毕业设计过程记录册（2025年4月6日更新）

一、上周计划完成情况（2025.3.30-2025.4.5）

1. 核心功能模块开发

* 实现基于eBPF的动态链接事件捕获框架
* 通过dlopen/dlclose的uprobe挂载，成功捕获库加载/卸载事件
* 完成事件数据结构设计（包含时间戳、PID/TID、库路径、加载地址等）
* 实现内核态与用户态的数据传输（通过BPF\_PERF\_OUTPUT机制）

开发关键辅助功能模块 ：

* 库路径解析（支持绝对路径/相对路径自动补全）
* 动态库依赖分析（集成ldd命令实现依赖树解析）
* 时间戳纳秒级精度转换（基于std::chrono实现）
* 符号地址与库文件映射关系维护（全局loaded\_libraries映射表）

2. 测试验证

* 完成基础场景验证
* 成功捕获libc/libm等系统库加载事件
* 验证动态加载场景（如Python扩展模块加载）
* 测试多进程并发加载时的数据隔离性

1. 本周计划（2025.4.6-2025.4.12）

路径解析增强：

* 优化get\_full\_library\_path()函数，添加LD\_LIBRARY\_PATH环境变量搜索支持
* 增加缓存机制避免重复路径解析（使用static std::map缓存结果）

事件处理改进：

* 在handle\_event()中添加事件去重机制（基于PID+路径+时间戳哈希）
* 优化特殊标记处理（如"[主程序]","[返回句柄]"）的逻辑清晰度

依赖分析优化：

* 重写get\_library\_dependencies()改用libelf直接解析而非调用ldd
* 添加依赖树缓存避免重复解析相同库

关键路径优化

* 用std::string\_view替代部分字符串拷贝
* 将频繁调用的format\_timestamp()改为无锁实现

内存管理

* 为loaded\_libraries映射表添加LRU清理机制
* 预分配事件处理所需内存避免动态分配