嵌入式系统安全性的最新技术挑战

名古屋大学
李奕骁

个人简介

- 工作单位
  - 日本名古屋大学情报学（Informatics）研究科 助理教授
    - ERTL（嵌入式实时系统研究室）
      - 侧重于学生培养、学术研究活动
    - NCES（附属嵌入式系统研究中心）
      - 侧重于企业合作，共同进行研究开发

- 开源活动
  - TOPPERS Project

- 今天的报告均来自ERTL、NCES及TOPPERS相关的研究课题

关于TOPPERS Project

TOPPERS
Toyohashi OPen Platform for Embedded Real-time Systems

- 由名古屋大学高田广章教授（NCES所长）发起的项目
- 活动目标
  - 提供嵌入式系统所需要的高品质开源内核及相关规范文档、组件
  - 成果在日本学界及产业市场都有广泛采用，且有商业公司提供功能扩展和技术支援等服务
- 2003年成立NPO（非盈利组织）法人
  - 现有162位会员：其中72家企业、14家团体
- 主要成果例
  - TOPPERS第3代实时内核系列（uITRON-like）
  - TOPPERS/ATK2（AUTOSAR CP 开源内核）及各种中间件
  - TOPPERS/SafeG系列嵌入式虚拟机
  - TOPPERS/EV3RT 乐高EV3机器人开发平台
  - Hakoniwa IoT机器人仿真环境
    - https://toppers.github.io/hakoniwa/en/

高田广章 教授

报告内容

- 软件脆弱性对策
- 虚拟化技术
- 实时Linux

软件脆弱性对策

嵌入式软件开发模式的变化趋势

- 形式化验证尚存瓶颈
  - 目前主要适用于内核、驱动、Hypervisor等核心组件
  - 较难满足上层应用开发对迭代效率和成本控制的需求
- 软件脆弱性风险激增
  - 功能复杂度爆发，潜在漏洞无法准确评估
  - 联网设备成为主流，漏洞利用越来越容易

	经典方法	最新趋势
功能定位	硬件产品的附属	软件定义产品
开发规模	整车约千万行	可超数亿行
升级频率	慢（甚至为零）	OTA持续迭代
组件构成	大多为商用组件	积极采用开源组件
代码质量	充分测试或验证	难以得到保证

软件开发的各阶段都需要脆弱性对策

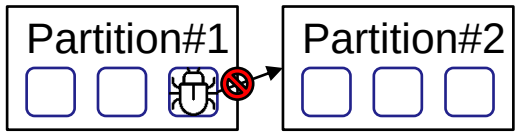
- 设计时：安全风险等级划分
  - 例如车载领域的ASIL分解
- 测试时：提高漏洞发现效率
  - 从手工测试到持续、自动化测试
- 运行时：潜在漏洞风险缓解
  - 结合硬件保护机制降低危险严重性

操作系统相关技术
需提供必要支援！

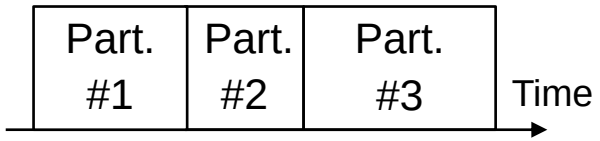
软件脆弱性对策：分区隔离

分区隔离已成为重要的标准技术

- 空间隔离
  将内核对象及内存等资源按功能和安全等级分区
  通过内存保护和权限限制等防止漏洞干涉其他分区
- 时间隔离
  对各分区分配的独立时间窗口用以执行
  可以保证各分区的时间确定性互不干涉



空间隔离
E.g. AUTOSAR CP SC3



时间隔离
E.g. ARINC 653 调度

TOPPERS/ATK2

- AUTOSAR CP开源内核
- 实现SC2、SC3、SC4扩展
- 提供RTE、CAN、WDG等相关组件
- 支持RH850、多核处理器

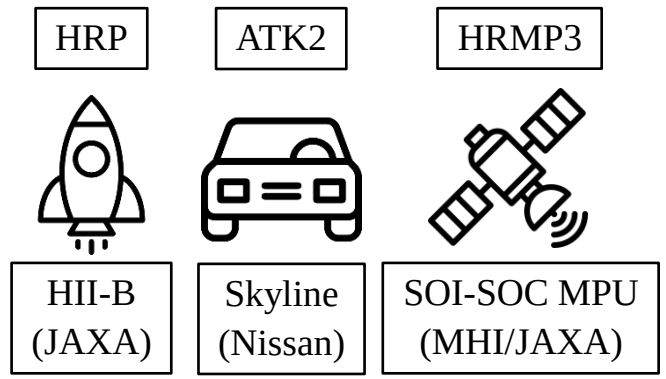
※ https://www.toppers.jp/en/atk2.html

TOPPERS/HRMP3

- uITRON-like开源内核
  发展自TOPPERS/HRP高信赖内核
- 面向多核处理器
- 实现空间隔离和时间隔离
- 满足航空需求的微秒级时钟管理

※ https://www.toppers.jp/hrmp3-kernel.html

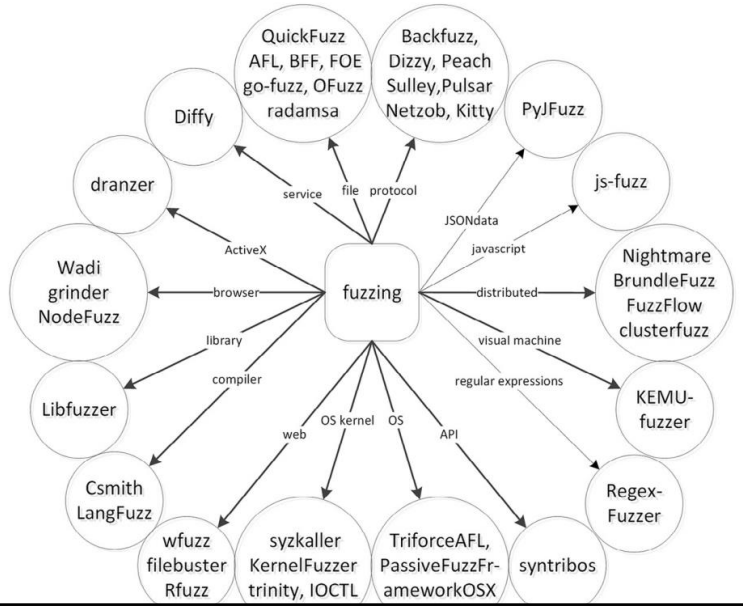
TOPPERS内核产业应用例



软件脆弱性对策：模糊测试

模糊测试是高效发现漏洞的关键技术

- 手工测试的局限性日益显著
  复杂的系统开发中，路径覆盖率低、维护成本高
- 早期漏洞发现需要海量测试用例
  模糊方法无需人工干预，可持续自动生成测试
  已在安全性要求高的云计算领域广泛采用



Fuzzer相关研究多集中在Linux服务器领域※

嵌入式系统模糊测试的发展空间巨大

- 测试工具需支持多种架构和操作系统
  架构：Cortex-M、TriCore、RH、RISC-V等
  操作系统：AUTOSAR规范、POSIX规范、TOPPERS规范等
- 测试对象应涵盖底层资源
  内核服务、并发/中断、硬件驱动等
  还应尽可能对外部设备和物理环境进行仿真
- 近期值得关注的研究成果

SFuzz: Slice-based Fuzzing for Real-Time Operating Systems
基于Unicorn多架构CPU模拟引擎的RTOS模糊测试工具
https://doi.org/10.1145/3548606.3559367

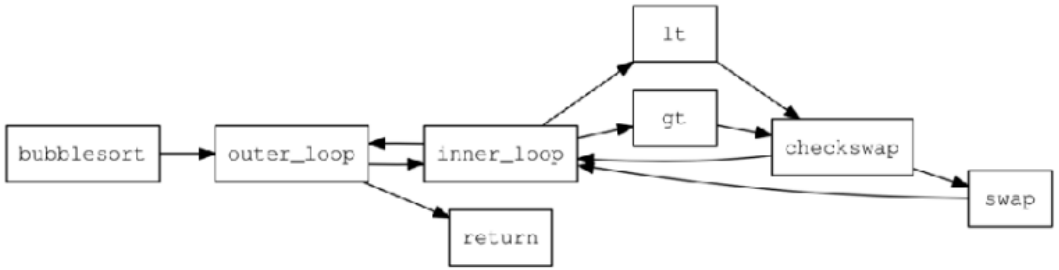
μAFL: Non-intrusive Feedback-driven Fuzzing for Microcontroller Firmware
基于Hardware-in-the-Loop仿真的嵌入式固件模糊测试工具
https://doi.org/10.1145/3510003.3510208

※ https://doi.org/10.1016/j.cose.2018.02.002

软件脆弱性对策：控制流完整性

控制流攻击能对安全造成极大危害

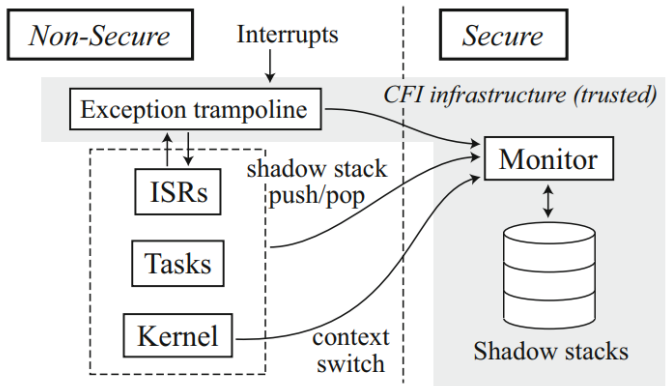
- 可绕开空间隔离的执行保护机制实现任意代码执行
  ROP/JOP代码复用攻击一般都可达到图灵完备
- 控制流完整性（Control Flow Integrity）防御机制
  确保编译时程序的控制流不会在执行中被篡改
  可大幅缓解潜在漏洞被利用时的安全风险



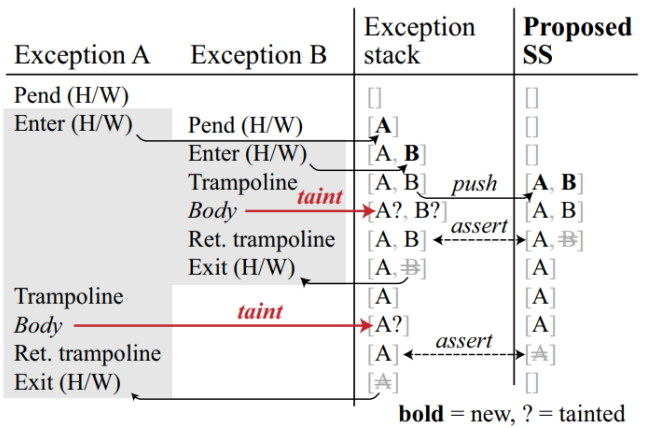
CFI对函数间合法的迁移路径进行保护

TZmCFI：面向ARMv8-M的系统级CFI技术

※ https://doi.org/10.1007/s10766-020-00673-z



利用TrustZone硬件隔离控制流信息



覆盖多任务及中断嵌套，完整保护RTOS

最近，处理器厂商也开始追加硬件CFI相关技术！

Intel：Control-flow Enforcement Technology

ARM：Pointer Authentication Code

虚拟化技术

嵌入式产品功能多样化

- 虚拟化的多系统共存架构可提高安全性和开发效率
  E.g. 实时控制RTOS+通信用IoT OS
  E.g. 安全RTOS+非安全Android
- 主流Hypervisor对嵌入式支持有限
  资源开销大，无法适用于内存较少的系统
  实现复杂，实时性和可预测性较难保证

主流开源Hypervisor示例

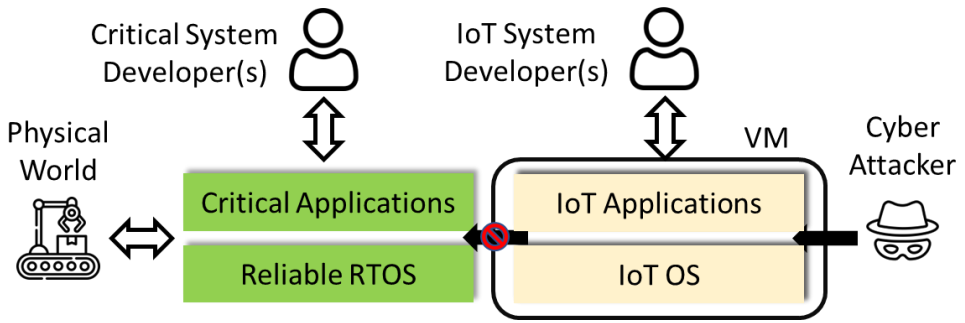
资源受限MCU也需要开源虚拟化技术

- 低端：单核心、无硬件虚拟化技术
  Cortex-M4F、Renesas RX等
- 中端：单核心、有硬件虚拟化技术
  Cortex-M33、Cortex-M55等
- 高端：多核心、有硬件虚拟化技术
  RH850/U2A、R-Car S4 (G4MH) 等

虚拟化技术：中低端MCU

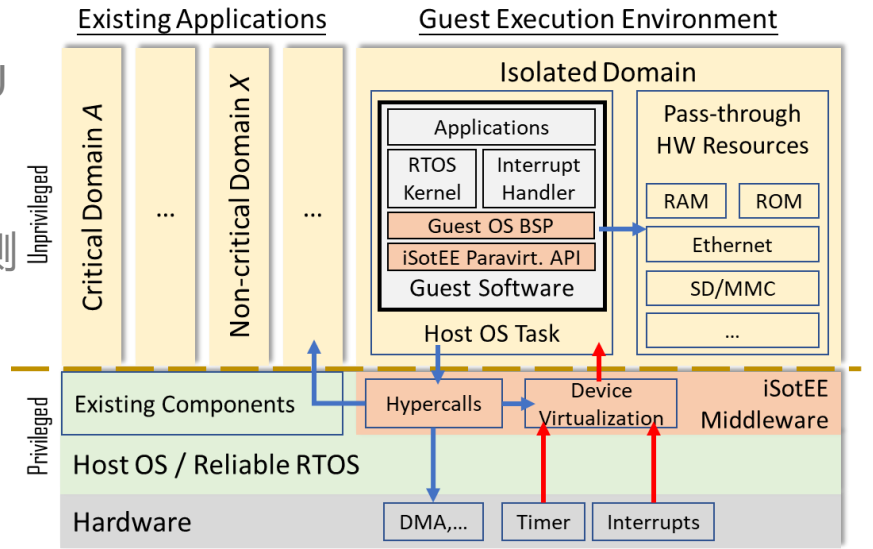
很多物联网嵌入式系统使用中低端MCU

- 中间件丰富的IoT OS开发容易但易遭受外部攻击
- 采用虚拟化技术构建Dual OS架构优点明显
  - IoT应用和OS服务隔离在非可信虚拟机内执行
  - 高可信的RTOS负责物理控制，不受外部攻击影响



iSotEE Hypervisor Middleware

- 不依赖需硬件虚拟化技术
  - 可支持STM32、RX65N等MCU
  - 准虚拟化：CPU、中断、时钟
- RTOS任务原生执行保证实时性
  - Hypervisor性能开销完全可预测
- IoT OS和应用都在同一个Host OS Task上执行
  - 类似Unikernel、提升处理性能



※ https://doi.org/10.1109/ACCESS.2022.3144044

TOPPERS/SafeG-M

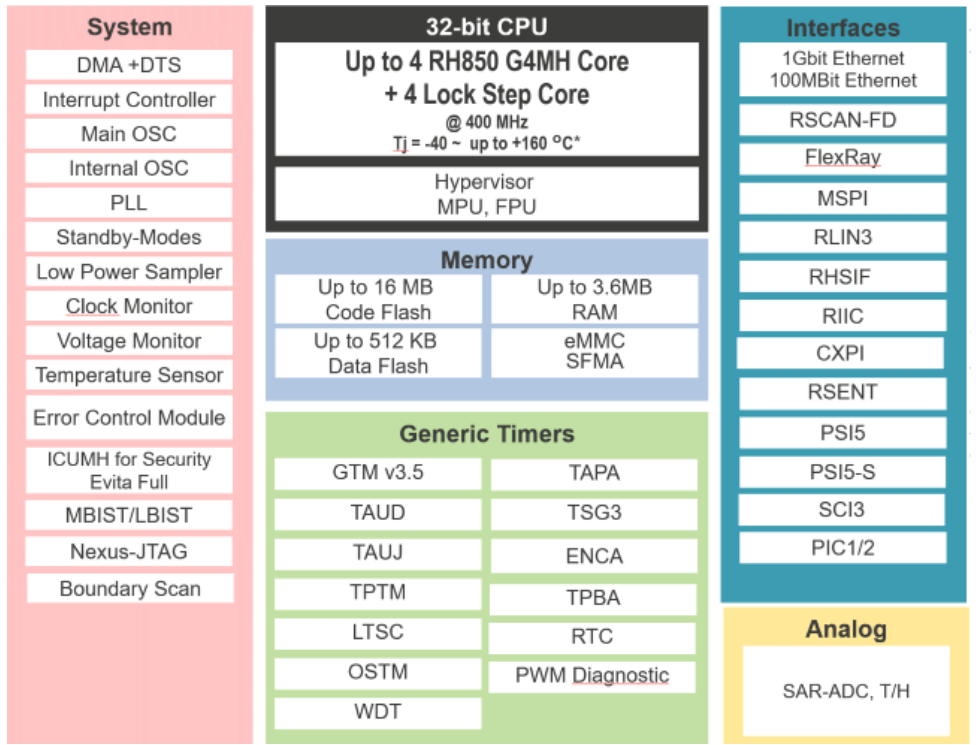
- 借助ARMv8-M的TrustZone硬件虚拟化技术实现
  - 近乎于0的性能开销
- 在Secure World执行RTOS、Non-secure World执行IoT OS
  - 对OS代码修改极少
- 局限：目前支持TrustZone的MCU不多

※ https://www.toppers.jp/safeg-m.html

虚拟化技术：高端车载MCU

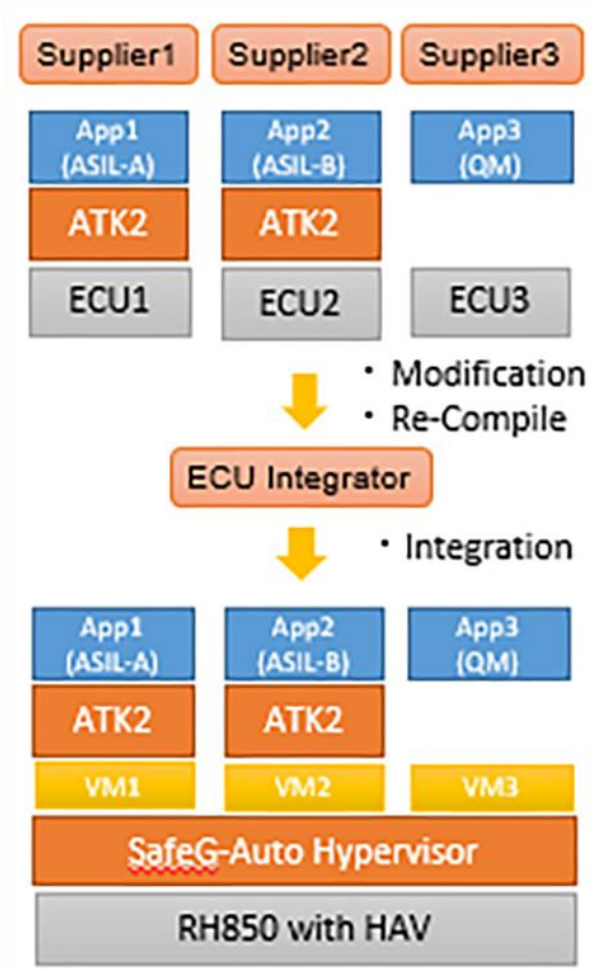
RH850/U2A

- 面向域控制器的高性能多核MCU
  最对4核心RH850 G4MH @ 400 MHz
- 支持硬件虚拟化辅助
  可优化虚拟机上下文切换



TOPPERS/SafeG-Auto Hypervisor

- CPU虚拟化
- 硬件资源分区
- 时间隔离提高确定性
  TDMA调度算法
- 支持虚拟机间通信
  1对1、1对N
- 可将多个ECU的功能集成到单一RH850/U2A处理器上
- 示例可同时执行数个AUTOSAR CP OS（TOPPERS/ATK2内核）



※ https://github.com/toppers/safeg-auto

实时Linux

Linux作为RTOS逐渐被广泛采用

- PREEMPT_RT补丁趋向成熟
  经过二十余年发展，已可满足很多实时系统需求
  很多修改都已被合并进标准内核（mainline）
- 产业界正在积极投入资源推动开发
  车载：Automotive Grade Linux
  机器人：Robot Operating System 2

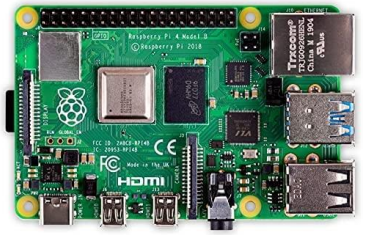
对于有安全要求的系统，以下问题值得关注

- PREEMPT_RT有多硬?
  有哪些因素可能对实时性造成影响
- 系统的隔离功能是否充分
  不同进程间干涉程度、减轻方法
- 共享资源的调度是否合理
  内核对象、存储、网络、GPU等

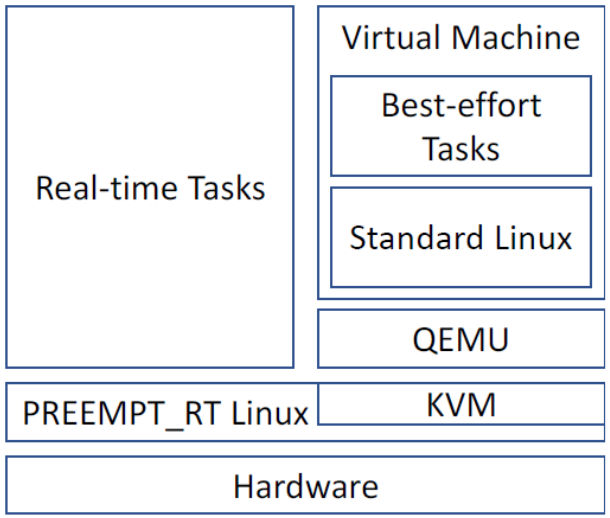
实时Linux：PREEMPT_RT性能评测

来自合作企业的性能要求

- 实时任务目标：1ms Period、100us Deadline
- 同时存在大量的非实时任务（Best-effort Tasks）
- 硬件规格：Raspberry Pi 4级别（1.5GHz A72 x4）



评测实验设计和结果



使用QEMU虚拟机隔离非实时任务
减少内核共享对实时任务的干涉
标准内核提高非实时任务的吞吐量

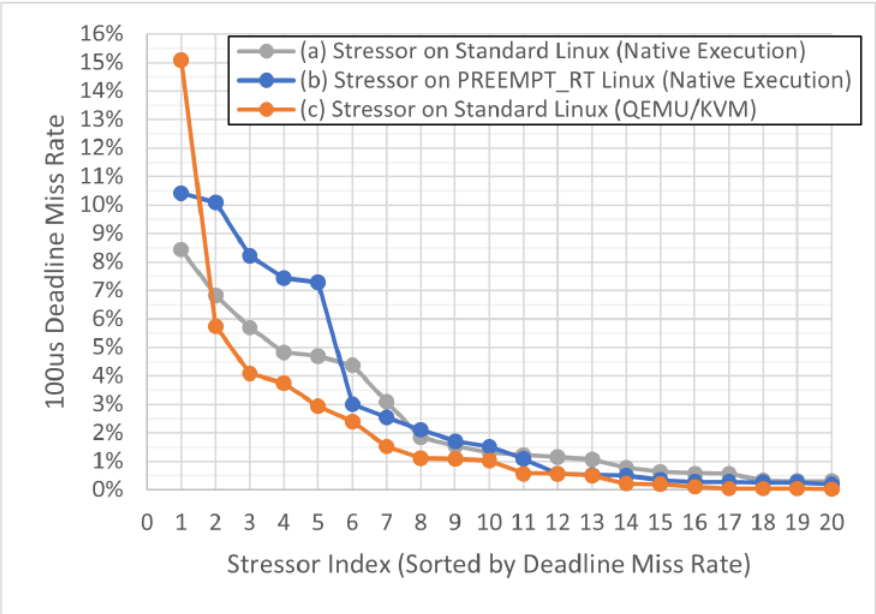


Fig. 5 The index plot of stressors sorted by measured deadline miss rate. (For clarity, only the first 20 stressors are plotted since the remaining ones have near-zero deadline miss rate.)

使用stress-ng压力测试的各种负荷作为非实时任务，用以测试PREEMPT_RT在极端负荷下的表现

结果表明通过虚拟机隔离非实时任务可有效降低Deadline Miss率（比较左图b和c）

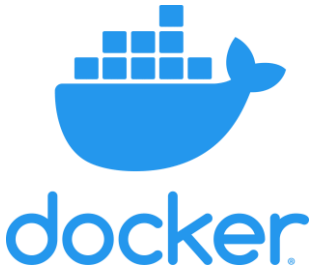
负荷1的Deadline Miss率很高经调查发现原因在于RasPi4的总线瓶颈于OS内核无关

在极端情况下，PREEMPT_RT 5.4内核也满足了95%的Deadline

实时Linux：开源容器的安全分析

容器技术开始在嵌入式系统设计中普及

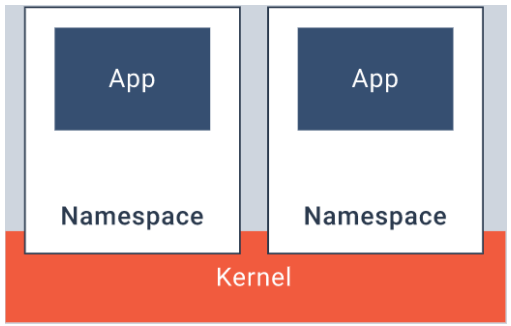
- 比进程拥有更明确的隔离边界
- 比传统虚拟机更轻量灵活
- 但内核共享的OS层虚拟化也存在不少安全隐患
  今年：CVE-2022-{0492,2588}

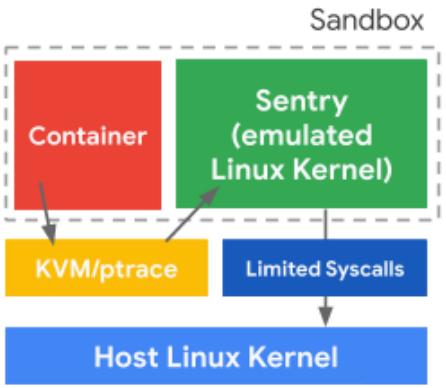


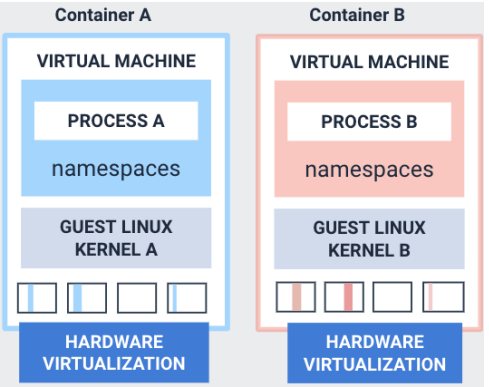
OCI（Open Container Initiative）运行时的安全分析

- OCI运行时：容器技术架构中负责实现资源隔离功能的组件　　主要分类↓
- 对于实时系统，安全性分析除了Security还要讨论性能隔离
- Linux内核参数众多，抽象资源也可成为攻击平面
  E.g. 连接数、线程数

内核共享型	沙盒型	轻量虚拟机型
代表：runc（Docker）	代表：runsc (gvisor)	代表：Kata Containers

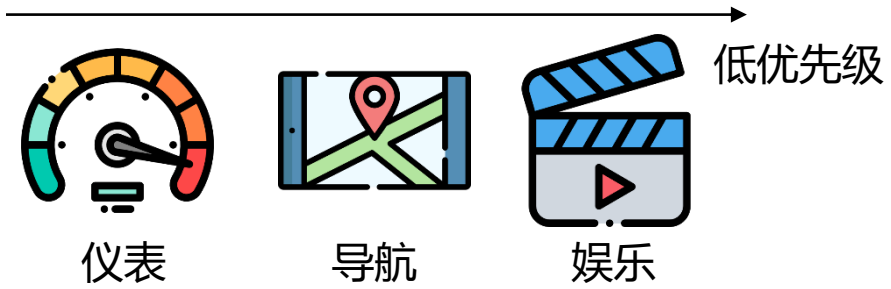






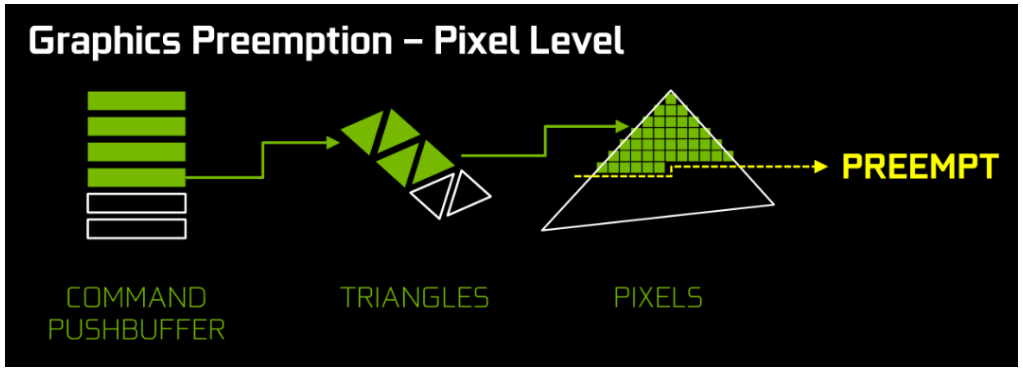
实时Linux：GPU渲染调度

高端车载系统多使用GPU进行3D渲染
- 随着功能进化、使用GPU的应用也在增加
- 共享GPU的应用，按安全等级有不同优先级
  低优先级应用不应影响高优先级的渲染帧率
- 当下Linux的GPU驱动对渲染调度支持极为有限



各厂GPU硬件陆续支持细粒度抢占

NVIDIA:



AMD: Mid-command Buffer Preemption

Intel: ExecList Submit Port

Linux DRM（开源GPU驱动子系统）进展
- AMD、Intel已经正式转向开源多年
- NVIDIA、Imagination今年开始发布DRM驱动

https://github.com/NVIDIA/open-gpu-kernel-modules

https://gitlab.freedesktop.org/frankbinns/powervr

GPU资源管理和渲染调度
- 预计将成为未来实时Linux系统的重要功能
  DRM core中已经有了初步的调度器实现
- 目前正与车企、SoC厂商就相关课题展开合作研究

- 谢谢！

邮箱： liyixiao7@gmail.com liyixiao@ertl.jp