可编程调度框架

OS内核实验室

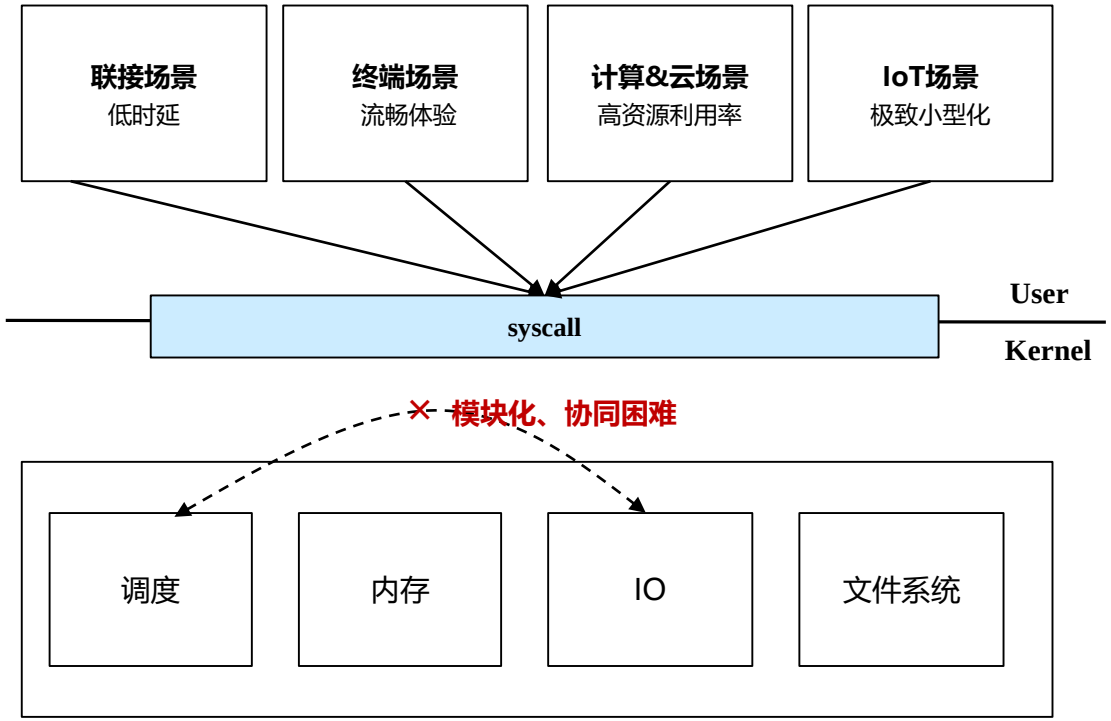
Author/ Email: 陈辉/00515652

Version: V1.0(202301)

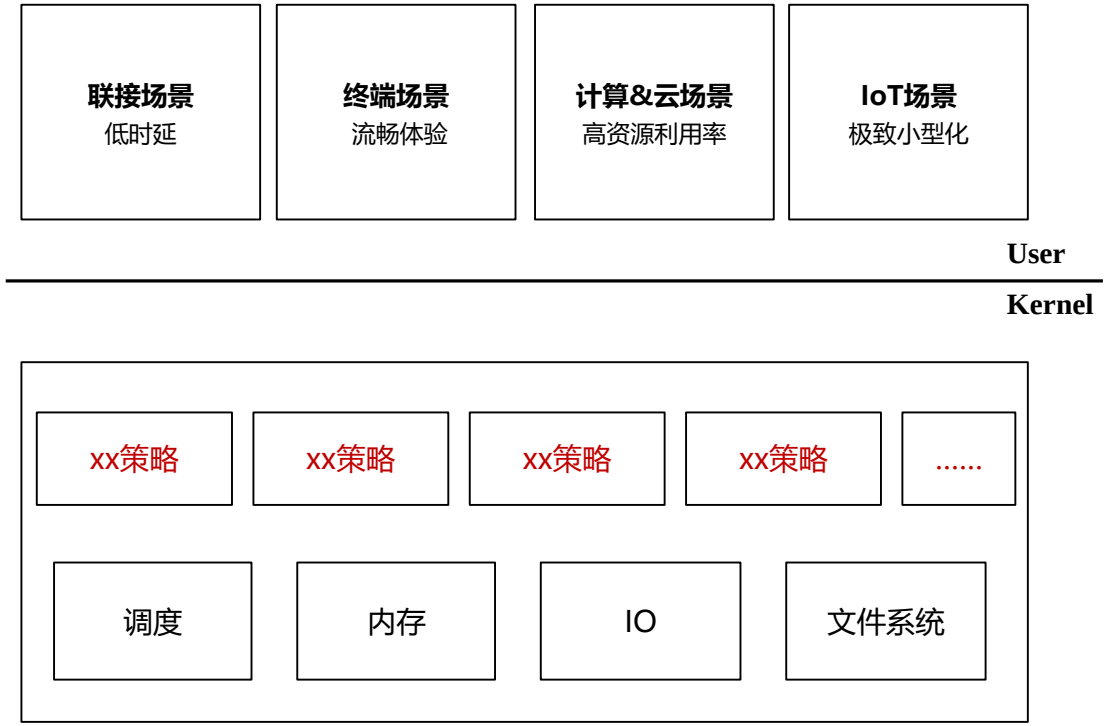
HUAWEI

<一> 背景

- 抽象化，模块化导致信息丢失、协同困难，性能不佳



- “千人千面”，定制策略多，架构臃肿腐化，维护成本高



发展历程

2004年

Con Kolivas 提交“Pluggable cpu scheduler framework”patch

2009年

IEEE Student Conference on Research and Development 发表《Runtime CPU Scheduler Customization Framework for a flexible mobile operating system 》论文

2019年

Aleix Roca Nonell 发表《A Linux Kernel Scheduler Extension For Multi-Core Systems 》论文

2021年

SOSP 论文《ghOst: Fast & Flexible User-Space Delegation of Linux Scheduling》

2021年

Peter Oskolkov提交《User-managed concurrency groups》patch

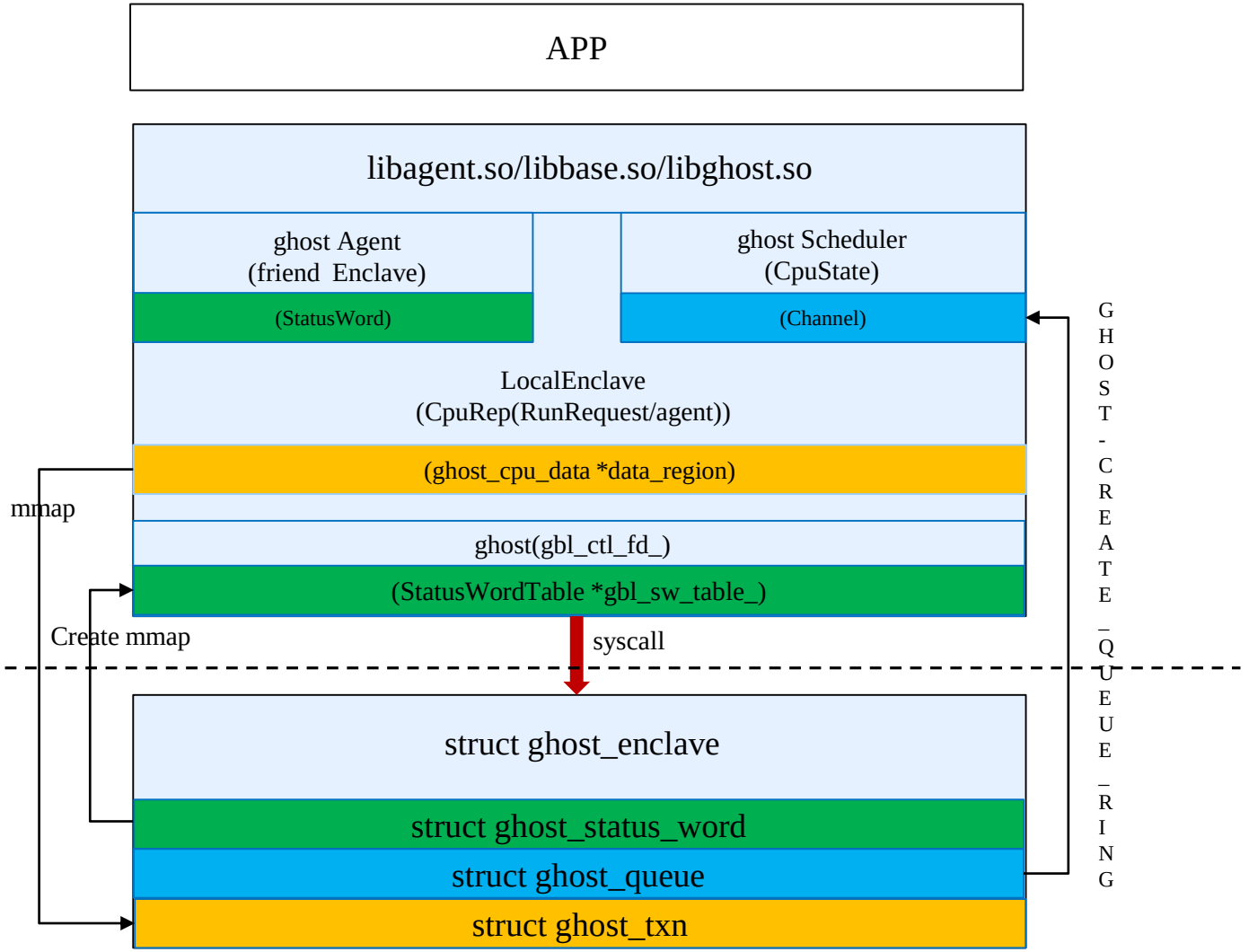
2021年

Roman Gushchin提交《Controlling the CPU scheduler with BPF》patch

2022年

Tejun Heo *提交《Sched: Implement BPF Extensible Scheduler Class》*

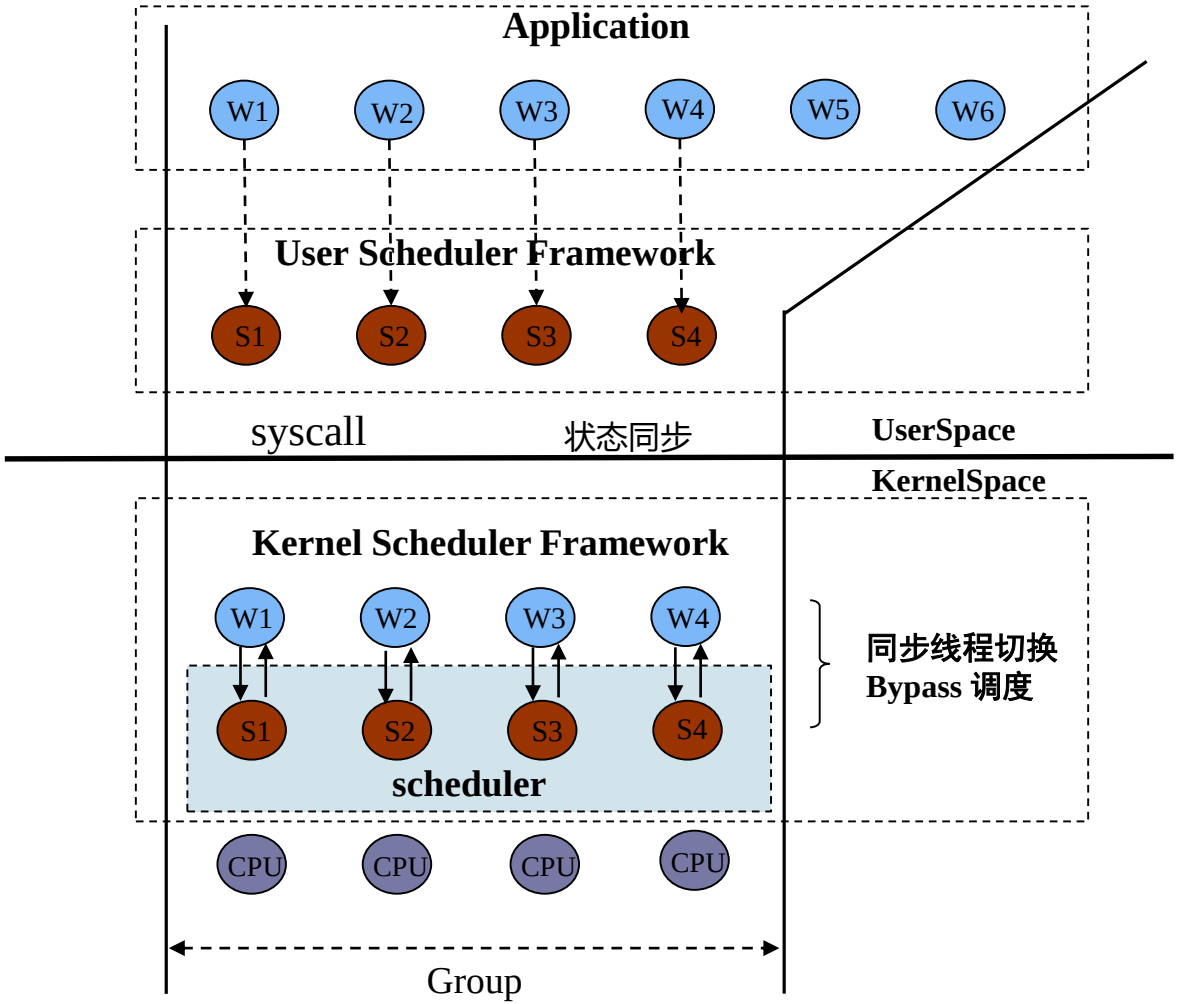
Google ghOst scheduler Framework



1）每个用户进程可以自定义此进程的调度策略，由进程开发人员来编写和管理；

2）此框架由两部分组成，一部分是内核态ghost调度类，一部分是用户进程中的ghost agent线程； ghost调度类主要负责监控业务线程运行状态如启动，休眠，抢占等，并通知ghost agent线程执行对应的动作。 ghost agent线程负责接收内核传递过来的信息，执行用户编写的各种策略。

Google UMCG scheduler toolkit



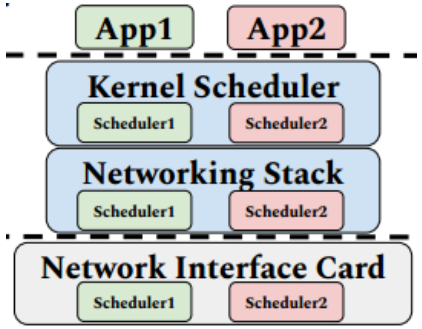
1、进程级自定义调度，进行隔离，应用于安全沙箱；
2、server与worker相互感知，感知worker线程内核态抢占或者阻塞，应用在协程场景，提升吞吐；

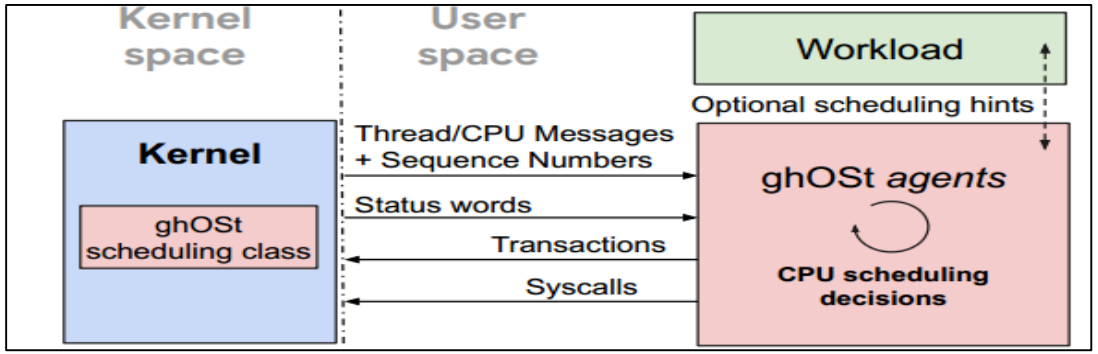
设计思路：
1、视一个server为一个UMCG的虚拟CPU，对应到一个物理CPU上；
2、一个worker必须绑定到一个server上才能运行；
3、server运行挑选一个worker，切换到worker上运行；
4、worker阻塞或者被抢占时，唤醒server线程。

Syrup scheduler Framework

业务需求：

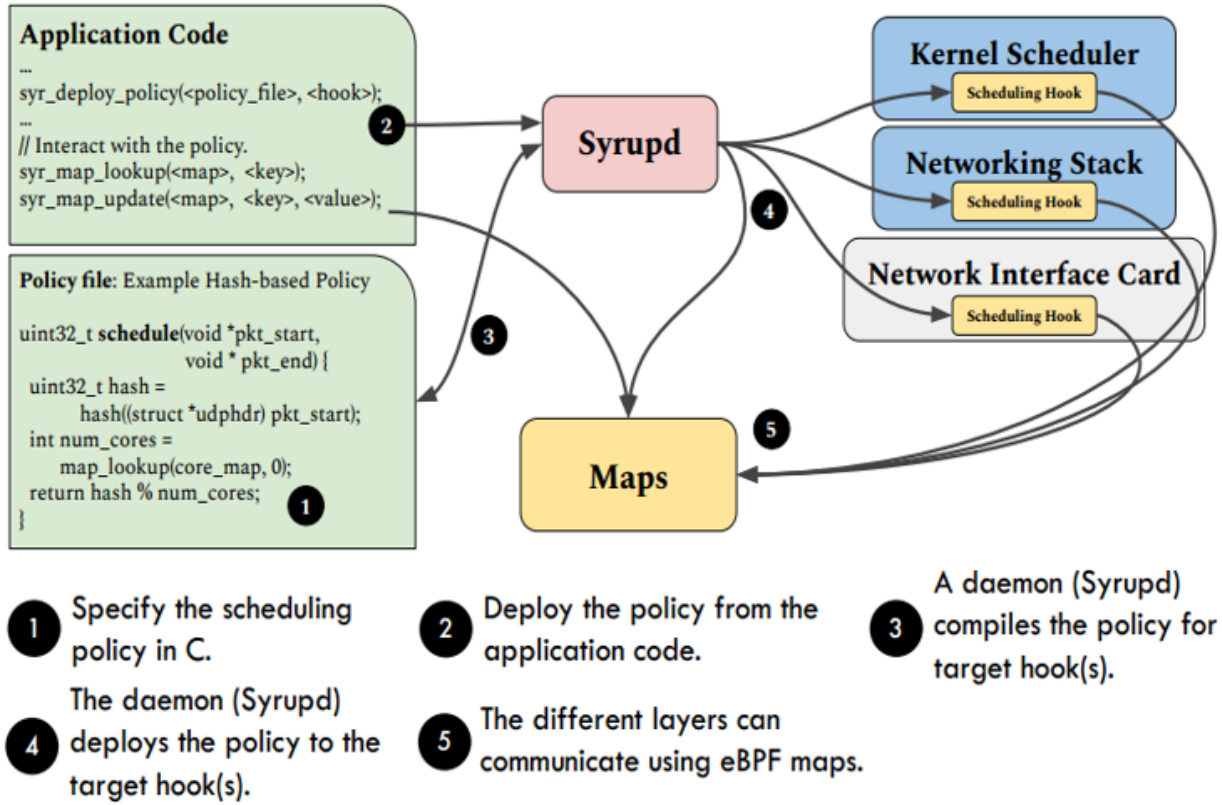
- **不同场景不同调度策略**：“任务排布稳定”场景适合FCFS调度策略；“任务排布高度变化”场景适合抢占和资源分割的调度策略(如CFS)；“内存敏感性”。
- **低底噪负载：** ~1us or less！
- **调度策略安全隔离：**





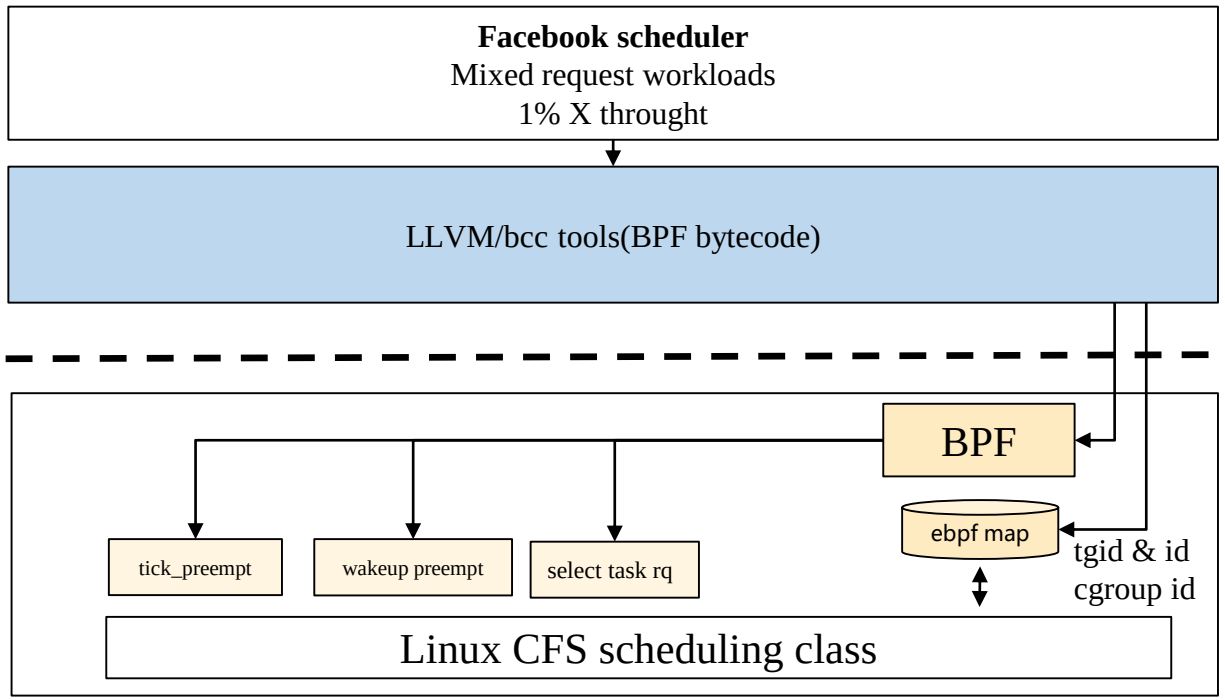
Syrup设计如何满足业务需求：

- **不同场景不同调度策略**：基于eBPF和ghOSt，将调度装换成策略匹配问题
- **低底噪负载：** ~1us or less！
- **调度策略安全隔离：**引入全局仲裁器，为不同应用使能不同调度策略。



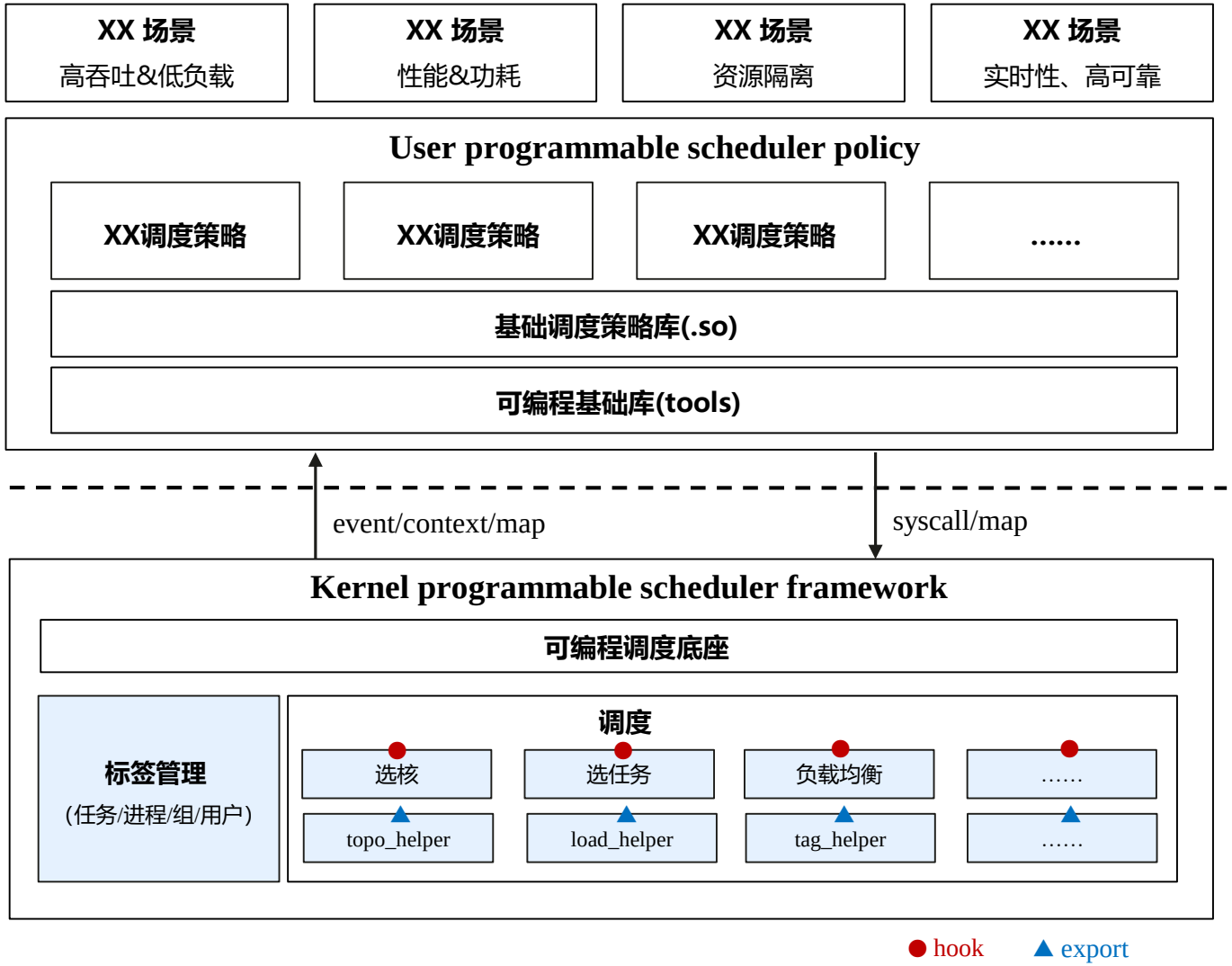
1. Specify the scheduling policy in C.
2. Deploy the policy from the application code.
3. A daemon (Syrupd) compiles the policy for the target hook(s).
4. The daemon (Syrupd) deploys the policy to the target hook(s).
5. The different layers can communicate using eBPF maps.

Facebook ebpf scheduler Framework



1) 打通ebpf scheduler的基础功能，目前主要实现一个简单的案例。
2) 基于ebpf 提供简单的用户态调度策略，实现用户态抢占&选核策略，应用于长短请求混合场景，提升1%的吞吐。

可编程调度框架



可编程调度框架：

1、可编程调度底座，提供基础系统调用，事件通知，数据共享机制与一致性保护。

2、提供标签化管理机制，支持任务/进程/组/用户等对象的自定义标签扩展；

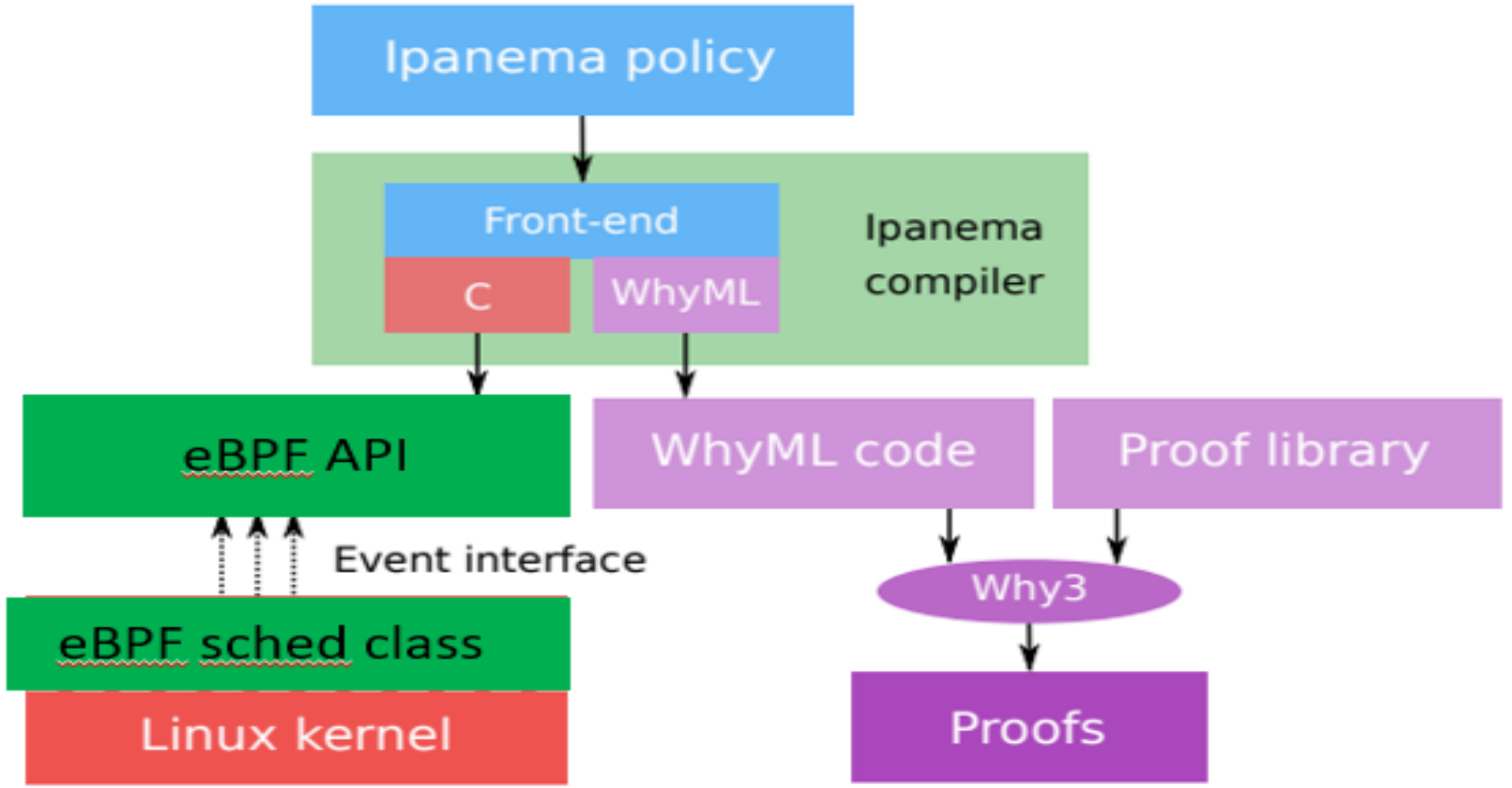
3、提供基础的库函数，支持用户快速编排&扩展，用户编程友好。

支持用户态调度策略扩展的方式：

1、基于标签化的调度策略：基于对象的标签信息，定制对应的调度策略；

2、基于反馈式的调度策略：动态收集环境信息，系统状态信息，PMU信息等，进行数据分析，预测的动态调度策略；

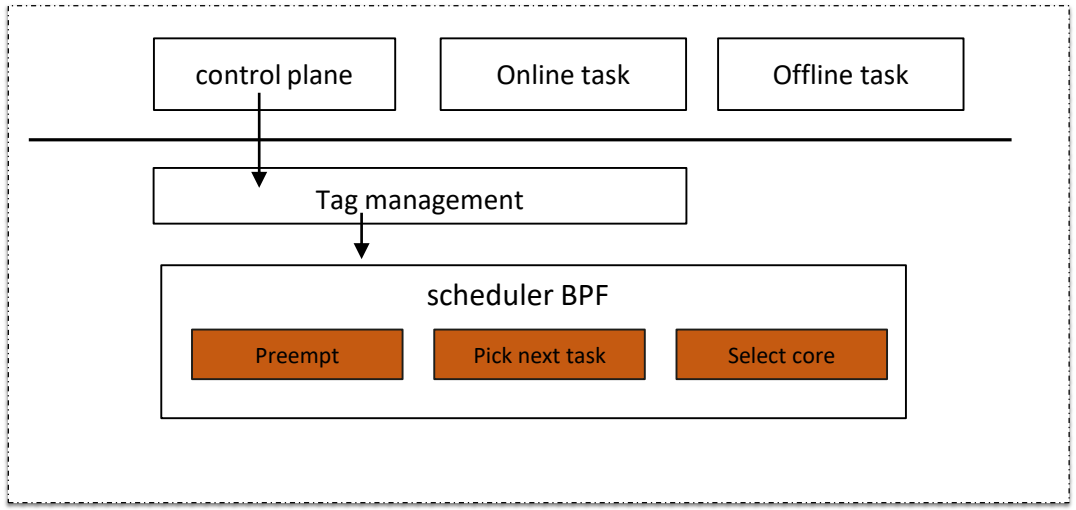
可编程调度框架


Ipanema policy
Front-end
C
WhyML
Ipanema
compiler
eBPF API
WhyML code
Proof library
Event interface
Why3
eBPF sched class
Linux kernel
Proofs

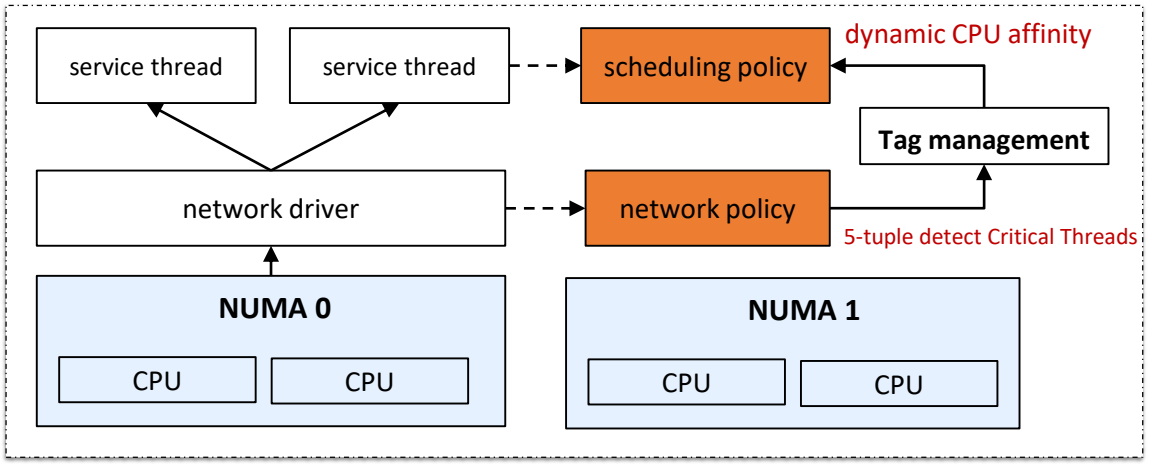
Scheduler BPF

application scenario

Cloud： support hybrid deployment, online tasks suppress offline tasks, improved resource utilization



HBase： coordinate scheduling with the network, dynamic tuning CPU affinity, improved I/O throughput



openEuler

<六> 生态建设

验证

开发

部署 prog

快速部署，缩短验证周期

- 支持热升级

简单高效开发

- 支持高级语言
- 丰富的基础库

发布prog

社区

交流&反馈

发布

安全可靠

- 数字签名
- prog商店

推广

pull

部署

- 支持安全验证
- 在线无缝部署

变化：

1、内核越来越薄，主要实现框架能力；

2、内核开发模式，趋于短平快的开发模式；

3、新的商业模式，prog商店，生态环境；

Thank you.