CSCC 全国大学生计算机系统能力大赛

2025年全国大学生计算机系统能力大赛
-操作系统设计赛(全国)-OS功能挑战赛道

基于eBPF的容器异常检测工具Agent

答辩人：刘周康 毕喜舒 马永媛

国防科技大学 NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY 1953

指导教师：任怡 赵欣

CONTENTS

项目背景与目标

为什么需要新的容器监控方案？

我们要解决什么问题？

绪论背景：为什么需要新的容器监控方案？

容器环境挑战

- 高动态性：容器生命周期短（平均 < 3 小时），传统静态监控失效
- 强隔离性：namespace/cgroup 隔离导致用户态工具难以穿透容器边界
- 大规模性：微服务集群复杂，跨容器调用拓扑难追踪

传统监控的痛点

- 开销高：用户态工具上下文切换频繁，性能损耗大
- 粒度粗：仅能获取容器级聚合指标，缺乏进程级细粒度行为
- 响应慢：异常检测延迟高，难以捕捉瞬时异常

eBPF 技术的机遇

内核级可编程能力，低侵入、高实时、细粒度监控

绪论背景：我们要解决什么问题？

内核级监控

实时捕获容器内进程、系统调用、文件、网络等行为

智能异常检测

融合规则、统计、机器学习模型识别安全与性能异常

轻量化部署

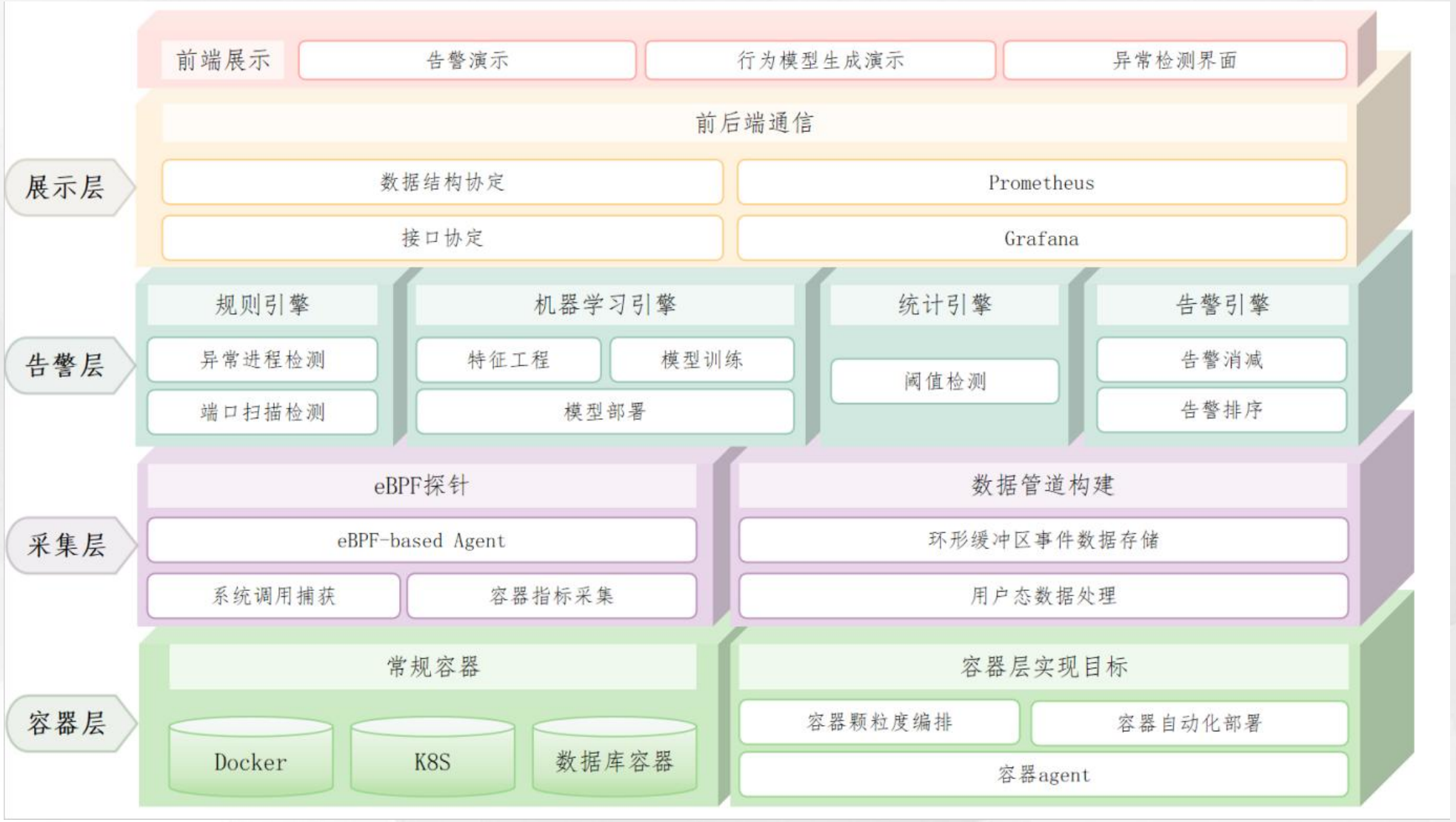
适配高密度容器场景，资源开销极低

云原生兼容

支持 Docker/K8s，提供可视化与自动化运维能力

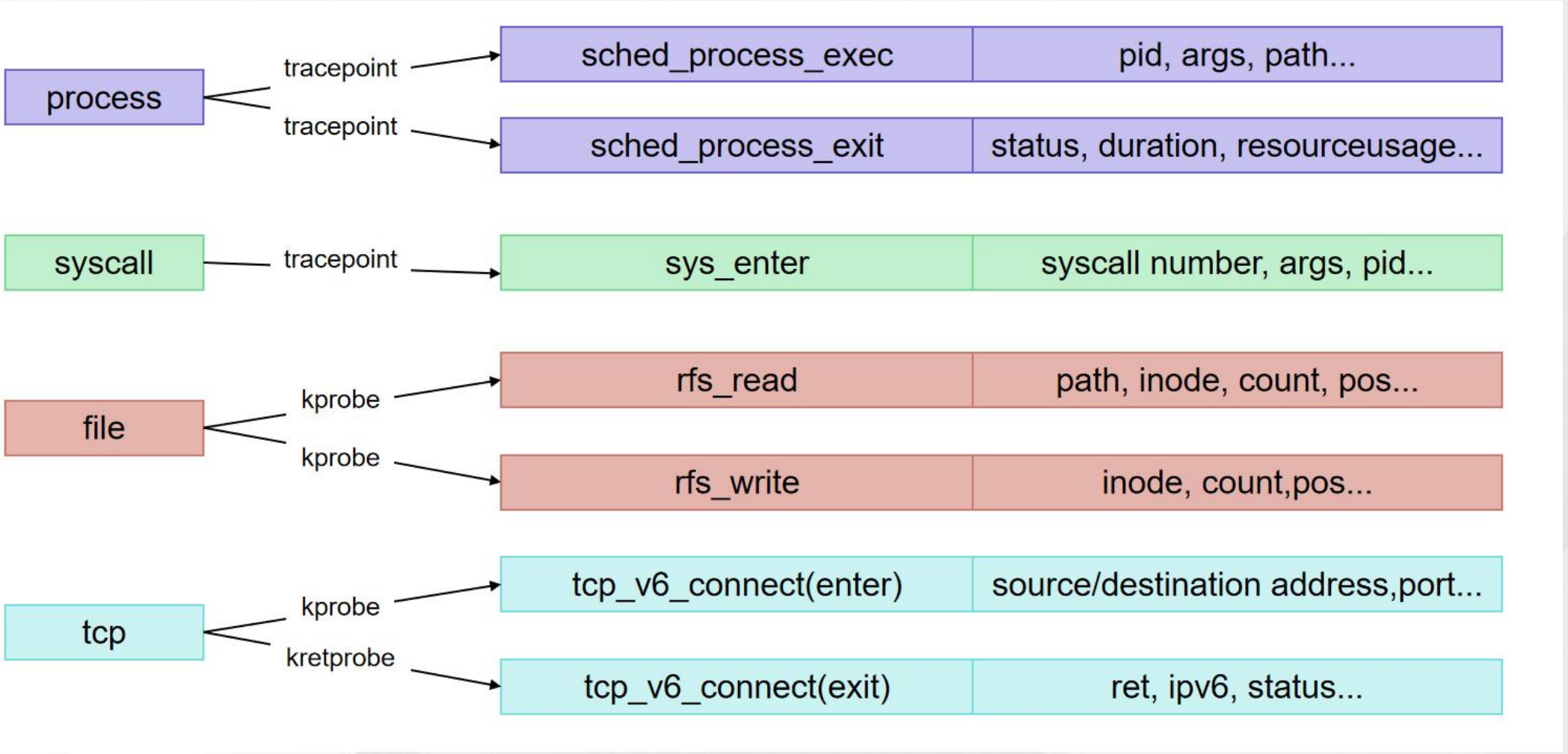
核心技术方案

核心技术方案：整体架构



内核态事件捕获→用户态特征提取→多维度异常分析→可视化告警

核心技术方案：eBPF 探针设计

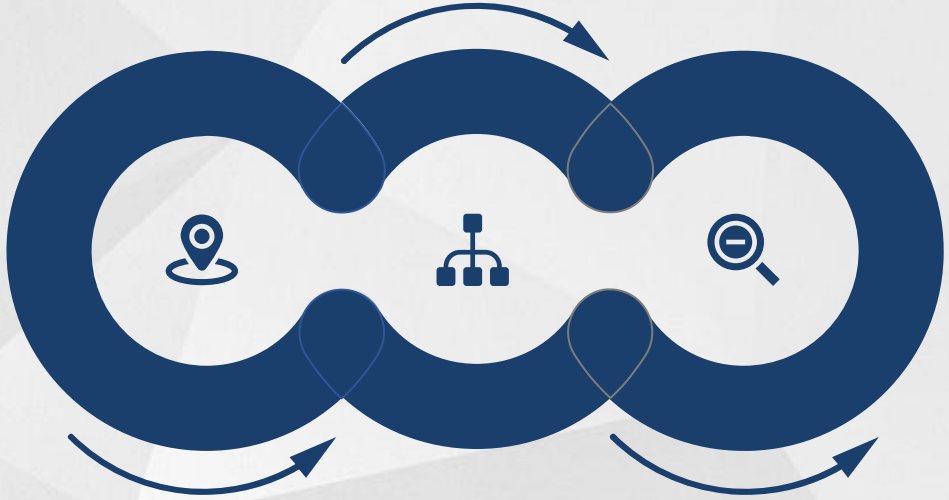


tracepoint 优先保障稳定性　kprobe 补充覆盖细分场景

核心技术方案：容器元信息关联

动态更新

基于进程事件（exec/exit）实时更新，利用 namespace/cgroup 特征推导容器归属



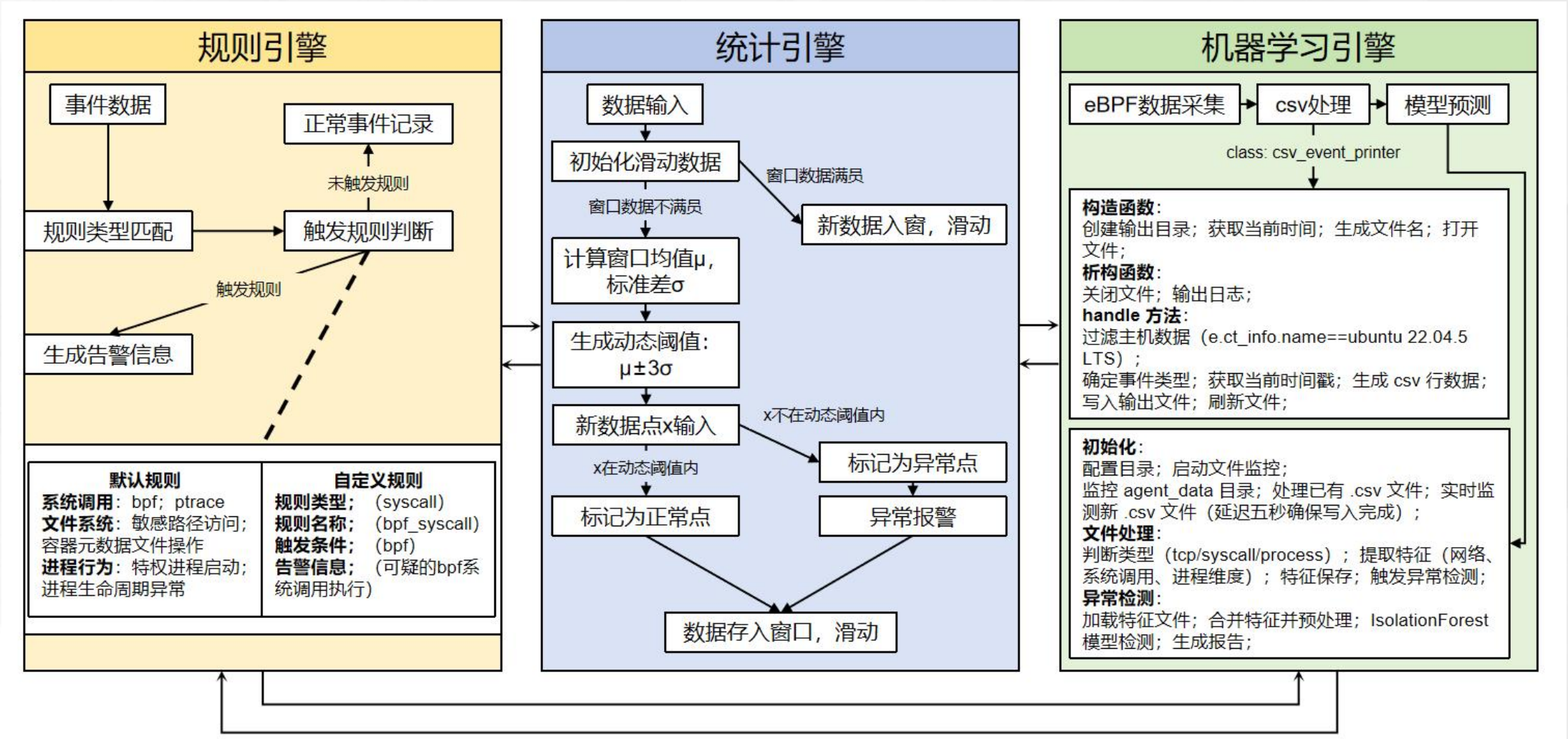
初始化

通过 Docker API 批量获取容器与进程映射

毫秒级关联

父进程 namespace 继承关系 + 异步刷新机制，确保映射延迟 < 10ms

核心技术方案：多维度异常检测模型



规则响应快、统计抗噪声、机器学习适配动态场景

关键创新点

关键创新点：内核级动态观测技术

突破传统监控局限，基于 eBPF 实现内核级细粒度观测

动态挂载 **tracepoint/kprobe**，覆盖进程 / 文件 / 网络全维度行为

内核态数据过滤：预先筛选无效事件，减少 **70%+** 用户态数据传输

跨内核版本兼容：基于 **libbpf** 与 **BTF** 实现 “一次编译，多机运行”

数据采集延迟从秒级降至毫秒级，性能损耗降低 **40%+**

关键创新点：多维度异常检测融合

规则引擎即时响应已知威胁，机器学习挖掘未知异常

Isolation Forest 适配无标签容器环境，支持动态更新基线

告警延迟≤1 秒，准确率达 92%

兼顾检测速度、准确率与自适应能力，覆盖复杂容器场景

关键创新点：轻量化可扩展架构

核心 Agent 二进制仅 4MB，CPU 占用≤5%，内存≤120MB

“探针 - 处理器” 责任链模式，支持动态加载 / 卸载探针

与 Prometheus/Grafana 无缝集成，提供 HTTP API 扩展

适配高密度容器部署，支持用户自定义监控规则

实现效果

实现效果：性能测试

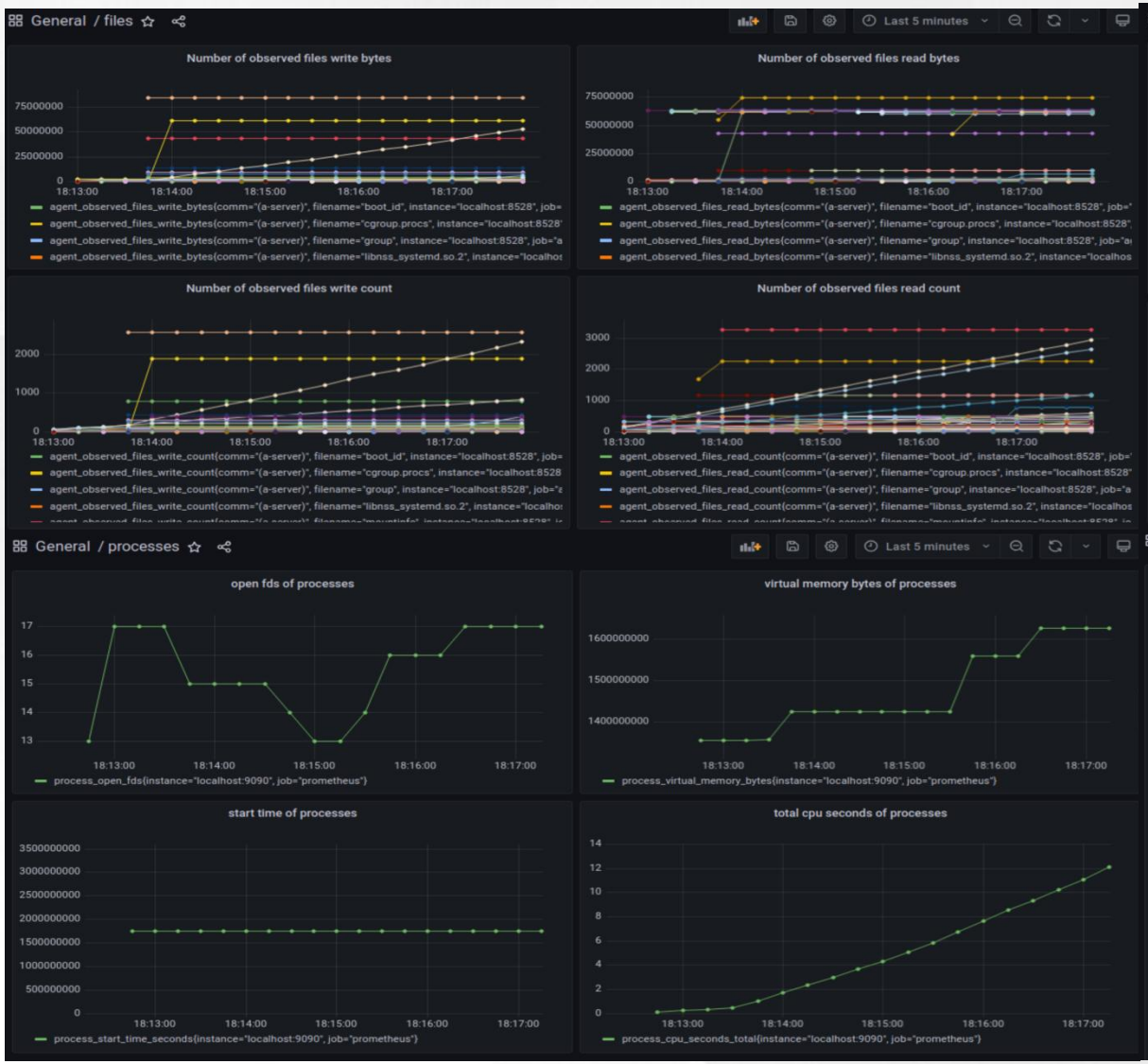
测试环境：Linux 5.10 内核，4 核 CPU+4GB 内存，Docker/K8s 集群

```
%Cpu(s): 12.5 us,  3.6 sy,  0.0 ni, 83.7 id,  0.1 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem :   3868.5 total,    292.5 free,   2024.4 used,   1551.6 buff/cache
MiB Swap:   3898.0 total,   3715.6 free,    182.4 used.   1502.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   3919 liuzhou+  20   0   32.4g 164500 142916 S  54.3   4.2   0:56.61 code
   3887 liuzhou+  20   0 1163.2g 320884 149160 S   3.3   8.1   0:53.82 code
   4001 liuzhou+  20   0 1161.5g 108600  73344 S   1.3   2.7   0:05.46 code
   5215 liuzhou+  20   0   10892   4224   3200 R   1.3   0.1   0:01.28 top
    190 root      20   0       0      0      0 I   0.7   0.0   0:01.37 kworker/1:2-events
    715 root      20   0  246748   9344   8064 S   0.7   0.2   0:01.87 vmtoolsd
   3772 liuzhou+  20   0 1161.6g 154152 118028 S   0.7   3.9   0:19.36 code
   3956 liuzhou+  20   0 1161.5g 104524  72064 S   0.7   2.6   0:06.05 code
     37 root      20   0       0      0      0 I   0.3   0.0   0:00.67 kworker/3:0-events
    115 root      20   0       0      0      0 I   0.3   0.0   0:01.27 kworker/2:2-events
    284 root      20   0       0      0      0 I   0.3   0.0   0:01.30 kworker/u257:26-events_unbound
    377 root      20   0       0      0      0 I   0.3   0.0   0:01.57 kworker/0:3-mm_percpu_wq
    673 systemd+  20   0   14836   6272   6016 S   0.3   0.2   0:01.49 systemd-oomd
    990 root      20   0 1874996  29796  15616 S   0.3   0.8   0:02.32 containerd
   2667 liuzhou+  20   0  143416  26156  17964 S   0.3   0.7   0:02.06 vmtoolsd
   3858 liuzhou+  20   0   32.3g  61224  58848 S   0.3   1.5   0:00.86 code
   3938 liuzhou+  20   0 1163.1g 431536  83596 S   0.3  10.9   0:42.10 code
      1 root      20   0  166948  11912   8200 S   0.0   0.3   0:03.35 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.03 kthreadd
      3 root      20   0       0      0      0 S   0.0   0.0   0:00.00 pool_workqueue_release
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-rcu_g
```

```
suan@suan-virtual-machine:~/Desktop/11/ebpf$ ./wrk -t16 -c320 -d30s http://localhost:9090/
  Running 30s test @ http://localhost:9090/
    16 threads and 320 connections
    Thread Stats   Avg      Stdev     Max   +/- Stdev
      Latency     6.95ms    5.35ms  94.38ms   79.71%
      Req/Sec     3.14k   535.42     6.65k    72.85%
    1501379 requests in 30.04s, 237.68MB read
  Requests/sec:  49980.55
  Transfer/sec:      7.91MB
suan@suan-virtual-machine:~/Desktop/11/ebpf$ ./wrk -t16 -c320 -d30s http://localhost:9090/
Running 30s test @ http://localhost:9090/
  16 threads and 320 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     6.72ms    4.86ms  60.22ms   75.55%
    Req/Sec     3.22k   375.38     8.11k    71.02%
  1541117 requests in 30.05s, 243.97MB read
Requests/sec:  51277.80
Transfer/sec:      8.12MB
```

实现效果：功能验证


General / files
Last 5 minutes
Number of observed files write bytes
Number of observed files read bytes
Number of observed files write count
Number of observed files read count


General / syscalls
Last 5 minutes
stack bytes of syscalls
heap bytes of syscalls
other bytes of syscalls
total bytes of syscalls

General / processes
Last 5 minutes
open fds of processes
virtual memory bytes of processes
start time of processes
total cpu seconds of processes


General / tcp connect
Last 5 minutes
Number of observed tcp v4 connect count
Number of observed tcp v6 connect count

总结与未来展望

项目成果

构建了完整的基于 eBPF 的内核级容器监控与异常检测体系

实现轻量化部署（4MB 二进制），性能损耗低（≤2%）

支持 Docker/K8s，提供实时监控、异常告警与可视化分析

解决容器动态性、隔离性带来的监控难题，为云原生环境提供安全与性能保障

未来展望

01

算法升级

融合 Transformer 模型挖掘时序依赖，提升缓慢异常识别能力

02

观测扩展

新增内核内存监控、中间件专用探针，覆盖微服务全链路

03

性能优化

动态调度探针采样频率，支撑万级容器规模化监控

04

生态融合

深化与云原生工具链集成，构建“监控 - 调度 - 自愈”闭环

CSCC 全国大学生计算机系统能力大赛

2025年全国大学生计算机系统能力大赛
-操作系统设计赛(全国)-OS功能挑战赛道

请各位老师指正

答辩人：刘周康 毕喜舒 马永媛

指导教师：任怡 赵欣