内核模块演示文档

V0.1

北京凝思软件股份有限公司

文档修订记录

VER	REV	修订人	修订日期	简要说明	批准人	批准日期
V0.1	C	谢永伟	2025-06-12	初次创建		
VER（版本编号）：V——版本编号；R——修订编号。						
REV（修订状态）：C——创建；A——增加；M——修改；D——删除。						

目录

表格

插图

1 简介

本文档描述了一个基于 Linux 内核模块的系统调用监控实现方案。该模块利用内核探测机制，能够实时捕获系统调用的进入和退出事件，记录相关信息，并通过用户空间接口进行配置和管理。旨在提供一个高效、灵活且相对通用的系统调用观测平台，提升系统的安全性。

此章节包含概述、功能介绍章节。

1.1 概述

此内核模块为 Falco 内核模块的简化版，保留了核心功能，便于理解和演示。本文档阐述了该内核模块的设计与实现的细节，已实现的功能，以及待完善的问题。该模块通过在系统调用的入口和出口处设置探测点，捕获系统中发生的系统调用事件，并通过 ring buffer 机制将信息高效的传输到用户空间程序进行分析和处理。同时可通过 ioctl 接口实现动态配置功能。

1.2 功能介绍

此内核模块的主要功能为捕获系统调用的相关信息，并将其传递给用户空间进行处理或分析。具体来说，实现了如下功能：

1. 系统调用事件捕获：在 syscall_enter 和 syscall_exit 两个关键点设置探测，捕获完整的系统调用生命周期。
2. 可配置监控目标：通过 ioctl 接口动态配置需要监控的系统调用。
3. 高效数据传输：采用 mmap 双重映射和 ring buffer 机制实现内核与用户空间的高效数据交换。
4. 基本信息采集：包括操作用户、系统调用号、时间戳、操作命令、PID、TID、PPID 等关键信息。

2 成果演示

用户空间命令行界面以及监控日志输出示例：

```
[2025-06-12 14:15:02.492380090]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.492416140]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.492425117]: read syscall=0 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.sh
pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.492557191]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.492564992]: read syscall=0 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.sh
pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.493229891]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.493298136]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.493371317]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.493823371]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
[2025-06-12 14:15:02.493835609]: openat syscall=257 dir=> res=0 user=root comm=test.sh cmdline=/bin/bash ./test.
sh  pid=3391 tid=3391 ppid=2524(bash)
```

插图 1: 内核模块日志输出

3 整体架构

此章节主要描述此系统调用监控模块的设计框架以及实现功能的相关技术，包含总体框架、关键技术章节。

3.1 总体框架

此系统调用监控模块包含内核层与应用层。内核层完成内核模块初始化，系统调用信息过滤、收集及封装，以及系统调用的配置管理。应用层完成命令解析、系统调用信息读取以及输出到日志文件中。

系统调用初始化完成探测点注册，字符设备注册功能。探测点当前仅支持系统调用入口点和退出点。字符设备用于配置管理和缓冲区映射，文件操作函数包括 open、release、ioctl 及 mmap。open()接口完成消费者注册功能，release()用于释放资源，ioctl()用于配置管理，包含系统配置，探测点配置等。mmap()用于映射缓冲区，完成内核与应用层信息交互。

系统调用信息的收集是在探测点收集进程信息并封装成事件，写入缓冲区。应用层则直接从缓冲区读取信息，输出到终端或日志文件，避免多次数据拷贝。

整体框架如下:

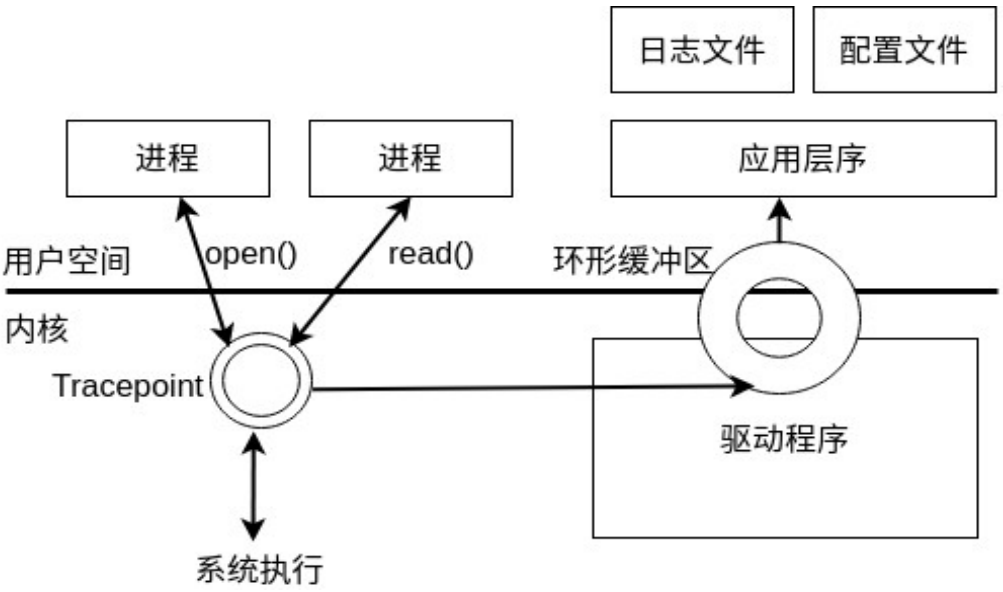

插图 2: 内核模块总体框架

3.2 关键技术

1. **探测机制**：采用 Linux 内核 Tracepoint 机制。

   Tracepoint 机制是 Linux 内核中一种轻量级的代码插桩技术，核心原理是在内核关键路径上预置静态钩子函数。相比与传统 kprobe 动态插桩相比，Tracepoint 为 Linux 内核静态定义探测点，开销小，类型安全以及多消费者支持等特点。

2. **双重映射**：使用 mmap 与 ring buffer 机制。

   内存映射（mmap）系统调用允许将文件或设备内存映射到进程地址空间。对于驱动程序，内核分配的内存将映射到用户空间。环形缓冲区（ring buffer）是固定大小的循环缓冲区，存储事件记录。双重映射是将统一物理内存同时映射到内核空间和用户空间不同的虚拟地址，实现零拷贝数据传输，同时解决环形缓冲区回绕问题。

3. **事件格式**：事件记录采用紧凑的二进制格式。

   进程信息的记录使用紧凑二进制格式，即将收集的信息，按预定义事件结构体的形式，以二进制流写入缓冲区，应用层在输出成易于识别的信息。应用输出信息输出格式如下：

   ```
   |时间戳|系统调用名称 |系统调用号|返回值|操作用户|进程名|操作命令|PID|TID|PPID(父进程名)|
   ```

4. **配置管理**：通过 ioctl 命令实现动态配置。

   应用层可通过 ioctl()接口配置具体系统调用是否采集。内核则通过位图标记系统调用是否采集。

4 配置选项

应用层命名支持指定日志文件输出路径及名称，支持测试日志接口测试功能。

内核模块可配置监控特定的系统调用，当前操作命令包含系统调用使能，以及统调用失能。详细如下：

表格 1: 配置参数

序号	配置参数	描述
1	CMD_ENABLE_SYSCALL	系统调用使能
2	CMD_DISABLE_SYSCALL	系统调用失能

5 现存问题

当前系统调用收集模块完成部分核心功能，但仍存在待完善和有化的地方，如多核支持、系统覆盖不全等问题，具体描述如下：

1. 仅处理 open、readat、unlink 等系统调用探测点仅支持系统调用入口及退出点。

表格 2: Falco 探测点

序号	探测点	描述	当前支持情况
1	sys_enter	系统调用入口	支持
2	sys_exit	系统调用出口	支持
3	sched_process_exit	进程中止	不支持
4	sched_switch	进程切换	不支持
5	page_fault_user	用户空间发生页面错误	不支持
6	page_fault_kernel	内核空间发生页面错误	不支持
7	signal_deliver	信号传递到用户空间	不支持
8	sched_process_fork	创建子进程	不支持
9	sched_process_exec	系统调用替换当前进程	不支持

2. 不支持动态加载的系统调用监控。
3. 不支持多线程、多核同步、多消费者场景

4. 不支持丢弃模式配置以及动态调整输出，采集等配置。