在OpenBMC上实现固件可观测性技术

郑春辉 字节跳动固件架构师

王志强 浪潮信息固件工程师

背景

传统BMC开发交付周期长。越来越难胜任多样化的应用场景。

OpenBMC的优势：开源支持、架构灵活、兼容性强、安全和稳定。

OpenBMC在产业中有越来越广的应用趋势。几乎所有的厂商都有应用OpenBMC的案例。

OpenBMC在实际项目中使用有迭代快、响应需求迅速等优点

问题简述

问题难以复现

- 问题复现概率很低
- 触发条件不明确
- 时序问题比如并发时序很难复现

复杂配置

- 线上环境配置复杂
- 实验室环境难以遍历所有配置

调试受限

- 线上环境对调试器使用严格受限
- 编译器优化导致抓不到某些内部变量
- 调试器本身对问题复现有影响

当前方案局限

- 当前BMC可观测方案严重依赖日志
- 非结构数据分析困难
- 过多日志时性能开销大
- 数据量不足
- 缺乏全局上下文

解决方案

出现问题时的内部状态（软件和硬件状态）不足以定位故障，所以才需要后续复现。

问题

如果能事先 充分记录内部状态和运行足迹，就可以避免问题复现和调试器介入。

分析

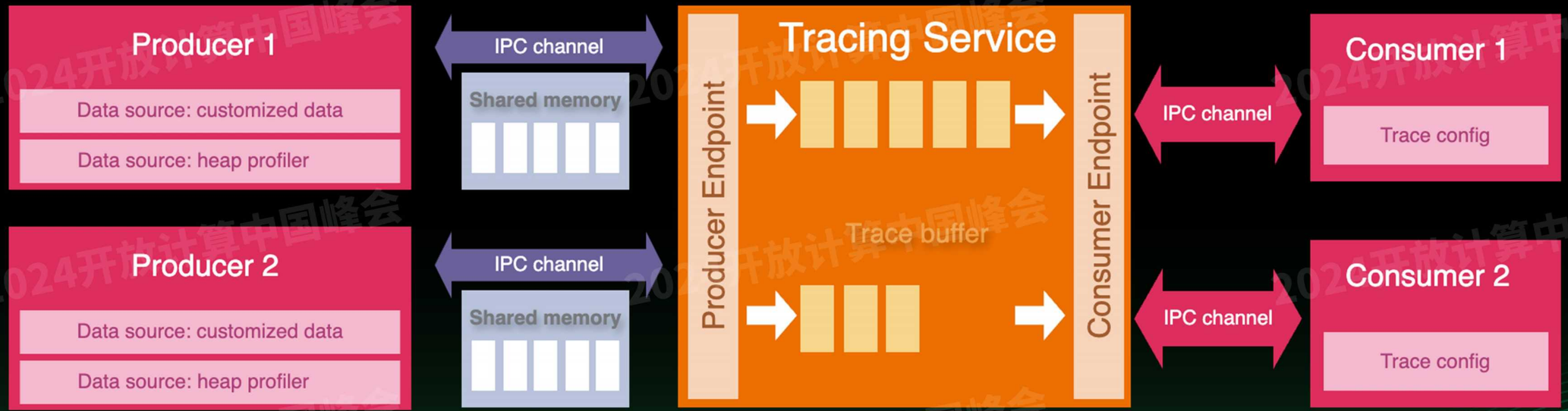
引入对BMC影响小的可观测技术，把内部状态记录到存储，就可以在数据级别重现故障现场。

方法

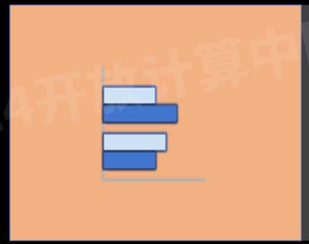
技术介绍

Perfetto

Perfetto原理



Perfetto优点



低开销

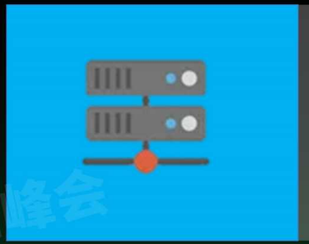
采用protobuf和string interning技术，开销达到最小。

灵活的数据采集

支持多种数据源，包括内核追踪（ftrace）、自定义数据等。可以根据需求灵活配置。

强大的数据分析能力

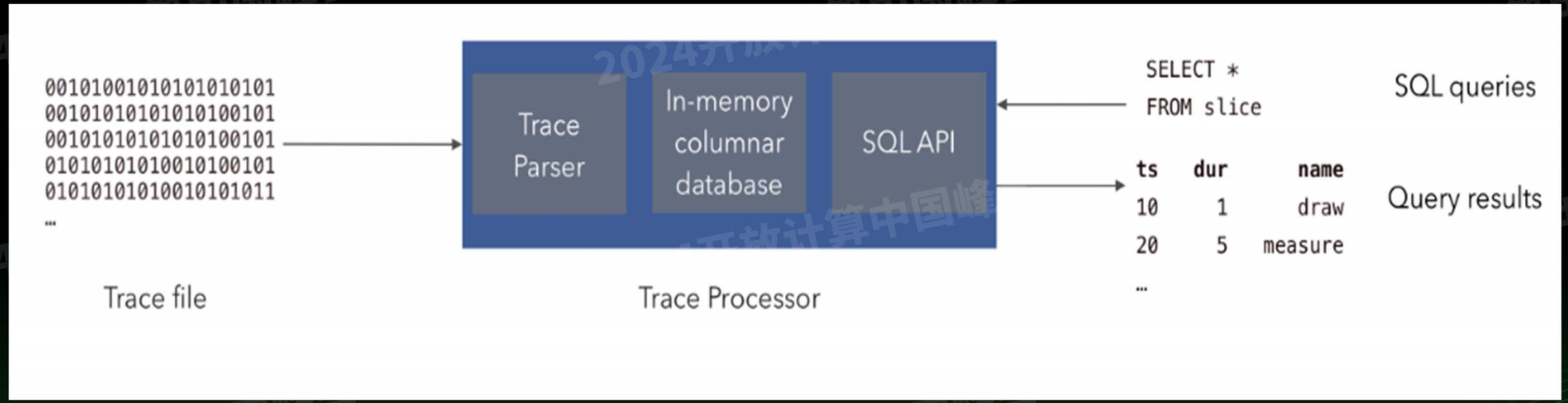
丰富的数据可视化和分析工具，可以通过图形界面查看和分析追踪数据，识别性能瓶颈



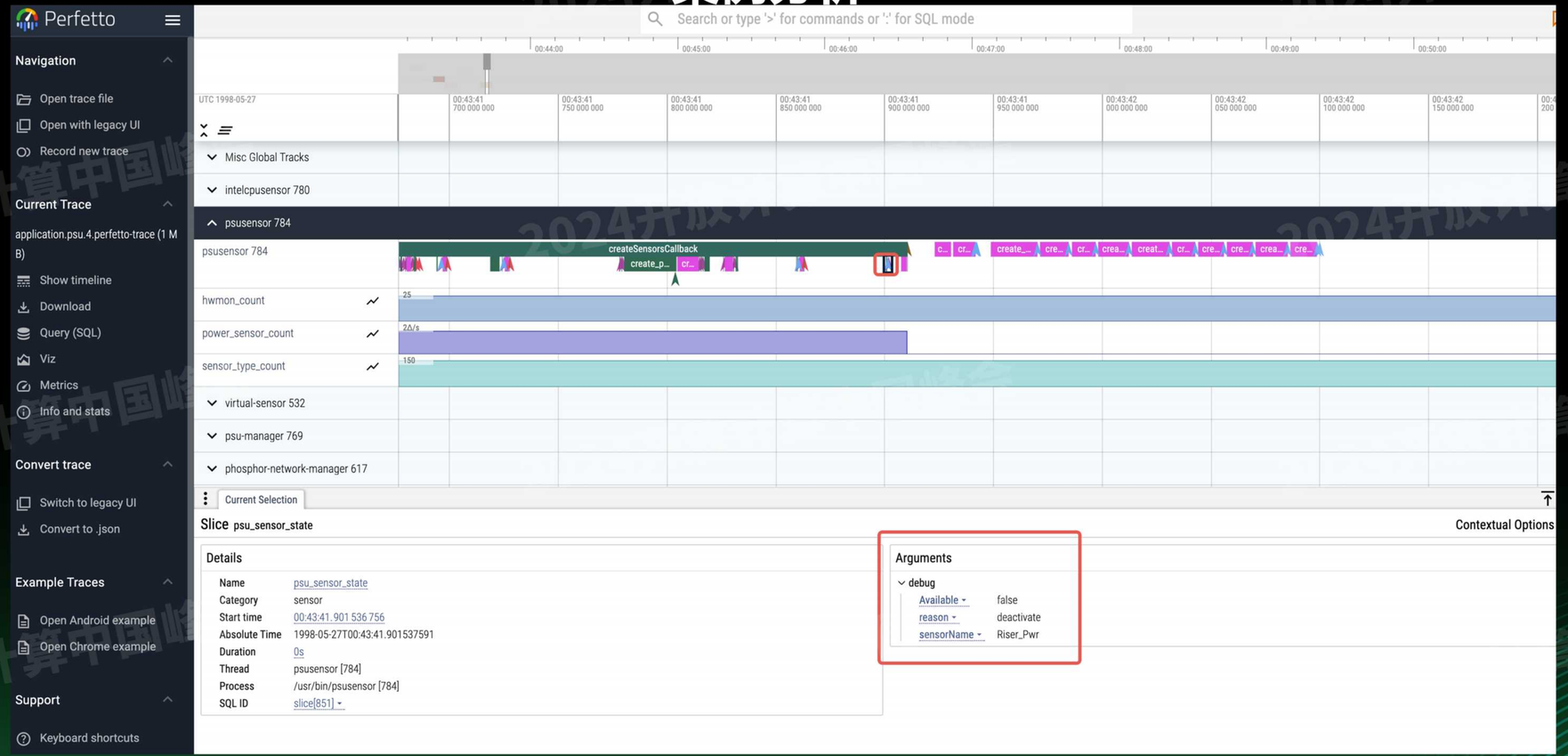
高精度追踪

捕捉系统级别的高精度事件，包括 CPU 调度、I/O 操作等，提供纳秒级别的时间戳

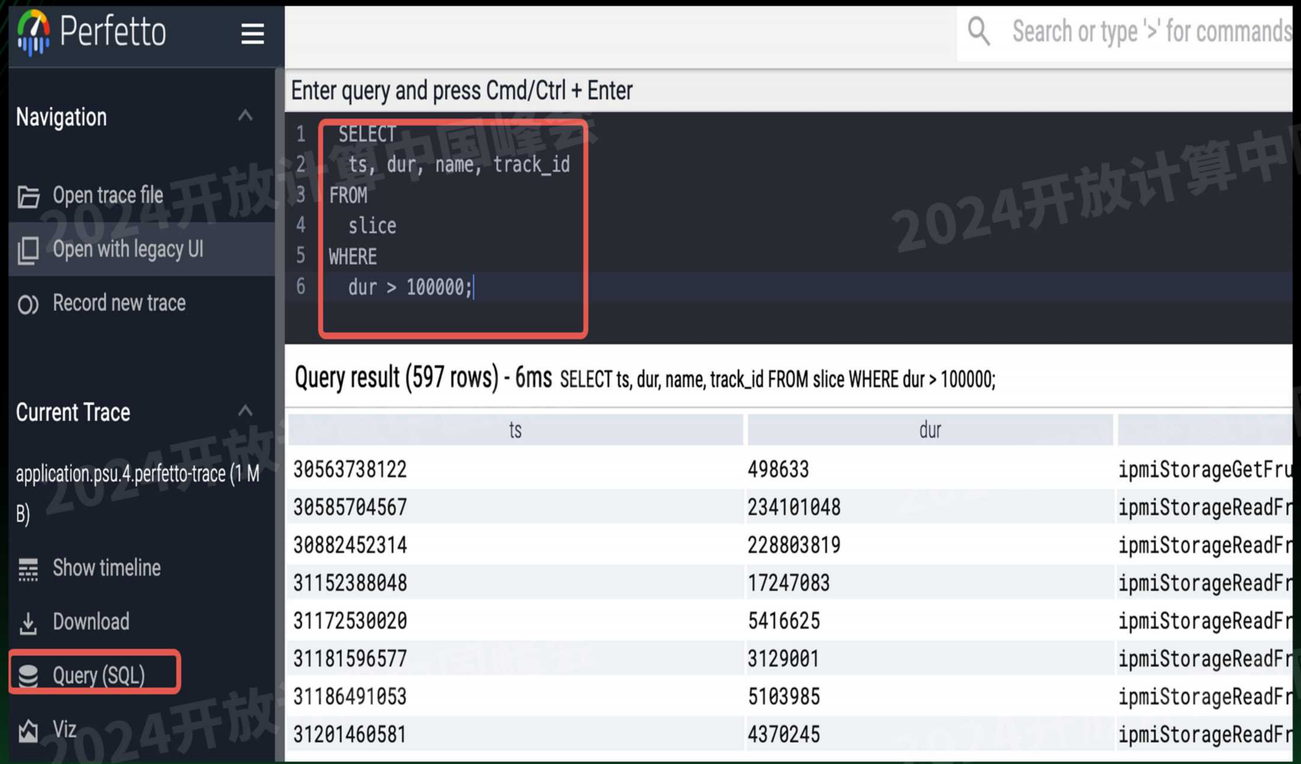
Perfetto原理

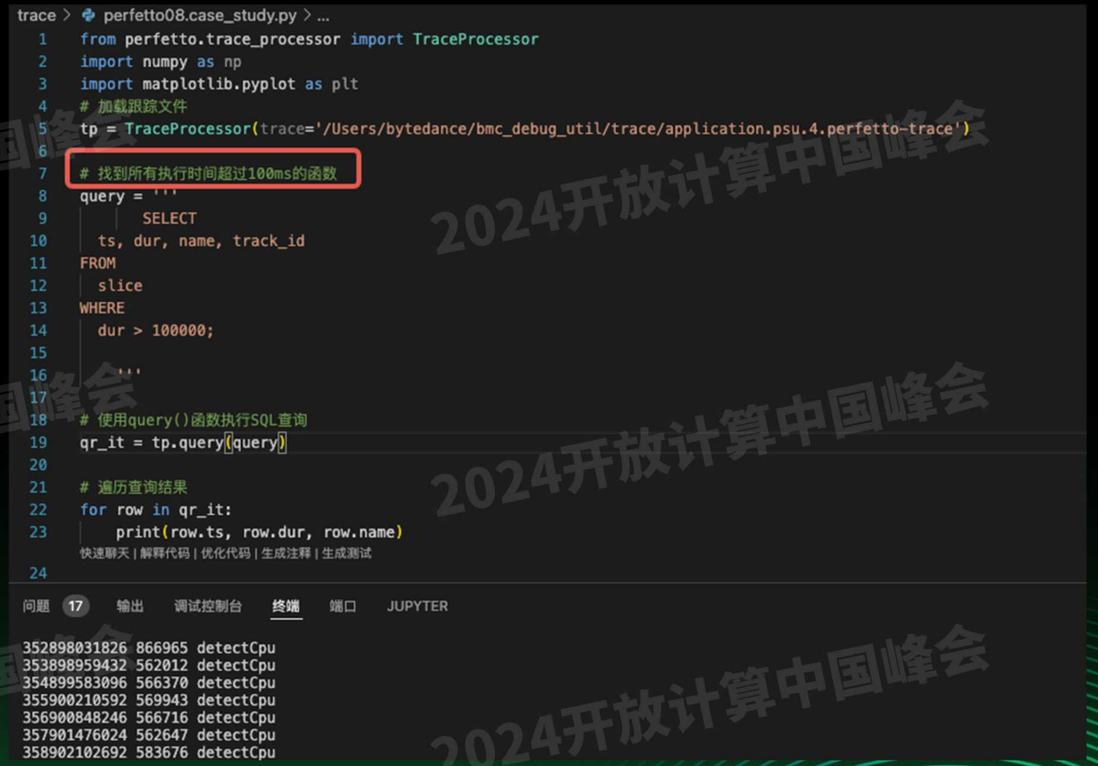

0010100101010101010101
0010101010101010100101
0010101010101010100101
0101010101001010100101
0101010101001010101011
…
Trace file
Trace Parser
In-memory columnar database
SQL API
Trace Processor
SELECT *
FROM slice
SQL queries
ts dur name
10 1 draw
20 5 measure
…
Query results

案例分析


Perfetto
Search or type '>' for commands or ':' for SQL mode
Navigation
Open trace file
Open with legacy UI
Record new trace
Current Trace
application.psu.4.perfetto-trace (1 MB)
Show timeline
Download
Query (SQL)
Viz
Metrics
Info and stats
Convert trace
Switch to legacy UI
Convert to .json
Example Traces
Open Android example
Open Chrome example
Support
Keyboard shortcuts
Misc Global Tracks
intelcpusensor 780
psusensor 784
psusensor 784
createSensorsCallback
hwmon_count
power_sensor_count
sensor_type_count
virtual-sensor 532
psu-manager 769
phosphor-network-manager 617
Current Selection
Slice psu_sensor_state
Contextual Options
Details
Name psu_sensor_state
Category sensor
Start time 00:43:41.901 536 756
Absolute Time 1998-05-27T00:43:41.901537591
Duration 0s
Thread psusensor [784]
Process /usr/bin/psusensor [784]
SQL ID slice[851]
Arguments
debug
Available false
reason deactivate
sensorName Riser_Pwr

案例分析


Perfetto
Search or type '>' for commands
Navigation
Open trace file
Open with legacy UI
Record new trace
Current Trace
application.psu.4.perfetto-trace (1 MB)
Show timeline
Download
Query (SQL)
Viz
Enter query and press Cmd/Ctrl + Enter
SELECT
ts, dur, name, track_id
FROM
slice
WHERE
dur > 100000;
Query result (597 rows) - 6ms SELECT ts, dur, name, track_id FROM slice WHERE dur > 100000;
ts
dur
30563738122 498633 ipmiStorageGetFru
30585704567 234101048 ipmiStorageReadFr
30882452314 228803819 ipmiStorageReadFr
31152388048 17247083 ipmiStorageReadFr
31172530020 5416625 ipmiStorageReadFr
31181596577 3129001 ipmiStorageReadFr
31186491053 5103985 ipmiStorageReadFr
31201460581 4370245 ipmiStorageReadFr


trace > perfetto08.case_study.py > ...
from perfetto.trace_processor import TraceProcessor
import numpy as np
import matplotlib.pyplot as plt
# 加载跟踪文件
tp = TraceProcessor(trace='/Users/bytedance/bmc_debug_util/trace/application.psu.4.perfetto-trace')
# 找到所有执行时间超过100ms的函数
query = '''
SELECT
ts, dur, name, track_id
FROM
slice
WHERE
dur > 100000;
'''
# 使用query()函数执行SQL查询
qr_it = tp.query(query)
# 遍历查询结果
for row in qr_it:
print(row.ts, row.dur, row.name)
快速聊天 | 解释代码 | 优化代码 | 生成注释 | 生成测试
问题 17 输出 调试控制台 终端 端口 JUPYTER
352898031826 866965 detectCpu
353898959432 562012 detectCpu
354899583096 566370 detectCpu
355900210592 569943 detectCpu
356900848246 566716 detectCpu
357901476024 562647 detectCpu
358902102692 583676 detectCpu

Kernel trace

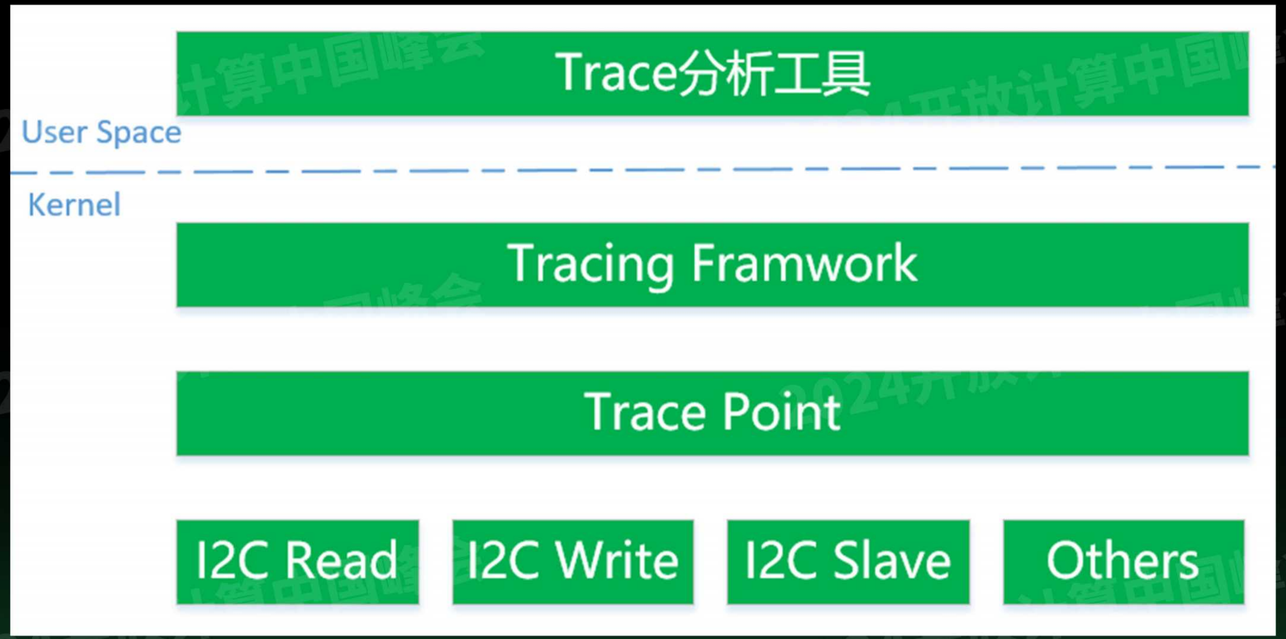
问题背景

如何快速，精准定位硬件故障

300台机器，运行7天7夜，中间发现有5台机器的

- PSU无法获取信息，且当前也无法获取
- PSU偶尔有1-5分钟无法获取信息，当前正常
- PSU温度超温，且当前也是高温
- PSU偶尔有1-5分钟超温，当前温度正常。

Kernel Trace 简介

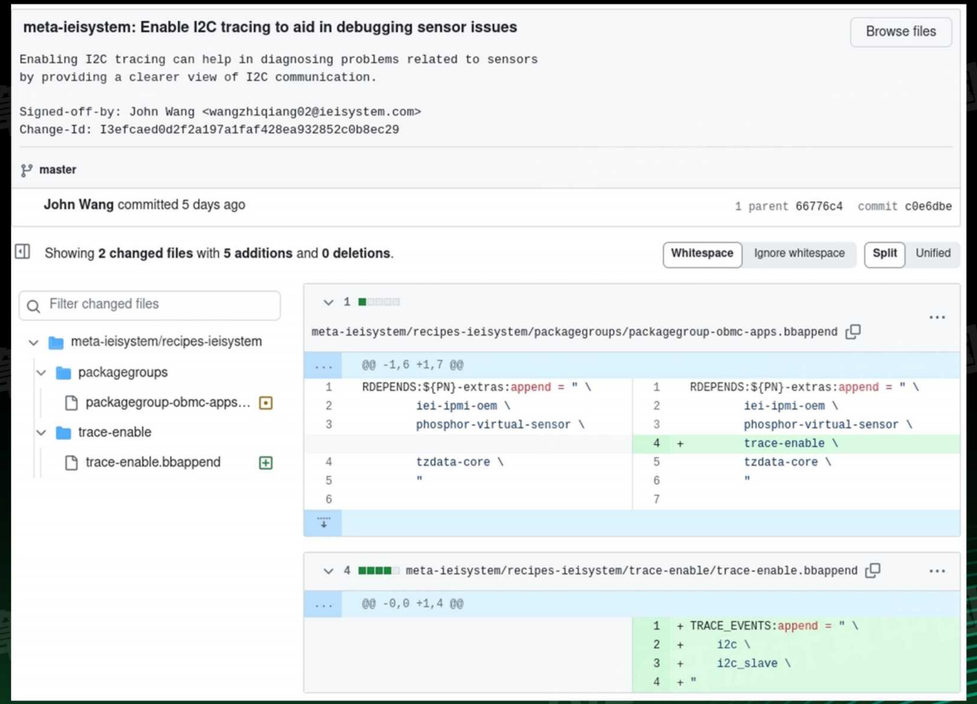



```
i2c_write: i2c-11 #0 a=062 f=0004 l=3 [00-01-21]
i2c_result: i2c-11 n=1 ret=1
i2c_write: i2c-11 #0 a=070 f=0000 l=1 [00]
i2c_result: i2c-11 n=1 ret=1
i2c_result: i2c-95 n=1 ret=1
i2c_write: i2c-95 #0 a=062 f=0004 l=1 [00]
i2c_read: i2c-95 #1 a=062 f=0005 l=2
i2c_write: i2c-11 #0 a=070 f=0000 l=1 [20]
i2c_result: i2c-11 n=1 ret=1
i2c_write: i2c-11 #0 a=062 f=0004 l=1 [00]
i2c_read: i2c-11 #1 a=062 f=0005 l=2
i2c_reply: i2c-11 #1 a=062 f=0005 l=2 [01-59]
i2c_result: i2c-11 n=2 ret=2
i2c_write: i2c-11 #0 a=070 f=0000 l=1 [00]
i2c_result: i2c-11 n=1 ret=1
i2c_reply: i2c-95 #1 a=062 f=0005 l=2 [01-59]
i2c_result: i2c-95 n=2 ret=2
i2c_write: i2c-95 #0 a=062 f=0004 l=1 [79]
i2c_read: i2c-95 #1 a=062 f=0005 l=3
i2c_write: i2c-11 #0 a=070 f=0000 l=1 [20]
i2c_result: i2c-11 n=1 ret=1
i2c_write: i2c-11 #0 a=062 f=0004 l=1 [79]
```

Kernel Trace 日志收集

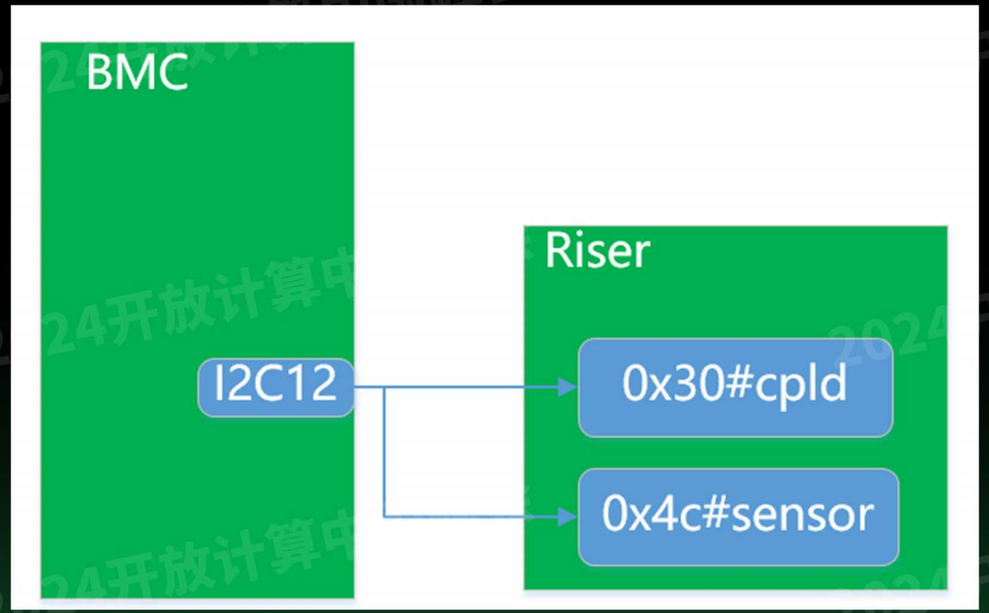
Kernel Trace使能

- 使能trace-enable模块并配置
- 采用一键日志的方案收集



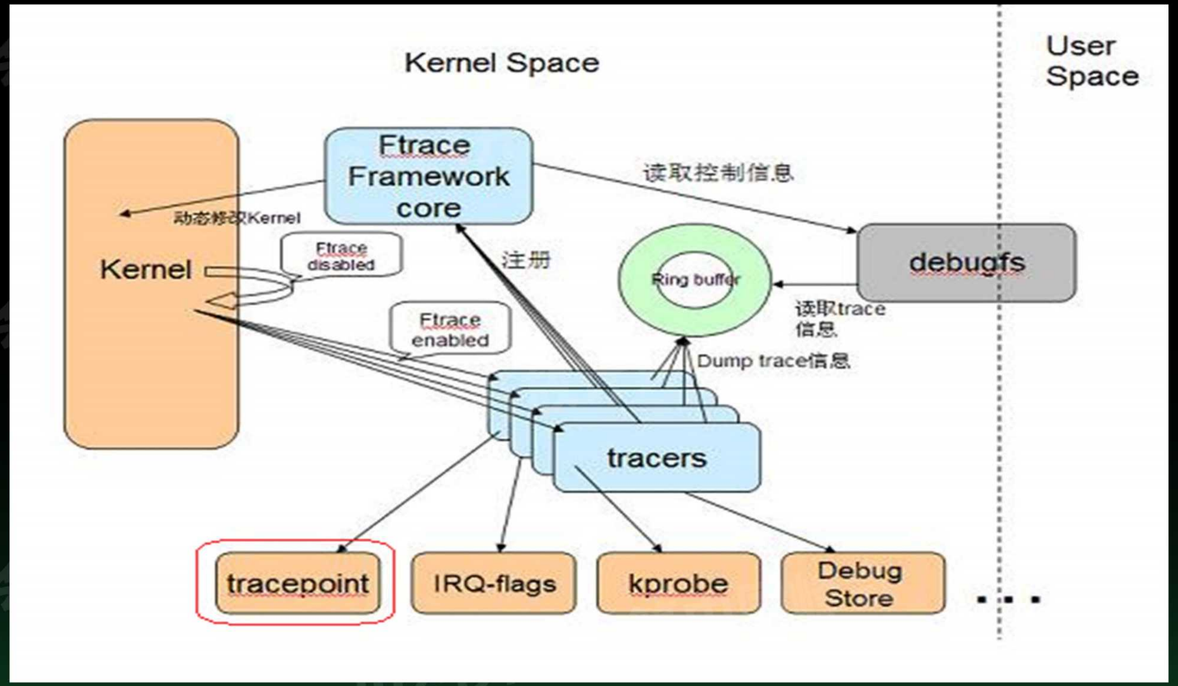
Kernel Trace 案例分析

问题现象: 某板卡温度sensor无法获取




```
43.994037: i2c_read: i2c-12 #1 a=030 f=0201 l=4
43.994751: i2c_reply: i2c-12 #1 a=030 f=0201 l=4 [03-ff-ff-ff]
43.994754: i2c_result: i2c-12 n=2 ret=2
43.995334: i2c_write: i2c-12 #0 a=030 f=0000 l=1 [15]
43.995340: i2c_read: i2c-12 #1 a=030 f=0201 l=3
43.997351: i2c_reply: i2c-12 #1 a=030 f=0201 l=3 [00-00-01]
43.997356: i2c_result: i2c-12 n=2 ret=2
44.007550: i2c_write: i2c-12 #0 a=04c f=0000 l=1 [00]
44.007553: i2c_read: i2c-12 #1 a=04c f=0001 l=1
44.007708: i2c_result: i2c-12 n=2 ret=-6
44.007738: i2c_write: i2c-12 #0 a=04c f=0000 l=1 [01]
44.007740: i2c_read: i2c-12 #1 a=04c f=0001 l=1
44.007877: i2c_result: i2c-12 n=2 ret=-6
44.007896: i2c_write: i2c-12 #0 a=04c f=0000 l=1 [23]
44.007898: i2c_read: i2c-12 #1 a=04c f=0001 l=1
44.008032: i2c_result: i2c-12 n=2 ret=-6
44.019870: i2c_write: i2c-12 #0 a=030 f=0000 l=1 [20]
44.019875: i2c_read: i2c-12 #1 a=030 f=0201 l=6
44.033659: i2c_reply: i2c-12 #1 a=030 f=0201 l=6 [05-6c-01-39-01-54]
44.033667: i2c_result: i2c-12 n=2 ret=2
44.033823: i2c_write: i2c-12 #0 a=030 f=0000 l=1 [01]
44.033827: i2c_read: i2c-12 #1 a=030 f=0201 l=4
44.034616: i2c_reply: i2c-12 #1 a=030 f=0201 l=4 [03-ff-ff-ff]
44.034623: i2c_result: i2c-12 n=2 ret=2
```

Kernel Trace 原理介绍



总结

Perfetto	可以有效监控BMC软件状态 方便快速精准定位BMC软件模块	问题定位分析的平均时间：5小时->1小时 一次定位分析成功率：40% -> 70% 典型项目中定位问题的数量：30+
Kernel trace	可以有效监控BMC和硬件的沟通 方便快速精准定位硬件故障	问题定位分析的平均时间：8小时->1小时 一次定位分析成功率：40% -> 80% 典型项目中定位问题的数量：40+

未来展望

引入对 Perfetto 和 Kernel Trace的数据分析

1. 引入大数据分析，分析大规模部署条件下的故障特征
2. 细化各种场景下问题定位
3. 在故障特征基础上引入自动问题定位过程

1.关注我们的[illegible]，获取最新的技术动态和成功案例。

THANKS