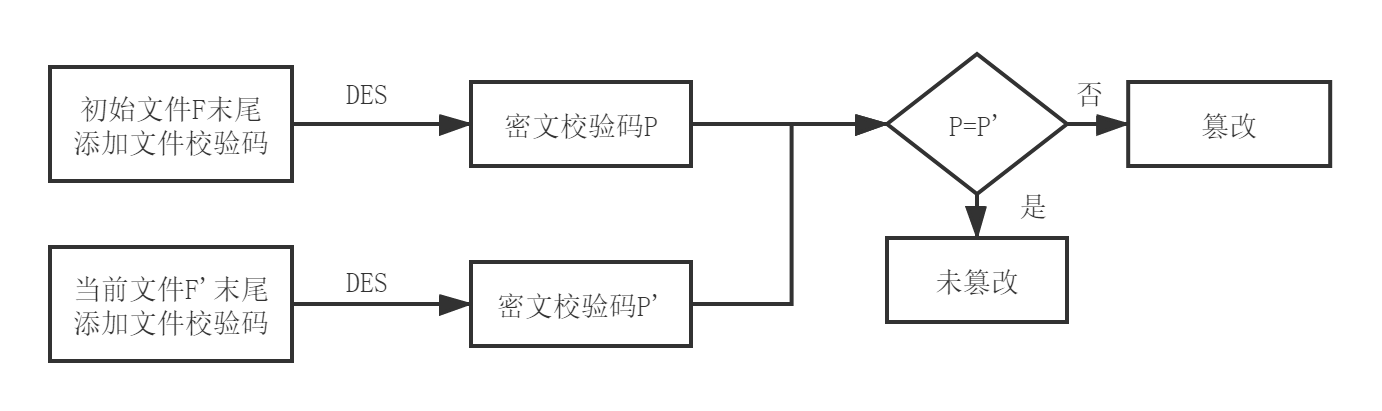


图2-2 基于DES算法的文件完整性检查模型图

因为基于DES的文件完整性检查技术相对繁琐，且消耗系统资源较大，只适用于少量、小型的文件进行文件完整性检查，所以本次课题使用基于MD5算法的文件完整性检查技术。

### 2.1.2 基于TCP协议的通信技术

随着互联网的普及和信息技术的发展，现代社会已经变成一个由网路服务器和主机节点组成的地球村。网络设备之间通信常用采用的常用模型是ISO（国际标准化组织）定义的七层网络互联模型。目前主流的技术是基于TCP协议的Socket通信，在该系统中企业服务器要接收大量的文件完整性检查请求，需要对服务器的并发处理能力进行良好的设计。本节将详细介绍利用Socket和并发编程技术实现检测端与服务器端的高效可靠通信。

TCP是一种面向连接的、可靠的、基于IP的全双工传输层协议，当应用层向TCP层发送8位字节表示的数据流时，TCP将数据流分割成适当长度的报文段，然后TCP把数据包传给IP层，由它来通过网络将包传送给接收端实体的IP层，再由对端IP层传输TCP层。

基于TCP的Socket通信结构是一个5元组：（协议、源IP地址、源端口、目的IP地址、目的端口）。Socket通信本质上是端到端通信（从源端到目的端），人们通过一个Socket连接就唯一确定了通信的双方及通信的协议，所以一旦Socket连接后，通信两端就可以通过这个Socket连接进行通信。TCP通信典型时序图见图2-3所示。

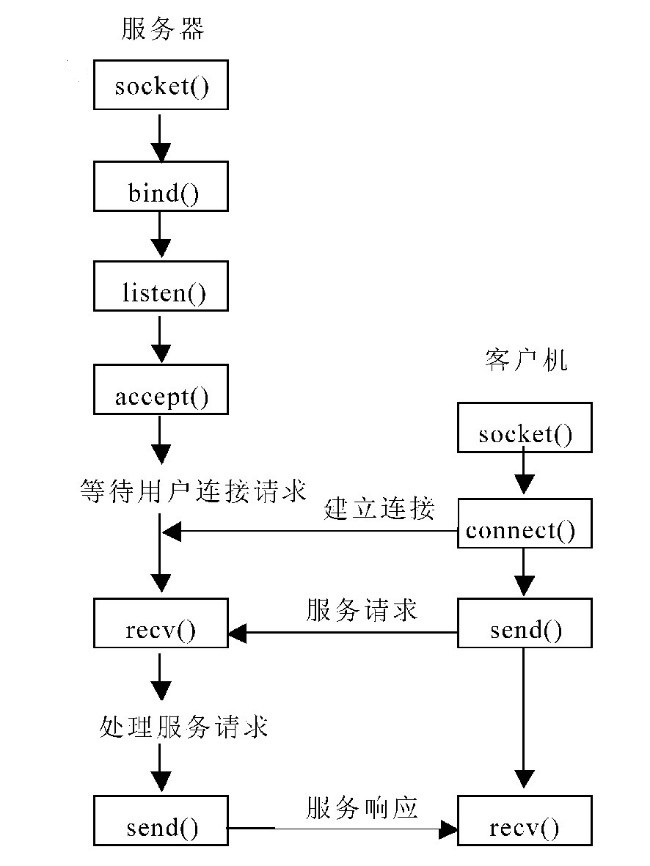


图2-2 TCP通信典型时序图

该系统中建立高性能并发的通信系统非常有必要，在服务器方面常见的编程模型包括：迭代模型、并发模型、异步模型。下面分别介绍这几个模型。

（1）迭代服务器

迭代服务器模型有唯一的侦听进程，它accept新的Socket连接之后，通过fork新的子进程的方式为这个请求进行专门的服务，直至请求处理完成，直至请求处理完成，之后才会释放在处理该请求过程中被占用的各种资源（包括进程、内存、Socket句柄等）。采用迭代型服务器模型编程最常见的代码是Apache服务器。这种模型的优点是开发简单，缺点是并发处理能力有限（受限服务器的资源），而且服务完成后的资源释放也是一个容易出错的地方，处理不当容易造成资源泄漏， 从而最终导致服务器无法响应新的服务。其流程图见图2-3。

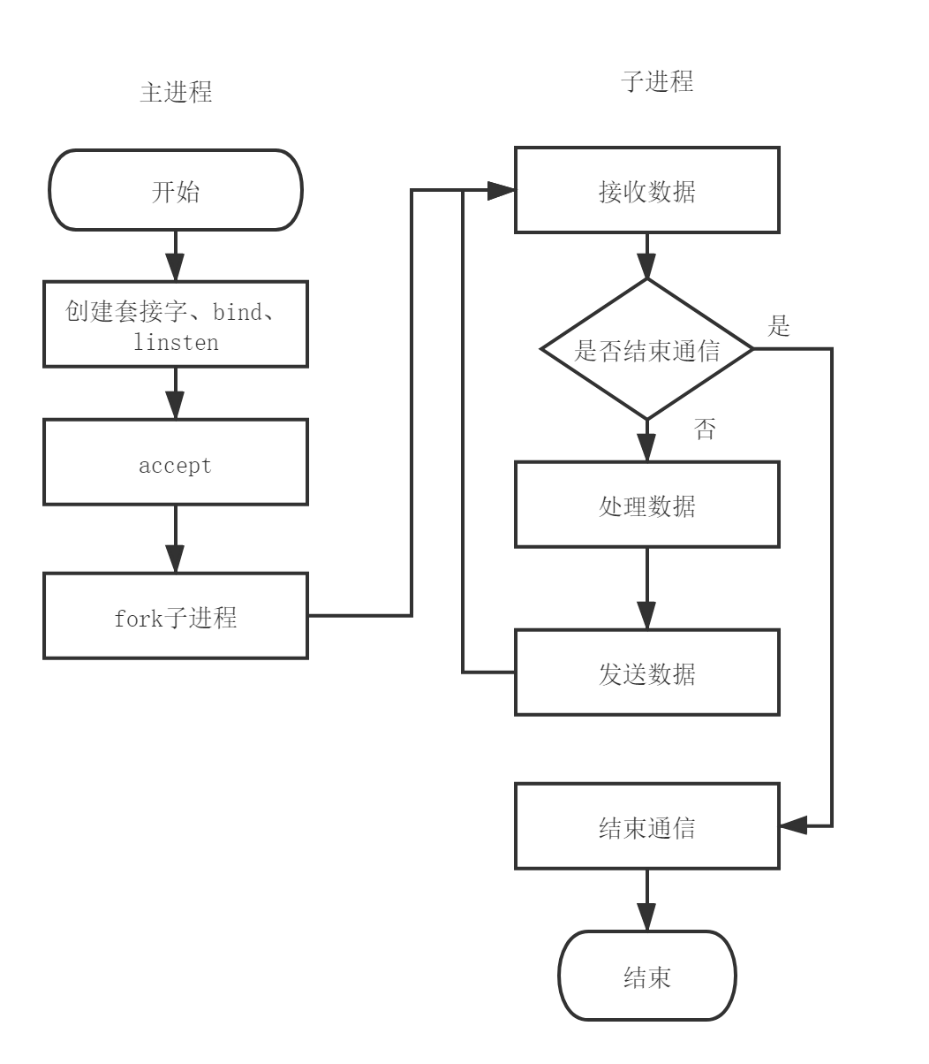


图2-3迭代服务器流程图

（2）并发处理器

并发服务器模型可以同时接受多个客户端的请求。通常有多线程和多进程两种模型。下面以多线程并发为例，在并发服务模型中，由主进程进行侦听和accept客户端请求，然后把请求放入一个服务队列，然后由有一个处理请求的线程池从请求队列不断地读取Socket请求进行处理，处理完成后线程继续读取队列，直到队列为空。这种服务器开发模型的优点是极大地提高了服务器的并发能力，并且编程也比较简单，不存在迭代模型的进程资源释放问题。缺点是该模型依然受限于服务器本地的资源，并发处理的线程数和请求服务队列受限于服务器的CPU和内存资源情况。

（3）异步服务器

异步服务器模型使用网络异步模型Select或Epoll（目前广泛使用的服务器一般采用Epoll），该模型单个进程就可以同时等待多个Socket的读写事件，大大地提高了网络操作的效率，目前常见的异步模型典型有：libevent。采用该网络模型的优点是网络操作高效，进程或线程切换开销，可以处理比并发模型更大的并发请求，缺点是编程复杂度较高，容易出错，另外还有一个缺点是业务代码和网络操作代码耦合，维护成本较高。

在本次课题中服务器端使用迭代服务器模型实现通信，这种模型兼含并发能力高，实现简单的特点。同时检查端发送检查请求将使用多线程技术实现并发地发送检查请求对多台企业服务器的不同文件进行完整性检查。

2.2 系统功能性需求分析

本论文的课题是企业服务器文件完整性检查系统的设计与实现，主要工作是基于MD5算法的文件完整性检测查统和基于TCP协议的通信系统，经过分析得到此系统的功能需求如下：

（1）文件初始信息获取与存储：能够根据客户需要将文件的初始信息，包括文件的大小、修改时间、文件的文件内容摘要值等准确存储在数据库中，以便系统进行文件完整性检查时能将文件的初始信息与文件的当前信息进行对比，从而得到检查结果。文件初始信息获取与存储用例规约如表2-1所示。

表2-1 文件初始信息获取与存储规约

|  |  |
| --- | --- |
| 用例名称 | 文件初始信息获取与存储用例规约 |
| 用例描述 | 该用例规定了获取与存储文件初始信息的过程 |
| 参与者 | 模型部署者 |
| 前置条件 | 已经完成数据库搭建以及编写好相关算法 |
| 后置条件 | 排除程序异常 |
| 基本事件流 | 从企业服务器中获取文件信息并存储至数据库 |

（2）检查结果获取：从企业服务器获取文件的当前信息并与数据库中文件的初始信息进行对比，并得到检查结果。检查结果获取的用例规约如表2-2所示。

表2-2 检测结果获取用例规约

|  |  |
| --- | --- |
| 用例名称 | 检测结果获取用例规约 |
| 用例描述 | 该用例规定了文件前后信息对比的过程 |
| 参与者 | 模型部署者 |
| 前置条件 | 文件初始信息已存储入数据库;  已编好当前文件信息与初始文件信息进行对比的算法 |
| 后置条件 | 排除程序异常 |
| 基本事件流 | 将文件前后的信息进行对比并得到检测结果 |

（3）通信系统搭建：检查端能够把检查请求放入一个服务队列，然后由有一个处理请求的线程池从请求队列不断地读取Socket请求进行处理，实现与服务器端的双向并发信息交互。服务器端使用迭代服务器模型实现与检查端的信息双向并发交互。通信系统搭建的用例规约如表2-3所示。

表2-3 通信系统搭建用例规约

|  |  |
| --- | --- |
| 用例名称 | 通信系统搭建用例规约 |
| 用例描述 | 该用例规定了检查端与服务器端信息交互的过程 |
| 参与者 | 模型部署者 |
| 前置条件 | 已编好通信程序 |
| 后置条件 | 排除程序异常 |
| 基本事件流 | 完成信息交互 |

（3）检查请求发送与存储：检查端能够使用搭建好的通信系统向服务器端发送检查请求，并将检查请求信息存储在数据库中，包括检查开始时间、检查方式、检查目录、检查频率、检查服务器IP地址。检查请求发送与存储的用例规约如表2-4所示。

表2-4 检查请求发送用例规约

|  |  |
| --- | --- |
| 用例名称 | 检查请求发送与存储用例规约 |
| 用例描述 | 该用例规定了检查请求的发送与存储过程 |
| 参与者 | 模型部署者 |
| 前置条件 | 已搭建好通信系统及数据库 |
| 后置条件 | 排除程序异常 |
| 基本事件流 | 发送与存储检查请求 |

（4）检查请求接收与检查结果发送：服务器端能够使用搭建好的通信系统接收检查请求并将检查结果发送给检查端，检查请求接收与检查结果发送的用例规约如表2-5所示。

表2-5 检查请求接收与检查结果发送用例规约

|  |  |
| --- | --- |
| 用例名称 | 检查请求接收与检查结果发送用例规约 |
| 用例描述 | 该用例规定了检查请求的接收与检查过程 |
| 参与者 | 模型部署者 |
| 前置条件 | 已搭建好通信系统及数据库 |
| 后置条件 | 排除程序异常 |
| 基本事件流 | 接收检查请求与发送检查结果 |

（5）检查结果可视化：检查端能够对接收到的检查结果进行可视化，根据检查结果发出弹窗警告，最终能够使用该系统将检查结果展示在程序图形界面中。检查结果可视化用例规范如表2-6所示。

表2-6 检查结果可视化用例规范

|  |  |
| --- | --- |
| 用例名称 | 检查结果可视化用例规范 |
| 用例描述 | 该用例规定了检查结果可视化过程 |
| 参与者 | 模型部署者 |
| 前置条件 | 已搭建好通信系统及数据库 |
| 后置条件 | 排除程序异常 |
| 基本事件流 | 展示接收到的检查结果；发出弹窗警告 |

2.3 系统可行性分析

可行化分析主要分析现有技术水平、经济条件等能否顺利完成企业服务器文件完整性检查系统的开发工作，各种配置能否满足开发的需求等。

企业服务器文件完整性检查系统涉及到大量的加密计算，网络交互的信息量较大，而这两个特点非常适合计算机和计算机网络的特点，发挥计算机的数据处理速度快、准确度高、计算机网络信息传输速度快的优势。

本系统使用Python语言进行企业文件完整性检查系统的程序编写，并基于MySQL数据库实现信息存储与提取。这几项相关技术现在均已成熟，发展使用都较好，所以本系统从技术角度来看是可实现的。

本系统的开发，可以通过虚拟机的技术模拟真实企业服务器存储环境，同时MySQL数据库也可部署在本地，因此只需要一台普通PC电脑即可，无需额外的成本投入。综上可行性分析，本论文的企业服务器文件完整性检查系统是可行的。

2.4 本章小结

本章对企业服务器文件完整性检查系统进行了需求分析，介绍了功能需求分析以及系统可行性分析。

其次，对本论文工作相关的理论知识进行了介绍，首先介绍了两种不同的文件完整性检查技术，其中重点分析了MD5算法的改进方法，这是本论文的基础性技术。

之后有介绍了基于TCP的通信技术，对于系统如何使用TCP协议进行信息交互，并对服务器并发模型进行了详细的介绍。

第3章 企业服务器文件完整性检查系统设计

本章主要对企业服务器文件完整性系统进行设计，主要从系统的整体架构、功能设计、功能模块的概要设计、系统数据架构设计、检查端与服务器端设计这些方面展开说明。

3.1 系统整体设计

3.1.1 系统架构设计

从系统架构方面来看，企业服务器文件完整性检查系统自下而上可以划分为数据层、文件信息获取存储层、信息交互层这三层，具体的层次架构图如图3-1所示。

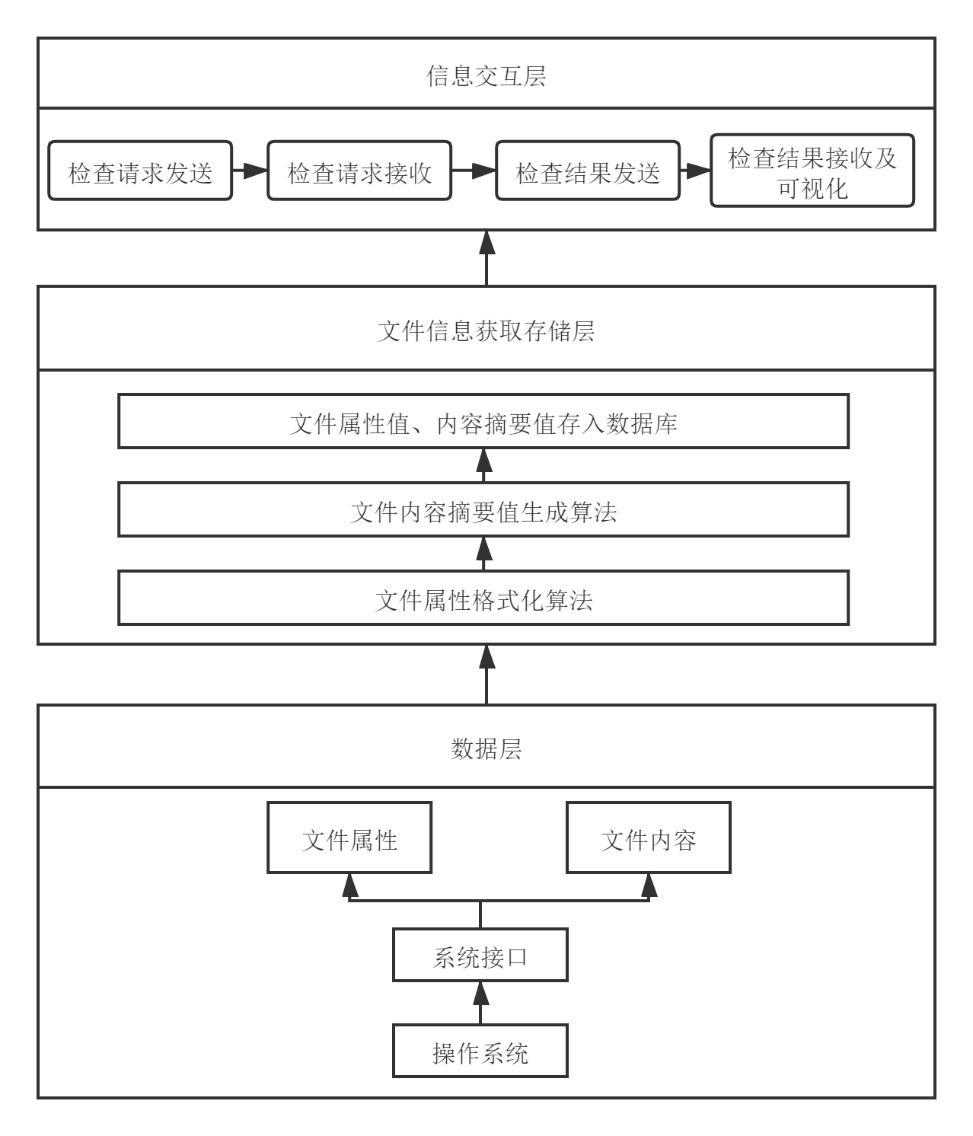


图3-1 系统架构设计图

数据层主要是用来解决文件初始数据获取的问题，根据图3-1所知是调用操作系统的接口来实现的，通过使用read()、stat()、walk()等系统函数可以获取文件目录下的文件名、文件属性、文件内容等信息。将获取的文件属性以适当形式存储、并将文件内容分块读取到内存中，等待下一步处理。

文件信息获取存储层主要是利用各种相关算法获取文件初始信息，并以适当的结构存储在数据库中。根据文件数据的处理过程，主要划分为文件属性格式化处理、基于MD5算法的文件内容摘要值生成、文件属性值、内容值存储这三个步骤。

信息交互层主要用来实现检查端与服务端之间的信息交互，具体包括检查端的检查请求发送与检查结果接收、服务器端的检查请求接收与检查结果发送、检查结果可视化。

3.1.2 系统功能设计

根据此系统对于功能性需求方面的分析，可以将系统划分为文件信息获取和存储、文件完整性检查模块和检测结果可视化这三大模块，具体划分如图3-2所示。

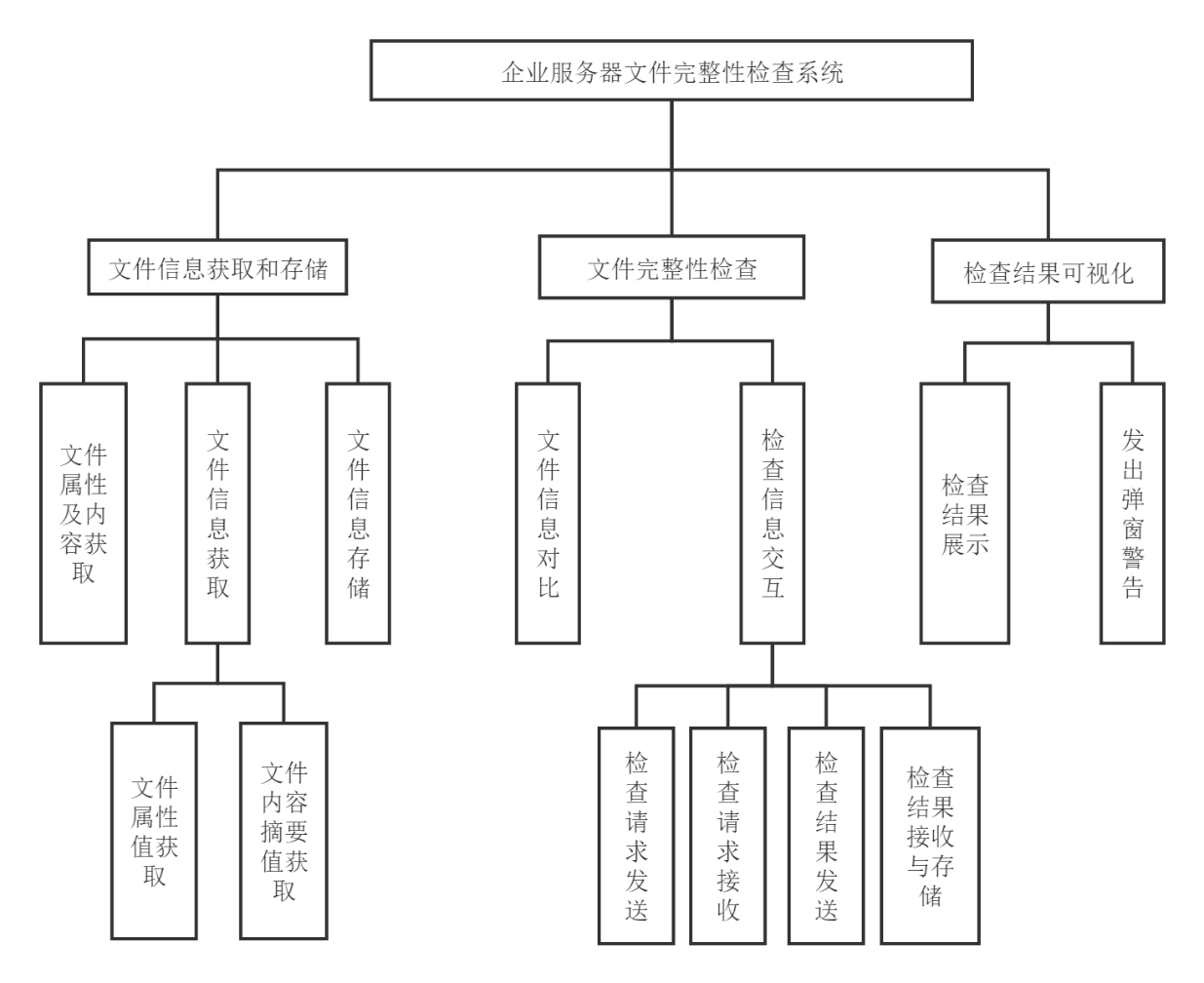


图3-2 系统功能模块划分示意图

文件信息获取和存储模块主要包括系统文件属性及内容获取、文件信息获取、文件信息存储这三个部分内容。文件属性、内容获取的功能主要是获取文件的相关属性、将文件内容读入内存。文件信息获取包括对得到的文件属性进行格式化处理、生成文件内容摘要值。文件信息存储用来实现将文件属性信息、以及文件内容摘要值合理存储，方便以后进行文件完整性检查使用。

文件完整性检查模块主要包括文件信息对比、检查信息交互这两部分内容。文件信息对比是实现将文件的当前信息与文件的初始信息进行对比，从而得到检查结果。检查信息交互是使用基于TCP的通信系统实现的，利用搭建好的通信系统检查端可以实现检查请求的发送、检查结果的接收与存储，服务器端可以实现检查请求的接收、检查结果的发送。

检查结果可视化模块主要包括检查结果展示和发出弹窗警告这两部分内容。检查结果展示主要实现检查端可以使用该系统实时查看检查结果以便采用相应的措施抵御攻击。发出弹窗警告实现当检查端接收到检查结果，判断有文件被篡改后能及时调用相关函数发出弹窗警告。

3.2 功能模块概要设计

3.2.1 文件信息获取和存储模块设计

文件信息获取和存储模块主要包括文件属性及内容获取，文件信息获取、文件信息存储这三个子功能，故下面从三者的概要设计方面进行描述。

(1)文件属性及内容获取

本系统将使用系统函数获取文件的属性以及内容，首先我们要获取文件目录下的所有文件的文件名。在Python语言中，我们可以使用os库中的walk（）函数实现。os.walk()用类似于深度遍历的方式遍历文件夹中的子文件夹以及文件，并得到一个三元Tupple(dirpath,dirnames,filenames)。其中dirpath是一个string，代表目录的路径，dirnames是一个list，包含了dirpath下所有子目录的名字。filenames是一个list，包含了该目录文件的名字。在得到list类型的filenames后，将其里面的元素依次作为参数传递给os.stat()函数，即可获得每个文件的属性信息。同时为了防止系统内存溢出，我们使用read()函数将文件按4096个字节分块读入内存中进行处理。

（2）文件信息获取

由于python语言中os.stat()函数返回的结果为class类型，为了方便存储，我们使用相关的算法将其转换为String类型。在该系统中，文件内容摘要值采用基于MD5算法计算得出，在Python语言中，我们可以使用hashlib库中的针对64位系统优化后的MD5算法进行处理，同时为了防止系统内存溢出及提高MD5算法的安全性，对文件进行了分块处理，将前一块文件内容生成的摘要值与下一块文件内容融合再继续生成摘要值，直到文件处理结束，得出最终摘要值，其流程如图3-3所示。

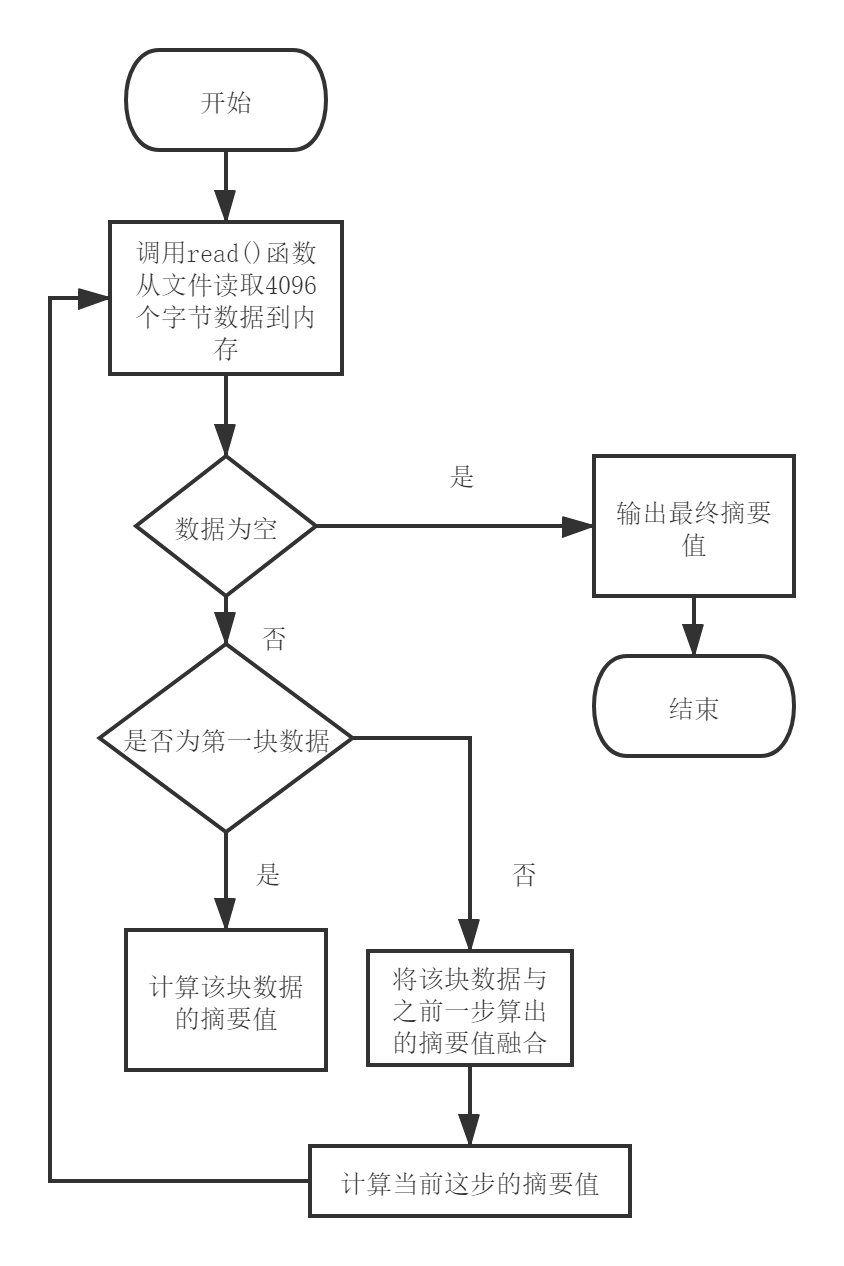


图3-3文件内容摘要值获取流程图

（3）文件信息存储

为保证数据的持久化及安全、不易修改，因此将文件信息存储在MySQL数据库中，在Python语言中可以使用pymysql库的相关函数，使用cursor()函数获取操作游标，进而执行SQL语句，将文件信息存储在Mysql数据库中。

3.2.2 文件完整性检查模块设计

该模块的目的是完成对企业服务器文件的完整性检查，检查端发送检查请求后，服务器端接收到检查请求，利用采用合适的对比算法，将前后两次文件信息进行快速对比，得到检查结果，并将检查结果及时发送给检查端，最后检查端将检查结果存储至数据库中。其主流程图如3-4所示。

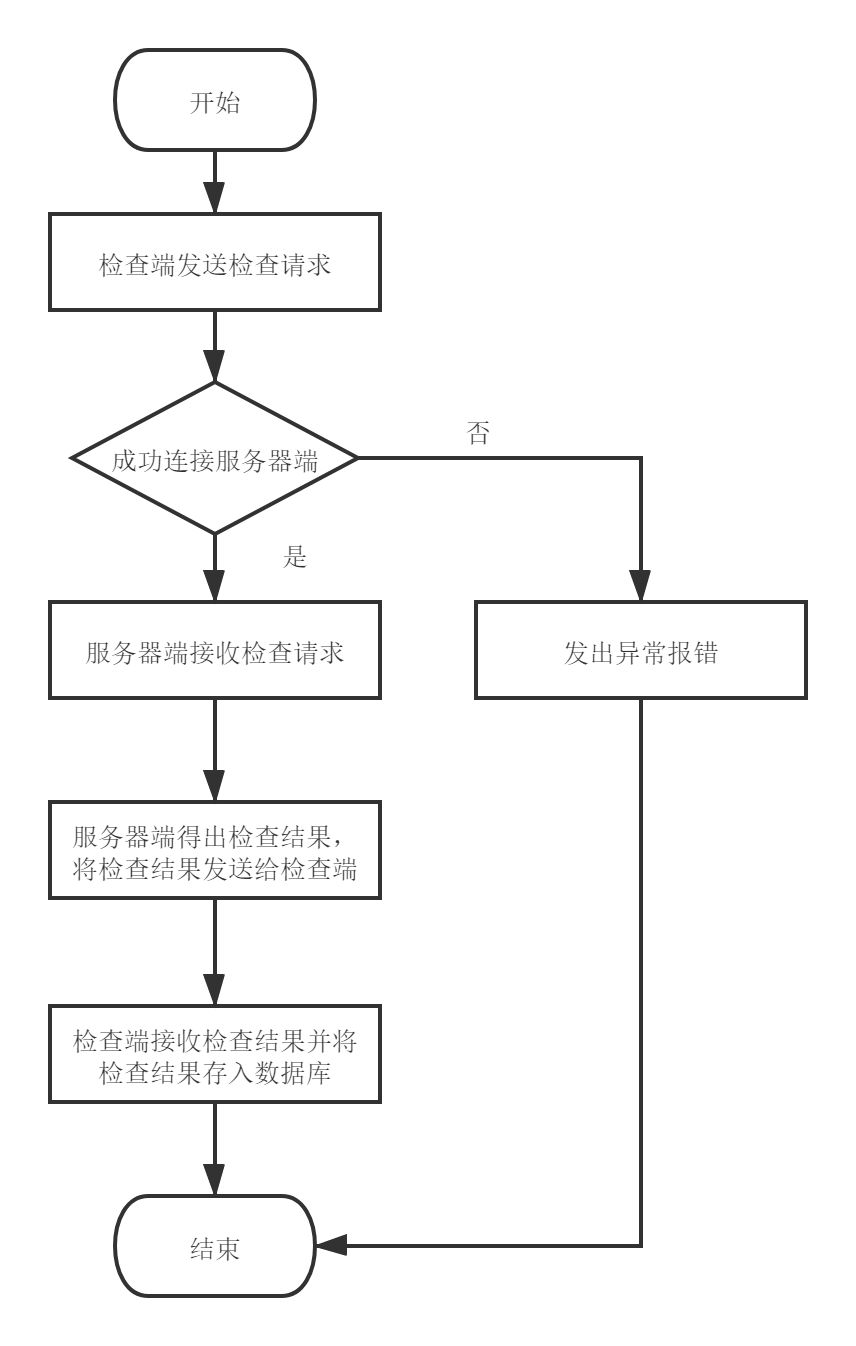


图3-4 文件完整性检查模块的主流程

3.2.3 检查结果可视化模块设计

该模块主要实现将MySQL数据库中的检查结果以一种合理的方式在程序中

中展示出来，同时实现当文件被篡改后，该系统能通过调用相关函数的方式发出

弹窗警告，提醒服务器管理人员及时查看检查结果并做出相应的措施来抵御入侵

攻击。

3.3 系统数据架构设计

系统使用MySQL数据库存储企业服务器文件的文件属性值、文件内容摘要值、在检查过程中产生的检查方案、检查结果。

为了避免单实例数据库遭受攻击时系统无法进行检查的问题，本次课题使用数据库冗余方案，一个数据库部署在本地服务器上，另一个数据库部署在云端服务器上，允许1个数据库实例发生故障，这2个数据库实例配置为主从式结构且开启双机热备，如图3-5所示。

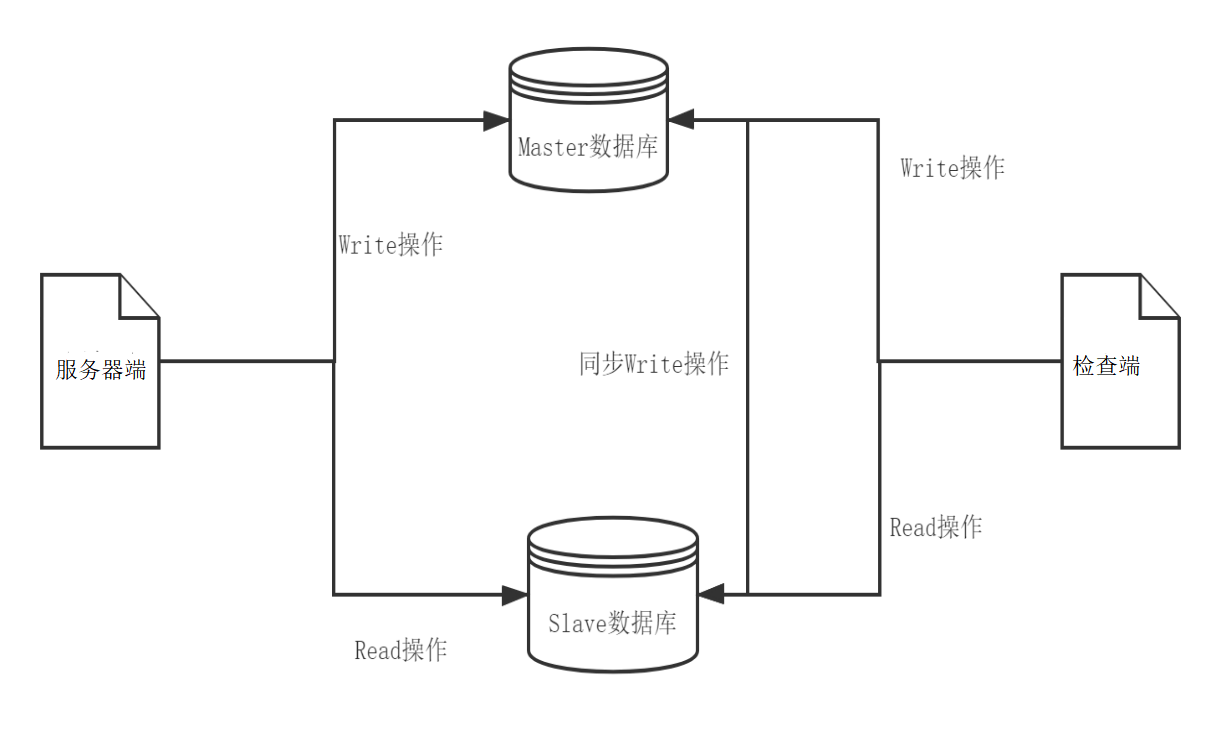


图3-5 系统数据架构设计

## 3.4 服务器端设计

围绕服务器端所需的功能，本节主要给出了服务器端文件信息对比与检查结果

发送的设计方案。

### 3.4.1 文件信息对比与检查结果发送

为了提高检查效率，系统不仅使用了MySQL的基于哈希表的索引技术来提高文件信息查找速度，同时采用多线程的方式进行文件信息对比，更快地得到检查结果。为降低系统通信负载该系统将同一文件目录下的检查结果融合为一个数据包统一发出。文件信息对比函数伪代码见表3-1。

|  |
| --- |
| 表3-1文件信息对比函数伪代码 |
| 函数名称：文件信息对比函数  函数输入：文件目录（p）、检查方式（m）、检查结果（r）  函数输出：检查结果 |
| 1.procedure comparem(p,m)  2. filenames = get\_dir\_name\_list(p)//获取文件目录p下的文件的文件名  3. for filename in filenames  4. t = threading.Thread(target = check,args = (p,m,r))//开启多线程对比  5. return r//返回检查结果  6.end procedure |
| 在获取检查请求时，客户端已建立其通信套接字，故发送检查结果时使用该套接字即可。由于文件信息返回的检查结果为list类型，需要将其转化为string类型。检查结果发送函数伪代码见表3-2。 |

|  |
| --- |
| 表3-2检查结果发送函数伪代码 |
| 函数名称:检查结果发送函数  函数输入：通信套接字（s）、检查结果（r）  函数输出：发送结果 |
| 1.procedure send\_result(s,r)  2. s = list\_to\_string(r)//将list类型的检查结果转为string类型  3. b = str.encode(s)//将string类型的检查结果转为byte类型  4. s.send(b)//将检查结果发送给服务端  5. if send\_result == successful do  6. Return 1  7. else  8. Return 0  9.end procedure |

## 3.5 检查端设计

从实际需求和使用场景出发，本节给出了检查端的设计方案，包括检查请求发送、检查结果接收与发出警告的设计方案。

检查端使用多线程技术向多个服务器的不同文件目录定期发送检查请求，再分别接收服务器端返回的检查结果，并将检查结果写入相应的数据库中。运行期间，会根据检查结果发出警告。检查端主功能流程图如图3-6所示。

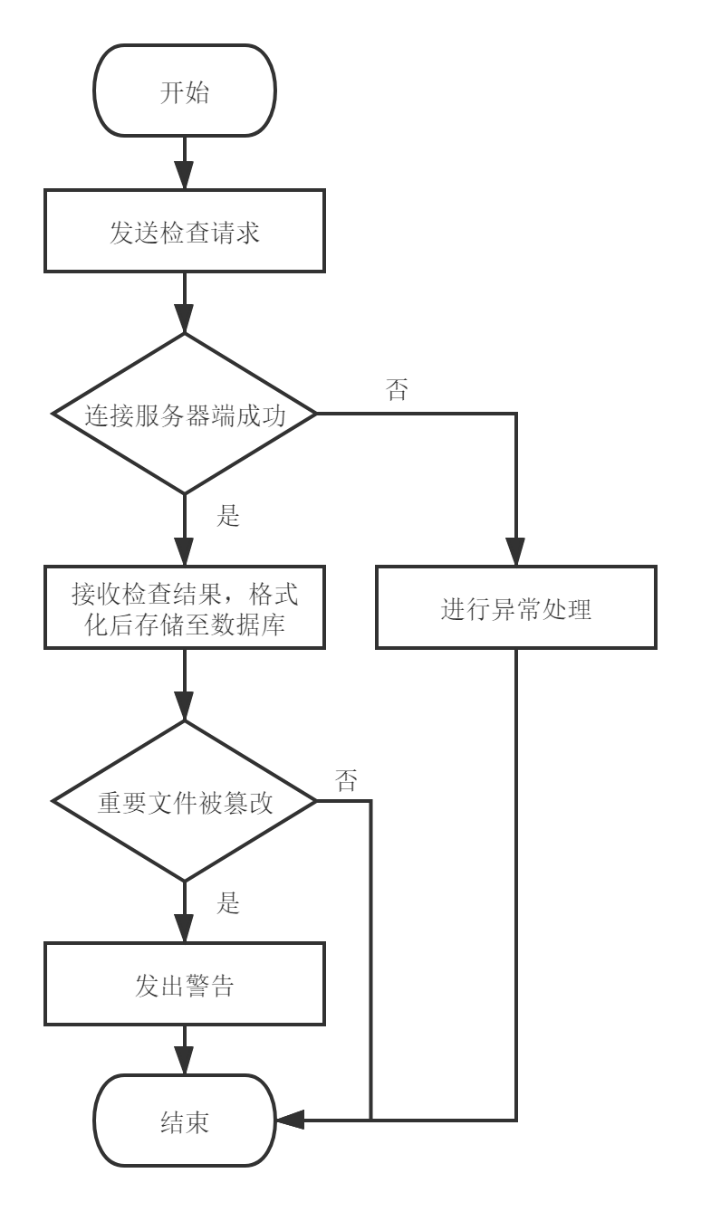


图3-6 检查端主功能流程图

### 3.5.1 检查请求发送

本系统使用基于TCP协议的多线程通信技术将检查请求发送给服务器端，基于TCP协议的多线程通信流程图如图3-7所示。

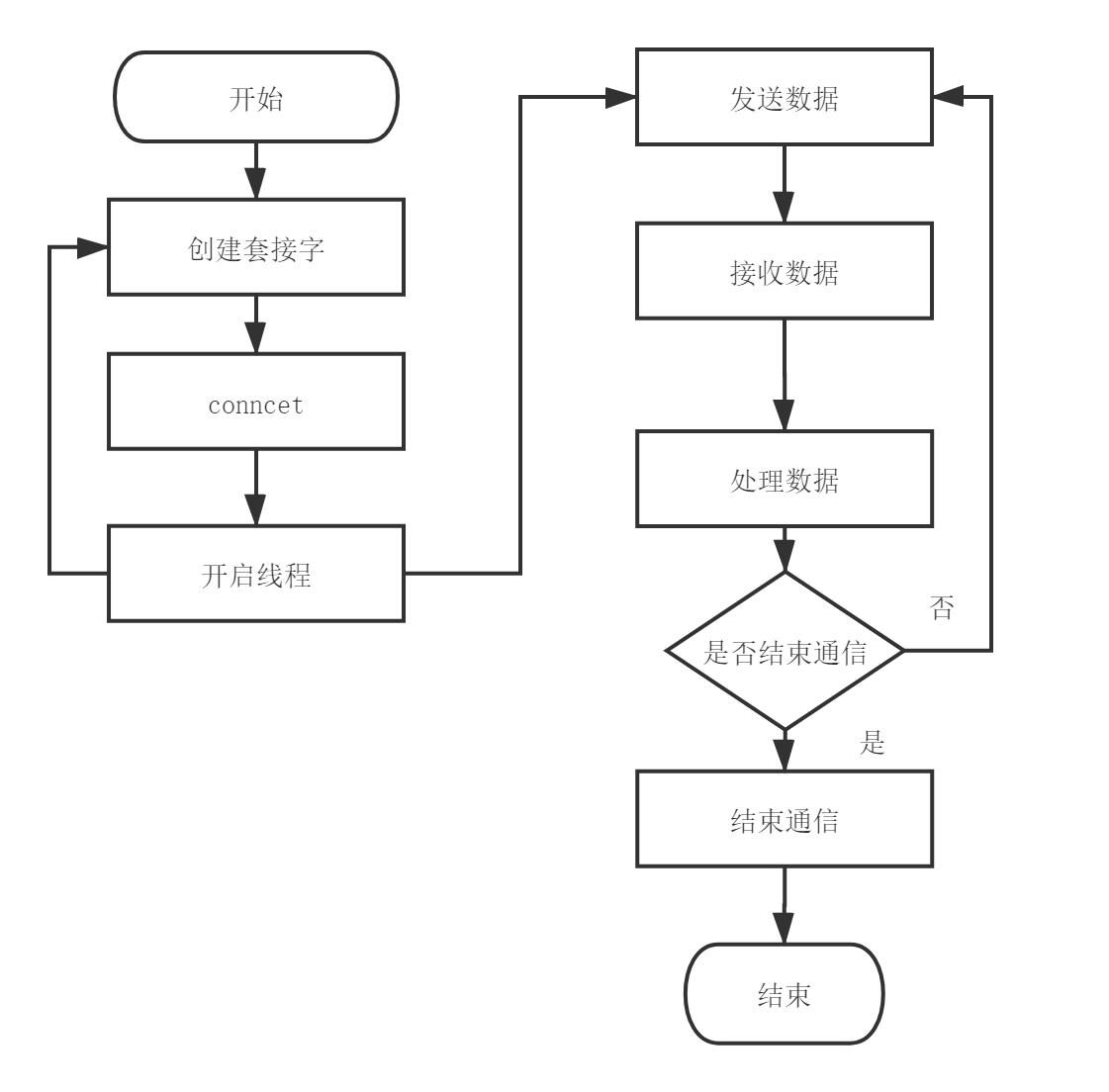


图3-7 检查端多线程通信流程图

检查请求发送函数伪代码见表3-3。

|  |
| --- |
| 表3-3 检查请求发送函数伪代码 |
| 函数名称：检查请求发送函数  函数输入：服务器IP（i）、服务器端口（p）、检查请求（r）  函数输出：无 |
| 1.procedure check\_request(i,p,r)  2. s=socket()//创建套接字  3. ip\_port = (i,p)  4. s.connect(ip\_port)//请求连接客户端  5. if connect\_result == ”successful” do  6. t = threading.Thread(target = send,args = (r,s))//创建线程进行通信  7. else  8. error()//进行异常处理  9. end if  10.end procedure |

### 3.5.2 检查结果接收与发出警告

检查端在运行过程中会接收到大量的检查结果，检查端需要将这些结果格式化处理以便后续存储并根据检查结果的信息发出警告。检查结果接收函数伪代码见表3-4。

|  |
| --- |
| 表3-4 检查结果接收函数伪代码 |
| 函数名称：检查结果接收函数  函数输入：通信套接字（s）  函数输出：检查结果 |
| 1.procedure result\_recv(s)  2. change\_list\_b = s.recv()//接收到的byte类型检查结果  3. change\_list\_s = bytes.decode(change\_list\_b)//转为string类型  4. change\_list=string\_to\_list(change\_list\_s)//以list类型格式化输出  5. return change\_list  6.end procedure |

当服务器中的重要文件遭受篡改时，检查端需及时通知审查人员查阅检查结果，采取相应措施来抵御入侵。发出警告函数伪代码见表3-5。

|  |
| --- |
| 表3-5 发出警告函数伪代码 |
| 函数名称：发出警告函数  函数输入：检查结果（l）  函数输出：无 |
| 1.procedure result\_warning(l)  2. for file in l do  3. if file is important do  4. warning() //弹出警告弹窗  5. end if  6.end procedure |

3.6 本章小结

本章阐述了企业服务器文件完整性检查系统的设计过程。首先描述了系统的整体设计，说明了系统的架构和功能。根据系统的功能性需求，在本章中进行了包括文件信息获取与存储、文件完整性检查、检查结果可视化功能模块的概要设计，对于各个功能的基础实现设计了初步方案，同时介绍了系统数据架构、检查端、服务器端、系统数据架构的设计方案。

第4章 企业服务器文件完整性检查系统实现

在企业服务器完整性检查的系统设计中，主要把此系统划分成了文件信息获取与存储模块、文件完整性检查模块和检查结果可视化模块，故本章将对功能模块的具体实现过程进行详细介绍。

4.1 文件信息获取和存储模块实现

4.1.1 文件属性与内容获取

获取文件的属性需要使用到Python语言中的os库中的函数，其中os.stat(path)用于在给定的路径path上执行一个系统stat（）函数的调用，返回包括文件大部分属性信息的Class类型statinfo对象。攻击者在修改文件后，通常文件的大小、最后一次修改时间、上次访问时间会发生改变即statinfo中st\_size、st\_mtime、st\_atimehi发生变化。使用该函数返回的statinfo的结构图如图4-1所示。由于该库函数为系统函数调用，耗费资源较小速率较快。

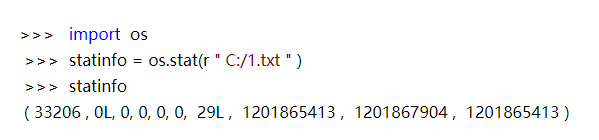


图4-1 statinfo结构图

获取文件的内容需要使用open()函数得到文件句柄，再使用read（）函数将文件内容按块读入内存。

4.1.2 文件信息获取与存储

在得到表述文件属性的Class类型stainfo对象后要对其进行格式化处理，可以将其中较为重要的st\_size（文件大小）、st\_mtime（修改时间）提取出来，从statinfo结构图中，我们发现这两个个属性均为浮点数类型，为了方便存储，我们可以将它们依次转换为string类型。

本次系统使用Python语言中的hashilib库计算文件内容的摘要值。hashlib是一个提供了一些流行的hash算法的python标准库，其中所包括的算法有md5、sha1、sha224、sha256、sha384、sha512。其中该库的md5算法对64位系统进行了优化，使在摘要值计算时变量长度始终与CPU寄存器长度一致，并适当减少了跳转指令，提高了文件内容摘要值生成效率。首先我们要使用hashlib.md5()函数生成一个hash对象指明使用的hash算法为md5，接下来对着这个hash对象重复调用hash.update(“文件内容分块”)，调用hash.hexdigest()函数返回文件内容摘要，作为十六进制数据字符串值。其伪代码如表4-1所示。

|  |
| --- |
| 表4-1 文件内容摘要值生成函数伪代码 |
| 函数名称：文件内容摘要值生成函数  函数输入：文件句柄（f）  函数输出：文件内容摘要值 |
| 1.procedure hashcompute(f)  2. m = hashlib.md5()//生成Hash对象，指明Hash算法为MD5算法  3. while True do  4. date = f.read(4096)//分块读取文件内容  5. if data is “empty” do  6. break  4. m.update(date)//迭代计算文件内容摘要值  5. return m.hexdigest()  6.end procedure |

本次系统使用MySQL数据库存储文件的初始信息，服务器端在运行后会在后

台自动在数据库中创建一个名为此台服务器IP地址的数据表，其字段设置如图4-

2所示。

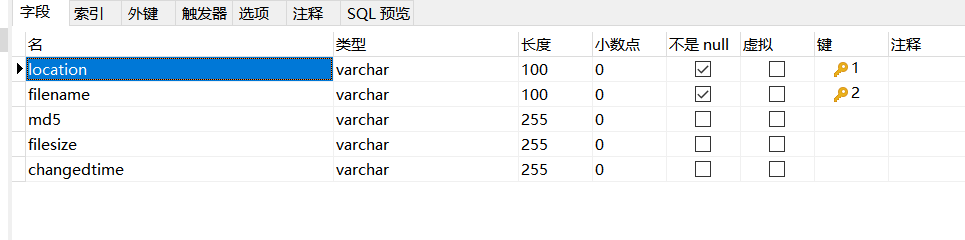


图4-2 文件信息数据表字段设置图

同时在本次系统中，对数据存在大量的查找操作，使用索引技术可以快速地

定位到第一个匹配的值，节省了大量地搜索时间，在MySQL数据库中通常使用

BTREE和HASH着两种技术来快速地定位索引值。该数据表的索引设置如图4-3

所示。

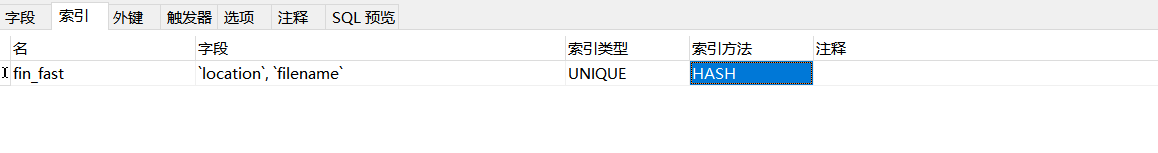


图4-3 文件信息数据表索引设置图

4.2 文件完整性检查模块实现

4.2.1 文件信息对比

由以上步骤得到了文件的初始信息并将其正确存储在了数据库中，如何快速地将文件的初始信息与当前信息进行对比，从而得到检查结果成为影响本系统检查效率的关键因素之一。

对于文件属性信息进行对比时，我们首先比较文件的修改时间是否发生变化，如未发生变化再比较文件的大小是否发生变化，最终得到检查结果，这样可以减少信息读取与对比的次数，从而提高检查效率。其流程图如图4-4所示。

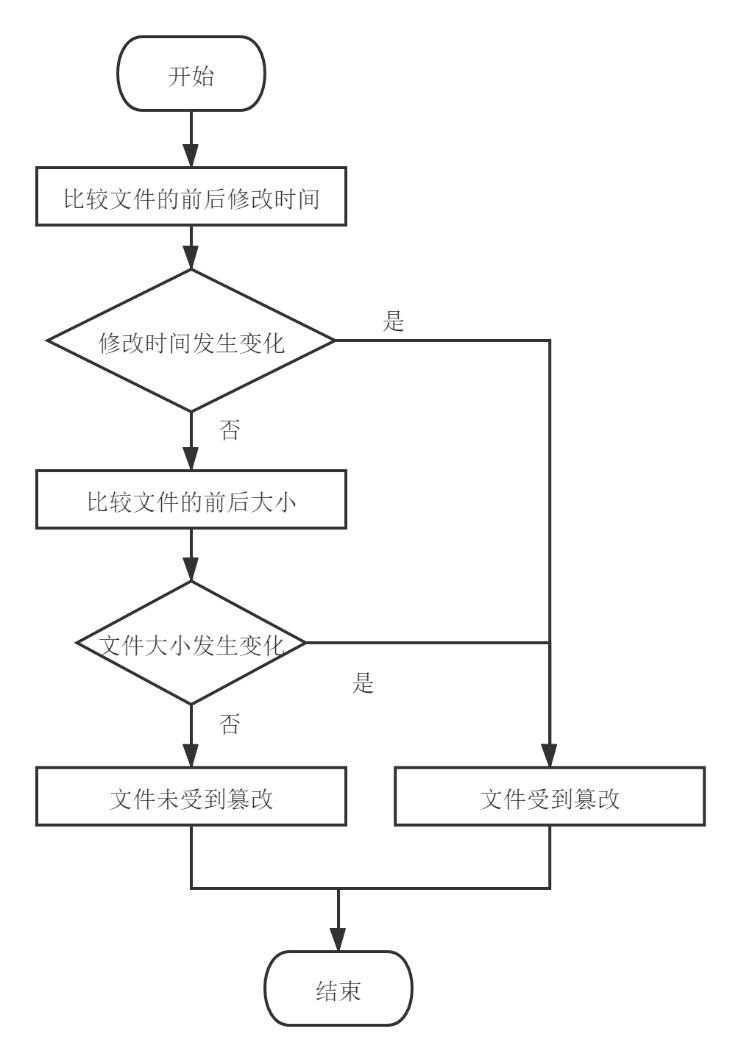


图4-4 文件属性信息对比流程图

对于文件内容摘要值进行对比时，在Python语言中字符串对比存在三种匹配方式：“in”、“==”和正则三种方式，为提高系统检查效率，选择合适的匹配方式，采用实验的方法对这三种的匹配方式的匹配效率进行对比，实验代码如图4-5所示，实验结果如图4-6所示。

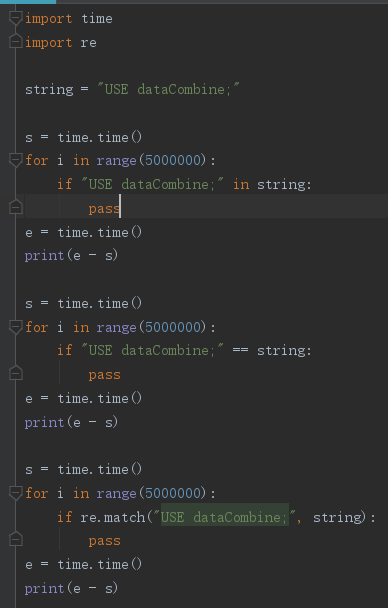


图4-5 匹配方式的匹配效率对比实验代码图

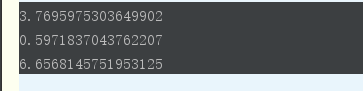


图4-6 匹配方式的匹配效率对比实验结果图

根据实验结果可得在Python语言中使用“==”这种匹配方式的匹配效率最高，因此该系统使用“==”的匹配方式将文件内容初始摘要值及当前摘要值进行对比，同时在Python中可以使用threading库并发地对文件信息进行对比，从而更加高效地得到对比结果，通常两种方式来创建多线程：一种是继承Thread类，并重写它的run（）函数；另一种是实例化threaing.Thread对象时，将线程要执行的任务函数作为参数传入线程，第二种方法编程更为简单且代码易读性更强，故该系统使用第二种方法。由于当前的对比结果信息量较小且需要立即被传输，故无需将其存储至数据库中，使用List类型进行保存即可。

4.2.2 检查信息交互

（1）检查端检查请求发送

在Python语言中使用socket库、threading库中的相关函数即可以实现对多台服务器的不同文件发送检查请求。在具体实现过程中，由于TCP协议中，数据包发送的时延较长，为了提高系统信息交互的速率，检查端将检查请求信息（如检查目录、检查方式）融合为一个字符串发送给服务器端，字符串融合采用Python中常用的字符串相加即可，为了方便服务器端识别，中间使用“.”作为分隔符。

（2）服务器端接收检查请求与发送检查结果

在Python语言中使用socket库、multiprocessing库中的相关函数可以完整并发服务器的搭建，从而服务器端可以并发处理多个检查请求。其监听主进程伪代码如表4-2所示。

|  |
| --- |
| 表4-2 监听主进程函数伪代码 |
| 函数名称：文件内容摘要值生成函数  函数输入：套接字（s）、服务器IP地址（i）  函数输出：无 |
| 1.procedure listen()  2. ip\_port = (ip,8080)//指明套接字绑定的IP地址、端口号  3. s.bind(ip\_port)  4. while True do  5. conn,client\_addr = s.accept()//服务器端接收连接请求，生成相应的通信套接字  6. p=Process(target = talk,args = (conn,))//创建子进程进行通信  7. p.start()  8.end procedure |
|  |

服务器端接收到检查请求信息后，首先要将Byte类型的数据转换为字符串类型，然后使用split函数进行划分，得到检查目录、检查方式，接下来根据检查请求信息，对文件进行文件完整性检查，得到一个List类型的检查结果change\_list，其中保存了发生变化的文件的文件名，由于List类型无法直接使用socket发送，故需要将其转换为字符串类型，其转换函数如表4-3所示。

|  |
| --- |
| 表4-3 List转换为字符串函数伪代码 |
| 函数名称：List转换为字符串函数  函数输入：List（l）  函数输出：字符串（s） |
| 1.procedure list\_to\_string(l)  2. s = ””//初始字符串  3. for i in list do  4. s = s+i+”?”//由于文件名中不能含”？”，故使用”?”进行划分  8.end procedure |

服务器端通信进程伪代码如表4-4所示

|  |
| --- |
| 表4-4 服务端通信进程函数伪代码 |
| 函数名称：服务端通信进程函数  函数输入：套接字（s）  函数输出：无 |
| 1.procedure talk(s)  2. while true do  3. try do  4. receive = s.recv(1024)  5. path = receive.split(“.”)[0]//获取检查目录  6. rec\_method = receive.split(“.”)[1]//获取检查方式  7. if rev\_method == “m” do  8. change\_list = dir\_md5\_compare(path,ip)//比较文件的前后内容摘要值  9. change\_list\_s = list\_to\_sting(change\_list)//转为字符串类型  10. change\_list\_b = str.encode(change\_list\_s)//编码为Byte类型  11. s.send(change\_list\_b)//将检查结果发送给检查端  12. if rev\_method == “s” do  13. change\_list = dir\_sizetime\_compare(path,ip)  14. change\_list\_s = list\_to\_sting(change\_list)//转为字符串类型  15. change\_list\_b = str.encode(change\_list\_s)//编码为Byte类型  16. s.send(change\_list\_b)//将检查结果发送给检查端  17. if not receive do  18. break  19. except Exception do  20. break  21.end procedure |

(3)检查端接收与存储检查结果

检查端可以使用建立起连接的套接字获取服务器端发送的检查结果，由于接收到的为一个包含所有检查结果的字符串，为了方便处理检查结果，可以将该字符串根据分隔符转换为List类型，检查结果接收函数伪代码如表4-5所示。

|  |
| --- |
| 表4-5 检查结果接收函数伪代码 |
| 函数名称：检查结果接收函数  函数输入：套接字（s）  函数输出：List类型的检查结果 |
| 1.procedure list\_to\_string(s)  2. change\_list\_b = s.recv(1024)  3. change\_list\_s = bytes.decode(change\_list\_s)//将接受到的字节数据解码为字符串  4. change\_list\_l = string\_to\_list(change\_list\_s)  5. return change\_list\_l  8.end procedure |
|  |

检查端需将接收到的检查结果以合适的结构存储在MySQL数据库中，其数据表字段设置如图4-7所示。

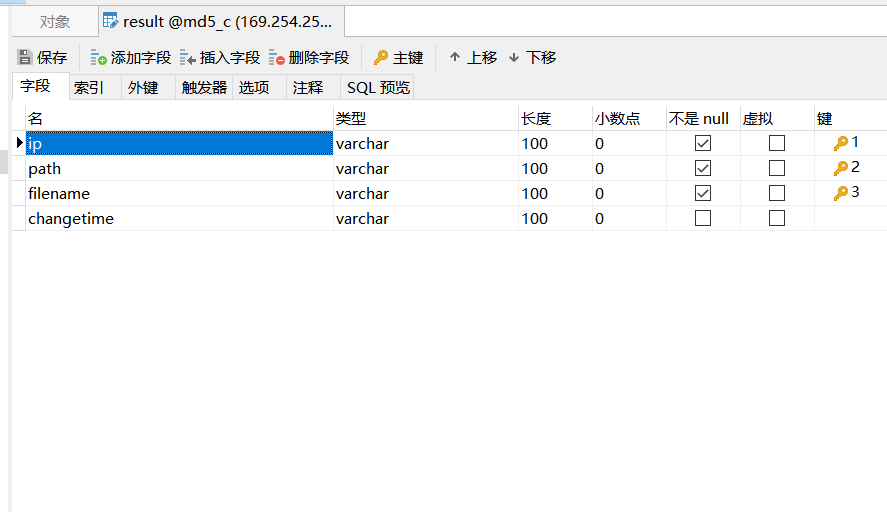


图4-7 检查结果数据表字段设置图

由于系统对该数据表的操作为大量的插入操作，且不存在查找操作，使用索引不仅会降低插入操作的速率而且是没有意义的，故不对该数据表进行索引设置。

在Python语言中，我们可以使用pymysql库中的相关函数执行数据插入操作，MySQL数据库数据插入函数伪代码如表4-6所示。

|  |
| --- |
| 表4-6 MySQL数据库数据插入函数伪代码 |
| 函数名称：MySQL数据库数据插入函数  函数输入：插入数据（v）  函数输出：无 |
| 1.procedure value\_insert(v)  2. db = pymysql.connect(“ip地址”，“用户名”，“密码”,“数据库名”)//连接至数据库  3. cursor = db.cursor()//获取操作游标  4. sql=f”insert into  表名 values(v)”//sql数据插入语句  5. cursor.execute(sql)//执行该语句  6.end procedure |
|  |

4.3 检查结果可视化模块实现

4.3.1 检查结果展示

本次系统的图形界面设计采用QT Designer实现，在Pycharm中可以使用内置工具将QT Designer生成的.ui文件转换为.py文件，同时使用Python语言的pyqt5库对信号进行设置，并编写相应的槽函数。首先系统要将数据库中的检查结果读取到内存中，检查结果读取函数伪代码如表4-7所示。

|  |
| --- |
| 表4-7 检查结果读取函数伪代码 |
| 函数名称：MySQL数据库数据插入函数  函数输入：检查结果数据表表名（r）  函数输出：检查结果（l） |
| 1.procedure result\_fetch(r)  2. db = pymysql.connect(“ip地址”，“用户名”，“密码”,“数据库名”)//连接至数据库  3. cursor = db.cursor()//获取操作游标  4. sql = f”select \* from r”  5. cursor.execute(sql)//执行该语句  6. result = cursor.fetchall()  7. for row in result do  8. l.append(row)  12. return l  13.end procedure |

经过上述操作，已经将数据库中检查结果数据表的所有数据读取到一个List类型的对象中，接下来要对该对象进行分组操作，将List中的每四个元素划为一组。分组操作函数伪代码如表4-8所示。

|  |
| --- |
| 表4-8分组操作函数伪代码 |
| 函数名称：分组操作函数  函数输入：检查结果数据表中所有数据（l）  函数输出：分组后的检查结果（g） |
| 1.procedure group(l)  2. for i in range(0,len(l),4) do  3. g.append(l[i:i + 4])  4. return g  5.end procedure |

经过上述操作我们到了一个合理存储检查结果的二元数组，可以很容易地通过调用pyqt5库中函数，将检查结果展示在Qwidget类中的tableWidget上，检查结果展示伪代码如表4-9所示。

|  |
| --- |
| 表4-9检查结果展示函数伪代码 |
| 函数名称：分组操作函数  函数输入：分组后的检查结果（g）、展示表格（t）  函数输出：无 |
| 1.procedure result\_view(g,t)  2. lenth = 0  3. while lenth<len(b) do  4. r = t.rowCount()//获取当前表格的行数  5. t.insertRow(r)//插入一行  6. t.setItem(r,0,g[lenth][0])  7. t.setItem(r,0,g[lenth][1])  8. t.setItem(r,0,g[lenth][2])  9. t.setItem(r,0,g[lenth][3])//检查结果写入展示表格中  10. lenth=lenth+1  11.end procedure |

4.3.2 发出弹窗警告

在Python语言中可以使用pyqt5库中的Qmessagebox类进行弹窗警告，当检查端接收到检查结果后，将根据检查结果判断是否有新的文件发生篡改，如果存在新的文件被篡改的情况，将发出弹窗警告，提醒审查人员查看检查结果，并做出相应的措施抵御入侵攻击，弹窗警告效果图如图4-8所示，检查结果展示效果图如图4-9所示。

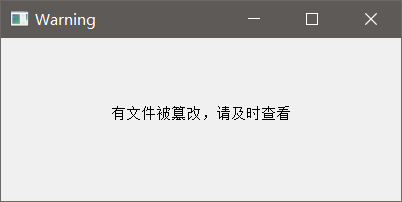


图4-8 弹窗警告效果图

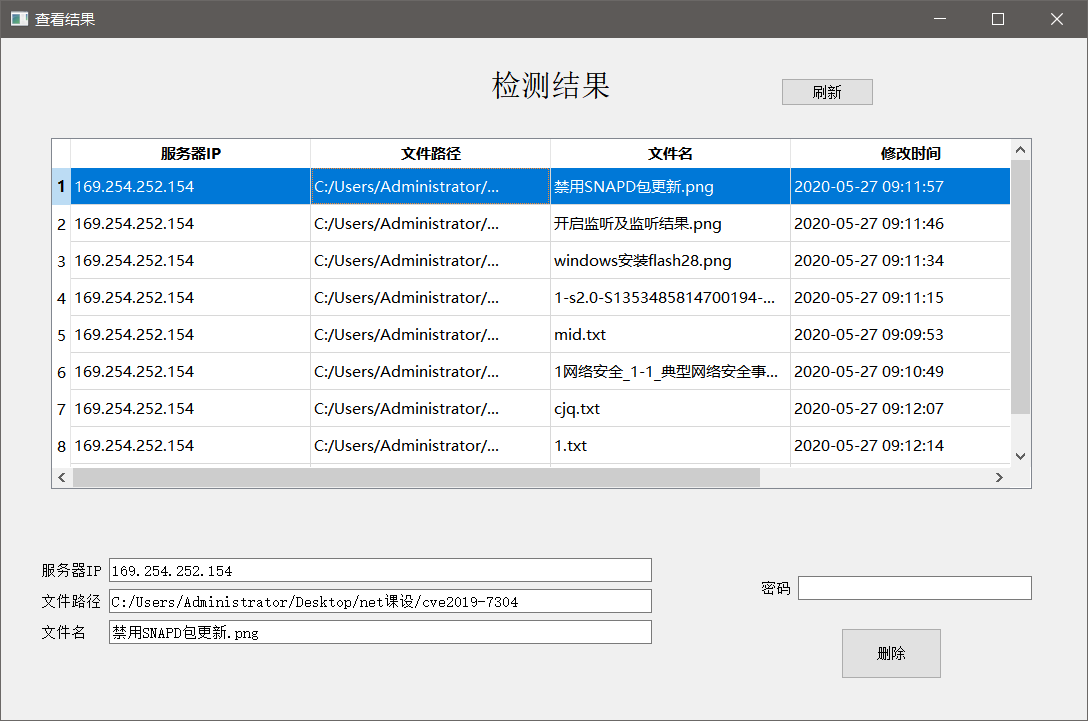


图4-9 检查结果展示效果图

4.4 本章小结

本章主要对企业服务器文件完整性检查系统的实现过程进行说明，实现功能模块包括文件信息获取和存储模块、文件完整性检查模块和检查结果可视化模块。同时利用流程图、伪代码的形式对各个函数进行了描述说明，并针对数据库的搭建进行了分析与描述，最终实现了企业服务器文件完整性检查系统，并展示了相关的页面设计。

第5章 系统测试与结果分析

本章将针对复杂网络环境下企业服务器文件完整性检查系统进行详细测试，搭建相关的测试环境，充分设计并验证了测试用例，并且测试用例完整覆盖了系统可能存在的各项功能、逻辑缺陷以及可能的实施问题。最后针对测试过程和测试结果，给出了对于系统整体质量的评价。

## 5.1 系统功能测试

本节主要通过配置恰当的测试环境并编写恰当的测试用例，以期全面覆盖系统功能，测试各项功能是否能够充分满足需求，发现并解决潜在的问题。

### 5.1.1 测试说明

对于这次课题设计的系统，测试重点如下：

（1）本系统在信息交互方面的准确性，包括检查请求发送与接收、检查结果发送与接收方面的准确性。

（2）本系统在检查过程方面的准确性，包括是否能够准确得到文件完整性检查结果。

（3）本系统在数据存储方面的准确性，包括是否能够准确将检查结果、检查方案写入MySQL数据库中。

（4）本系统的稳定性、计算能力方面的水平。

根据实际测试需要，本次课题测试环节使用Windows环境下的文件进行测试，测试样本包括25个不同目录下的共计427个文件，其中两个超过1G的文件用于测试系统的计算能力，其余文件用于测试系统稳定性、实时性、准确性。

### 5.1.2 测试用例

（1）信息交互准确性测试

为了测试检查信息在网络中交互是否准确，编写测试用例如表5-1所示。

表5-1 信息交互准确性测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 功能子项 | 输入与操作说明 | 期望测试结果 | 实际测试结果 |
| 信息交互  是否准确 | 使用该系统中的检查端发送检查请求，然后使用wireshark捕获数据包，与期望数据包进行对比 | 捕获的数据包与期望数据包一致 | 捕获的数据包与期望数据包一致 |

（2）检查结果准确性测试

为了测试检查结果是否准确，编写测试用例如如表5-2所示。

表5-2 检查结果准确性测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 功能子项 | 输入与操作说明 | 期望测试结果 | 实际测试结果 |
| 检查结果是否准确 | 准备一个文件目录，并对目录下的文件进行修改。一方面使用该系统对文件目录的文件进行完整性检查，一方面使用MD5 Checksum Verifier进行检查 | 两方面的检查结果一致 | 两方面的检查结果一致 |

（3）数据存储准确性测试

为了检查数据能否准确写入数据库，编写测试用例如表5-3所示。

表5-3 数据存储准确性测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 功能子项 | 输入与操作说明 | 期望测试结果 | 实际测试结果 |
| 数据能否准确写入数据库中 | 准备一个文件目录，并对目录下的文件进行修改。使用该系统对文件目录的文件进行完整性检查，一方面使用txt文件存储检查结果、检查方案，一方面使用Navicat数据库管理工具查看检查结果、检查方案。 | 两方面的检查结果、方案一致 | 两方面检查结果、方案一致 |
|  |  |  |  |

（4）系统综合方面测试

为了对系统的稳定性、计算能力进行测试，编写测试用例如表5-4所示。

表5-4系统综合方面测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 功能子项 | 输入与操作说明 | 期望测试结果 | 实际测试结果 |
| 系统稳定性 | 模拟系统环境不稳定的情景，当系统突然重启、或者本地数据库不可用时，检查数据完整性、数据可用性。 | 数据完整、可用 | 数据完整、可用 |
| 系统计算能力 | 对有数据较大的文件的文件目录进行完整性检查，测试系统在计算压力较大时的稳定性。 | 系统正常运行 | 系统小概率发生崩溃现象，经查是磁盘占用率过高导致。 |

## 5.2 系统性能测试

为了测试系统性能表现，本次课题使用25个文件目录中的文件作为测试用例并在本地服务器正常使用网络资源的环境下进行压力测试，25个文件目录中包括2个1G以上的文件，475个普通多种类型文件。

测试环境：使用单核主频2.5GHz的4核4线程处理器、4G内存的Windows10机器部署为MySQL主从数据库，同时将检查端程序安装在该台机器上。

使用单核2.3GHz的4核4线程处理器、8G内存的Windows7机器模拟为企业服务器，将服务器端程序安装在该台机器上。服务器端检查端处于同一网段。测试环境网络拓扑图见图5-1。

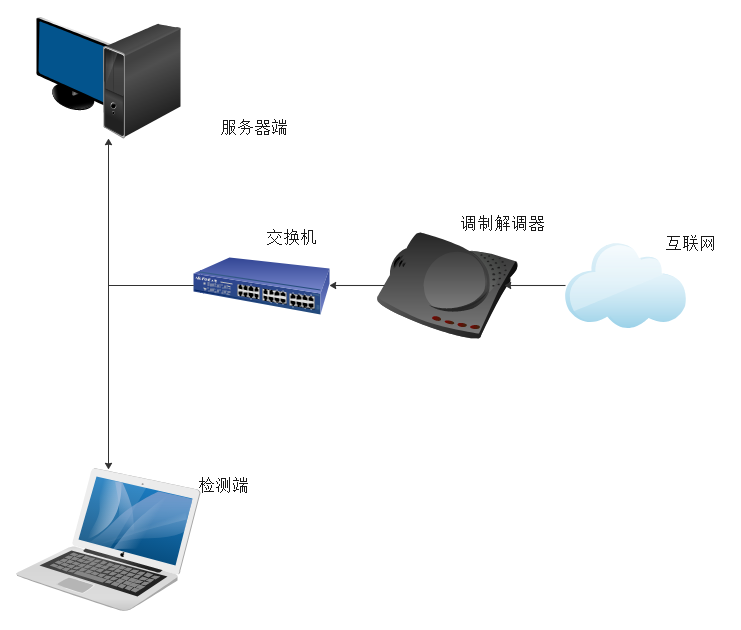


图5-1 测试环境网络拓扑图

系统性能测试主要包括实时性、并发处理量、抗压能力等指标。为全面覆盖上述指标，编写测试用例如表5-5所示。

表5-5 系统性能测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 功能子项 | 输入与操作说明 | 期望测试结果 | 实际测试结果 |
| 实时性 | 服务端发送检查请求，并对检查目录下的文件进行修改。 | 客户端实时处理检查请求，在文件检查结束后，系统立即发出警告。 | 客户端实时处理检查请求，在文件检查结束后，系统立即发出警告 |
| 并发处理量 | 服务端发出对所有文件目录进行检查的检查请求。 | 系统正常运行，且能及时得到检查结果。 | 系统正常运行，且能及时得到检查结果。 |
| 抗压能力 | 本地服务器大量网络资源被其他程序占用 | 能够抵抗异常网络环境的干扰，保障业务连续性。 | 系统能继续运行，没有崩溃，但系统效率大幅下降。 |

## 5.3 测试结果与分析

根据测试结果，系统功能基本按照设计规划全部实现、各项功能工作正常稳定输出；系统性能方面，实时性、并发处理量和容错能力均能够满足实际需求。测试过程暴露出的部分问题，大多与机器性能及网络环境相关，在改善机器性能及网络环境后，可以得到完美解决。

## 5.4 本章小结

为检验系统功能是否按照设计需要实现了各项功能，我们充分设计了各项测试用例，并根据测试结果得出了相关结论；为考量系统的抗压能力，在5.2节系统性能测试中的各项测试数据充分说明了系统性能。

# 结 论

本次课题基于互联网环境设计并实现了企业服务器文件完整性检查系统，现将本文研究工作做出以下概括：

（1）论文主要工作：本文对当前的文件完整性检查技术进行了研究，并根据实际需求综合应用/创新相关技术，实现了基于MD5算法的文件完整性检查技术、基于TCP协议的通信技术，成功实现系统的相关功能，保障各项需求得到充分满足。

（2）论文创新点：定制化设计了文件内容摘要值获取技术，确保系统能够稳定、快速地生成文件内容摘要值；针对TCP协议的特点，创新信息融合技术，降低了数据包发送次数，提高信息交互的速率；通过使用多线程技术对文件信息进行对比，有效提高了检查效率；数据库设计方面使用MySQL支持的主从同步、双机热备技术可避免依赖单一数据库引发的单点故障问题。

目前，本次课题设计实现的系统能够完整实现各项功能需求，但仍有进一步提升的空间，现说明如下：

（1）更好的文件内容摘要值获取技术：当前使用的算法在计算大文件的文件内容摘要值效率较低，未来可以使用云计算技术更高速地获取文件内容摘要值。

（2）更快的信息交互技术：当前的通信采用TCP协议，存在时延较高的缺陷，尤其是在发送数据报数量过多的时候。未来可以通过UDP与TCP结合的技术来实现多进程通信提高信息交互速率。

（3）更强的稳定性和适应能力：包括应对网络异常、系统重启、断电等极端环境的适应能力。

# 参考文献

[1] 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.

[2] 温强. 文件完整性检测的设计与实现[J]. 科技情报开发与经济, 2011, 21(09): 137-139.

[3] 田楠. 基于网络的入侵检测技术的研究[J]. 信息与电脑(理论版), 2011(09): 155-156.

[4] 许军. 浅谈入侵检测系统——IDS[J]. 广西大学梧州分校学报, 2004(04): 75-77.

[5]陈昕, 杨寿保. 入侵检测系统中文件完整性检查的研究[J]. 计算机科学, 2002(09): 101-103.

[6]陈琦珺, 陈性元, 唐慧林, 牛琳. 面向抗攻击测试的完整性检测系统设计[J]. 微计算机信息, 2007(33): 65-67.

[7]姚玉献. 浅析网络安全与入侵检测系统[J]. 网络安全技术与应用, 2007(12): 23-25.

[8]石蕊, 高立刚, 马一腾. 企业级文件完整性监控技术浅析[J]. 中国金融电脑, 2012(12): 43-45.

[9]Mark Kedgley. File integrity monitoring in the modern threat landscape[J]. Network Security, 2014, 2014(2).

[10]Patrick Bedwell. Finding a new approach to SIEM to suit the SME environment[J]. Network Security, 2014, 2014(7).

[11] H. Watanabe, T. Iwama, K. T. Murata, S. Ushiyama and Y. Muto, Improvement of the Integrity Verification Application Using Timestamp Mechanism for Distributed File System, [J]2014 IEEE 38th International Computer Software and Applications Conference Workshops, Vasteras, 2014, pp. 158-163.

[12] Vinı́cius da Silveira Serafim,Raul Fernando Weber. Restraining and repairing file system damage through file integrity control[J]. Computers &amp; Security, 2004, 23(1).

[13] Xiongwei Fei,Kenli Li,Wangdong Yang,Keqin Li. A secure and efficient file protecting system based on SHA3 and parallel AES[J]. Parallel Computing, 2016, 52.

[14] Pinheiro Alexandre,Dias Canedo Edna,de Sousa Junior Rafael Timoteo,de Oliveira Albuquerque Robson,García Villalba Luis Javier,Kim Tai-Hoon. Security Architecture and Protocol for Trust Verifications Regarding the Integrity of Files Stored in Cloud Services.[J]. Sensors (Basel, Switzerland), 2018, 18(3).

[15] 刘海永. 云计算环境下文件完整性检测系统的设计[J]. 信息技术与信息化, 2019(03): 61-64.

[16] 贾倍. 云存储中大数据完整性证明机制研究[D]. 西安电子科技大学, 2018.

[17] 姜红, 亢保元, 李春青. 改进的保护身份的云共享数据完整性公开审计方案[J]. 信息网络安全, 2018(10): 85-91.

[18] 刘广沛. 基于区块链的云数据完整性保护机制[D]. 南京邮电大学, 2018.

[19] 张永棠. 一种移动可信平台ELF文件完整性校验方法[J]. 井冈山大学学报(自然科学版), 2017, 38(02): 55-60.

[20] 邵必林, 吴书强, 刘江, 胡家发. 重要数据完整性分布式检测系统[J]. 探测与控制学报, 2018, 40(02): 93-98.

[21] 王晓宁. 企业网络安全应用[J]. 轻型汽车技术, 2019(Z4): 63-66+72.

[22] 赵一凡, 卞良, 丛昕. 基于MD5的加盐消息摘要Java实现[J]. 软件导刊, 2018, 17(03): 214-216+220.

[23] 任杰麟. MD5加密算法的安全性分析与改进[J]. 农业图书情报学刊, 2017, 29(07):39-42.

[24] 许琪. MD5加密算法的研究[J]. 福建电脑, 2014, 30(03): 95-96.

[25] 崔永辉, 徐鹏, 阳征鹏. 关于MD5算法的分析及其性能优化[J]. 山东工业技术, 2015(21):224.

# 致 谢

很荣幸能在哈尔滨工业大学（威海）度过大学四年，和HITER一起迎接哈工大百年校庆。我衷心祝福哈工大百年生日快乐！祝母校的未来更加美好、更加辉煌。

大学四年本人受到过无数人的帮助，正是有了你们的帮助，我不断成为更好的自己。至此，我要致谢下列组织和个人：

哈工大（威海）计算机学院的徐晓飞、王佰玲、张兆心、董开坤、刘杨、王佳浚；

哈工大（威海）海洋学院的朱永明；

网络交换与技术国家重点实验室的邱雪松、亓峰、李文璟；

哈工大（威海）计算机学院1604301班；

8公寓312寝室的穆松成、张瑞淇、赵岳阳；

以及四年来遇到的各位老师、同学，恕不在此一一列举。

其中：

朱永明老师是我大一的班主任，他为刚进入大学校门的我指明了前进的方向。王佰玲、张兆心老师、刘杨三位老师在学习上给我提供了很多指导。本课题在选题及研究过程中得到董开坤老师的悉心指导。董老师一丝不苟的作风，严谨求实的态度，踏踏实实的精神，让我对“规格严格、功夫到家”有了更深刻的理解。王佳浚作为我们的辅导员工作认真细致，感谢她这三年的辛勤付出。邱雪松、亓峰、李文璟三位老师在研究生复试阶段热心回答每一位考生咨询的问题，并为本次课题提供了很多的帮助和指导意见。感谢我的室友们的一路陪伴，我们一起不断进步。

在此特别致谢我的父母，感谢你们的养育之恩！