 

**需求分析报告**

|  |  |
| --- | --- |
| 项目名称： | 基于eBPF设计实现一个增强型网络文件系统 |
| 学生信息： | 【12012304、1120232179、马玉霞】 |
| 学生信息： | 【12012304、1120232412、唐思远】 |
| 学生信息： | 【12012304、1120230507、欧阳泽瑀】 |
| 课程名称： | 计算机软件基础I |
| 任课教师： | 翟岩龙 |

**目 录**

[1 项目介绍 1](#_Toc397327194)

[1.1 项目定义 1](#_Toc397327195)

[1.2 项目开发背景 1](#_Toc397327196)

[2 任务概述 1](#_Toc397327197)

[2.1 目标 1](#_Toc397327198)

[2.2 用户的特点 1](#_Toc397327199)

[2.3 假定和约束 2](#_Toc397327200)

[3 需求规定 2](#_Toc397327201)

[3.1 对功能的规定 2](#_Toc397327202)

[3.2 2对性能的规定 2](#_Toc397327203)

[3.3 输人输出要求 3](#_Toc397327204)

[3.4 数据管理能力要求 3](#_Toc397327205)

[3.5 故障处理要求 3](#_Toc397327206)

[3.6 保密要求 3](#_Toc397327207)

[3.7 其他专门要求 3](#_Toc397327208)

[4 运行环境规定 4](#_Toc397327209)

[4.1 设备 4](#_Toc397327210)

[4.2 支持软件 4](#_Toc397327211)

[4.3 接口 4](#_Toc397327212)

[4.4 控制 4](#_Toc397327213)

# 项目介绍

## 项目定义

本项目旨在开发一个基于eBPF（Extended Berkeley Packet Filter）技术的增强型网络文件系统（EENFS），通过在Linux内核层面进行深度优化，显著提升文件操作效率和网络通信性能。该系统将具备高性能、高安全性、高可扩展性以及灵活的动态策略调整能力，能够满足现代分布式存储环境中高并发、高吞吐量的需求。它将支持实时监控文件系统操作和网络流量，实现细粒度的访问控制，并可根据实时数据动态调整性能策略，以适应不同应用场景的多样化需求。

## 项目开发背景

随着云计算、大数据和人工智能等技术的快速发展，企业和科研机构对高性能网络存储的需求日益增长。现有的网络文件系统（如NFS、Ceph）在面对大规模分布式存储场景时，逐渐暴露出延迟高、吞吐量低以及动态策略调整能力不足等问题，难以满足现代应用对数据存储和访问的严格要求。而eBPF技术作为一种强大的内核态编程工具，能够以极低的开销实现高效的网络包处理和文件操作优化，为解决上述问题提供了新的思路。通过将eBPF技术应用于网络文件系统的设计与实现，不仅可以显著提升文件读写的效率，还能增强数据的安全性和实时监控能力，从而更好地满足现代分布式存储场景的需求。

1. **项目系统名称**：eBPF-Enhanced Network File System (EENFS)。
2. **用户**：需要高性能网络存储的企业、科研机构及云计算服务提供商
3. **项目开发动机**：

（1）现有网络文件系统（如NFS、Ceph）在延迟、吞吐量和动态策略调整方面存在瓶颈；

（2）eBPF技术可通过内核态编程实现低开销的网络包处理和文件操作优化；

（3）提升数据安全性和实时监控能力，满足现代分布式存储场景需求。

**4、系统关系**：兼容主流Linux内核（≥5.4），可与现有文件系统（如NFS）集成，提供扩展功能模块。

# 任务概述

## 目标

**1、核心目标**

利用eBPF技术在Linux内核层面实现网络文件系统的深度优化，通过减少用户态与内核态之间的切换开销，显著提升文件操作的效率和性能。同时，系统将支持实时流量分析、访问控制以及性能调优等功能，进一步提高文件读写的效率和安全性。此外，系统还将提供动态加载eBPF程序的能力，以便根据不同的应用场景需求灵活调整和优化系统性能。

**2、系统关系**

该增强型网络文件系统将作为独立模块运行于Linux内核之上，与用户态管理工具进行交互，实现对文件系统的高效管理和监控。通过eBPF Hook机制，系统将与网络协议栈和VFS（虚拟文件系统）紧密协作，实现对网络数据包和文件操作请求的实时处理和优化。此外，系统还将兼容主流Linux内核（≥5.4），并能够与现有的文件系统（如NFS）无缝集成，提供扩展功能模块，从而在充分利用现有基础设施的基础上，实现对现有系统的性能提升和功能扩展。

## 用户的特点

**1、最终用户**：

系统管理员：具备丰富的Linux内核配置和网络管理经验，能够熟练操作和维护Linux系统，对网络文件系统的性能和安全性有较高的要求。他们将负责系统的日常运维工作，包括系统配置、监控、故障排查以及性能调优等任务。因此，系统需要提供直观、易用的管理工具和详细的文档资料，以帮助系统管理员高效地完成各项工作。

开发人员：熟悉eBPF编程及文件系统原理，能够根据实际需求开发和优化eBPF程序，以实现对网络文件系统的定制化功能扩展。他们将负责系统的开发和优化工作，包括eBPF程序的编写、测试以及与内核的集成等任务。因此，系统需要提供完善的开发环境和调试工具，以支持开发人员高效地进行开发和调试工作。

**2、使用频率**：

该网络文件系统将作为企业或科研机构的核心存储系统，需要7×24小时不间断运行，以确保数据的持续可用性和业务的连续性。因此，系统必须具备高可靠性和低维护成本，能够在长时间运行过程中保持稳定性能，同时减少维护工作量和停机时间。

## 假定和约束

**1、假定**

用户环境为Linux内核版本≥5.4，这是系统能够正常运行的基本前提条件。因为eBPF技术在Linux内核5.4及以上版本中得到了较为完善的支持，能够充分发挥其性能优势和功能特性。如果用户环境的内核版本低于5.4，可能需要进行内核升级或采用其他替代方案。

硬件支持eBPF指令集，如x86\_64或ARM64架构。这是因为eBPF程序需要在硬件上高效执行，而这些架构的处理器对eBPF指令集提供了良好的支持，能够确保系统的高性能运行。如果硬件平台不支持eBPF指令集，系统将无法正常运行或性能会受到严重影响。

**2、约束**

（1）开发周期：5个月。

（2）资源限制：项目开发过程中仅允许使用开源工具链，如LLVM（≥10.0）和BCC（BPF Compiler Collection）。这意味着开发团队需要充分利用这些开源工具链的功能和特性，以实现系统的开发和优化。同时，团队还需要关注开源社区的动态，及时获取最新的工具链版本和相关技术支持，以确保项目的顺利进行。

# 需求规定

## 对功能的规定图表

| **功能模块** | **输入** | **处理** | **输出** |
| --- | --- | --- | --- |
| 实时流量监控 | 网络数据包（TCP/UDP协议，MTU=1500字节） | eBPF程序解析协议头，统计流量信息，包括但不限于数据包大小、传输速率、源/目标IP地址、端口号等，并对流量进行分类和分析，以便实时掌握网络通信的详细情况 | 流量日志（JSON格式），记录详细的流量信息，包括时间戳、数据包大小、传输速率、源/目标IP地址、端口号等，存储于/var/log/eenfs目录下，供后续分析和审计使用 |
| 动态访问控制 | 用户请求（文件路径、操作类型，如读、写、删除等） | eBPF程序根据预定义的访问控制策略，校验用户的权限，判断请求是否合法。如果请求合法，则允许操作；如果请求非法，则拦截操作并记录相关信息，同时触发安全告警 | 访问审计报告，详细记录用户的访问请求、操作类型、请求时间、是否允许操作等信息，以便对用户的访问行为进行审计和追溯 |
| 性能优化 | 文件操作请求（如文件读写请求） | eBPF程序对高频访问的数据进行缓存，减少磁盘I/O操作次数，从而提高文件读写的效率。同时，根据实时的系统负载情况，动态调整缓存策略，以确保系统在不同负载下的性能最优 | 优化后的文件操作性能，表现为文件读写延迟显著降低，吞吐量显著提高，系统整体性能得到明显提升 |
| 异常检测 | 系统调用序列 | eBPF程序实时监控系统调用序列，匹配异常模式，如非法的文件访问操作、频繁的错误操作等，并在检测到异常时触发告警，及时通知系统管理员进行处理 | 安全告警（Syslog），记录异常事件的详细信息，包括时间、操作类型、涉及的文件路径等，以便系统管理员快速定位和处理问题 |

## 对性能的规定

### 精度

网络流量统计误差率≤0.1%,这意味着系统在统计网络流量时，能够以极高的精度记录数据包的大小和传输速率等信息，确保流量统计结果的准确性和可靠性。这对于实时监控网络通信情况和进行网络性能分析至关重要。

访问控制策略匹配准确率≥99.9%,这表明系统在对用户请求进行访问控制时，能够以极高的准确率判断请求的合法性，有效防止非法访问和数据泄露等安全问题。同时，高准确率的访问控制策略匹配也有助于提高系统的性能和稳定性，避免因误判而导致的频繁拦截和告警。

### 时间特性要求

1. 响应时间

文件读写延迟≤5ms（在千兆网络环境下),这意味着系统能够在极短的时间内完成文件的读写操作，满足高并发、低延迟的应用场景需求。这对于提高系统的整体性能和用户体验至关重要，尤其是在处理大量文件操作请求时，能够显著提升系统的响应速度和效率。

1. 数据处理

eBPF程序执行时间≤1μs/请求,这表明eBPF程序在处理每个请求时能够以极高的效率执行，几乎不会对系统的性能产生明显的延迟。这对于实现系统的高性能运行和实时性要求至关重要，确保系统能够在高负载情况下仍然保持快速响应和高效处理的能力。

### 灵活性

支持动态加载/卸载eBPF程序,这意味着系统可以根据不同的应用场景需求，灵活地加载或卸载相应的eBPF程序，而无需重新启动系统或进行复杂的配置操作。这为系统的动态调整和优化提供了极大的便利，能够快速适应不同的业务需求和运行环境。

策略可基于CPU负载、网络带宽自适应调整,这表明系统能够根据实时的系统资源使用情况（如CPU负载、网络带宽等），自动调整访问控制策略、性能优化策略等，以确保系统在不同负载条件下的性能最优。这种自适应调整能力能够提高系统的智能化水平和灵活性，更好地满足动态变化的应用需求。

## 输人输出要求

### 输入

网络数据包（TCP/UDP协议，MTU=1500字节）,这是系统进行实时流量监控和网络通信优化的基础数据来源。系统需要能够高效地接收和处理这些网络数据包，以便实时掌握网络通信的详细情况，并进行相应的优化和控制操作。

用户请求（文件路径、操作类型）,这是系统进行动态访问控制和文件操作优化的重要输入信息。系统需要根据用户的请求内容，结合预定义的访问控制策略，判断请求的合法性，并进行相应的处理操作，以确保文件系统的安全性和高效性。

### 输出

日志文件（JSON格式，存储于/var/log/eenfs），这是系统运行过程中产生的各种日志信息的存储载体。日志文件以JSON格式记录了详细的流量信息、访问审计信息、安全告警信息等，便于后续的分析和审计工作。通过存储日志文件，系统管理员可以全面了解系统的运行状态和用户行为，及时发现和处理潜在的问题。

实时监控仪表盘（通过Prometheus+Grafana展示）,这是系统为用户提供的一种直观、实时的监控界面。通过集成Prometheus和Grafana等监控工具，系统能够将实时的流量数据、性能指标、访问控制信息等以图形化的方式展示在监控仪表盘上，使用户能够直观地了解系统的运行状态和性能表现，及时发现异常情况并进行处理。

## 数据管理能力要求

最大支持10,000个并发连接，系统能够同时处理大量的用户连接请求，满足高并发场景下的数据访问需求。这对于提高系统的吞吐量和性能至关重要，能够确保系统在面对大量用户同时访问时仍然保持稳定运行。

单节点文件存储规模≥1PB，系统具备强大的文件存储能力，能够满足大规模数据存储的需求。这对于处理海量文件和数据的应用场景非常关键，能够确保系统在长时间运行过程中不会因存储空间不足而影响业务的正常运行。

日志保留周期≥30天，系统能够保留较长时间的日志文件，以便用户进行历史数据的查询和分析。这对于系统的安全审计和故障排查等工作非常有帮助，能够为用户提供足够的数据支持，以便全面了解系统的运行历史和用户行为。

## 故障处理要求

1. **硬件故障**

支持RAID冗余存储，自动切换备份节点，系统采用了RAID技术对存储设备进行冗余配置，当发生硬件故障时，能够自动切换到备份节点，确保数据的完整性和可用性。这种冗余机制能够有效提高系统的可靠性，减少硬件故障对系统运行的影响。

1. **软件故障**

eBPF程序崩溃后，内核自动恢复并重启，系统具备一定的容错能力，当eBPF程序出现异常崩溃时，内核能够自动检测并进行恢复操作，重新启动eBPF程序，确保系统的正常运行。这种自动恢复机制能够提高系统的稳定性和可用性，减少软件故障对系统的影响。

1. **网络中断**

提供断点续传和一致性校验机制，系统在网络中断的情况下，能够自动记录中断时的数据传输状态，并在网络恢复后继续传输未完成的数据，同时进行一致性校验，确保数据的完整性和准确性。这种机制能够有效提高系统的容错能力和数据可靠性，减少网络中断对数据传输的影响。

## 保密要求

数据传输使用TLS加密，这确保了数据在网络传输过程中的安全性，防止数据被窃取或篡改。通过采用TLS加密协议，系统能够对传输的数据进行加密处理，只有合法的接收方才能解密并获取数据内容，从而有效保护数据的机密性和完整性。

用户权限遵循RBAC（基于角色的访问控制），根据用户的角色和职责分配相应的权限。用户只能访问与其角色相关的文件和数据，从而有效防止未经授权的访问和数据泄露。这种访问控制机制能够提高系统的安全性，确保数据的访问行为符合组织的安全策略。

## 其他专门要求

1. **可维护性**

提供详细的API文档和调试工具，这有助于开发人员和系统管理员更好地理解和使用系统的功能接口，快速定位和解决系统运行过程中出现的问题。详细的API文档能够为开发人员提供清晰的开发指南，调试工具则能够帮助他们高效地进行问题排查和修复，从而提高系统的可维护性和开发效率。

1. **可扩展性**

支持插件式扩展eBPF功能模块，系统采用了模块化的设计思想，允许用户根据实际需求灵活地扩展eBPF功能模块。通过插件式扩展机制，用户可以方便地添加新的功能特性，而无需对系统的核心代码进行修改，从而提高了系统的可扩展性和灵活性，能够更好地适应未来的发展需求。

1. **兼容性**

适配主流通用文件系统（如Ext4、XFS），确保了系统能够与现有的文件系统无缝集成，充分利用现有的基础设施和数据存储资源。通过兼容主流通用文件系统，系统能够更好地融入现有的IT环境，降低用户的使用成本和迁移难度，提高系统的实用性和市场竞争力。

# 运行环境规定

## 设备

**1）处理器**：x86\_64或ARM64架构，≥4核，系统需要具备高性能的处理器，可以满足高并发、高吞吐量的计算需求。多核处理器能够同时处理多个任务，提高系统的整体性能和效率。

**2）内存：**≥8GB，这确保了系统有足够的内存空间来存储和处理大量的数据，同时支持多个并发任务的运行。充足的内存能够提高系统的响应速度和性能表现，减少因内存不足而导致的性能瓶颈。

**3）存储**：SSD/NVMe硬盘，≥1TB可用空间；使用固态硬盘（SSD）或非易失性内存快速存储（NVMe）硬盘，以提高数据的读写速度和存储性能。同时，系统能够具备足够的存储空间来满足大规模数据存储的需求。

**4）网络**：北京理工大学校园网。

## 支持软件

**1）操作系统**：Linux内核≥5.4，这是因为eBPF技术在Linux内核5.4及以上版本中得到了较好的支持，能够充分发挥其性能优势和功能特性。

**2）工具链**：LLVM≥10.0，BCC（BPF Compiler Collection），系统需要使用LLVM（≥10.0）和BCC作为开发工具链，用于编译和加载eBPF程序。这些工具链提供了强大的编译和调试功能，能够支持开发人员高效地开发和优化eBPF程序。

**3）监控工具**：Prometheus, Grafana，用于实时监控系统的运行状态和性能指标，并通过图形化的方式展示监控数据。这些监控工具能够为用户提供直观的监控界面，帮助用户及时发现和处理系统运行过程中出现的问题。

## 接口

1. **内核接口**

通过eBPF Hook与网络协议栈、VFS交互，系统通过eBPF Hook机制与Linux内核的网络协议栈和虚拟文件系统（VFS）进行深度交互，实现对网络数据包和文件操作请求的实时处理和优化。这种内核接口机制能够充分发挥eBPF技术的优势，提高系统的性能和效率。

1. **用户态接口**

REST API（管理策略及监控数据），系统提供了REST API接口，用于用户态管理工具与内核态系统之间的交互。通过REST API，用户可以方便地管理访问控制策略、监控数据等信息，实现对系统的动态管理和优化。

## 控制

1. **控制方式**

通过用户态管理工具（CLI/Web UI）动态加载eBPF程序；允许用户通过用户态管理工具（如命令行界面CLI或Web用户界面Web UI）动态加载和卸载eBPF程序，实现对系统的灵活配置和优化。这种控制方式为用户提供了便捷的操作手段，能够快速适应不同的应用场景需求。

1. **信号来源**

管理员指令或自动化脚本触发策略更新，系统支持通过管理员指令或自动化脚本触发访问控制策略和性能优化策略的更新。管理员可以根据实际需求手动调整策略，也可以通过编写自动化脚本实现策略的自动更新，提高系统的智能化水平和管理效率。