
KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2023


KubeCon | CloudNativeCon

OPEN SOURCE SUMMIT

China 2023

使用新的异步 I/O API 构建代理：探索 Envoy 的 io_uring 集成

徐贺杰 / 谢之皓

我们是谁

徐贺杰 (Alex)
英特尔云软件工程师 / Envoy 维护者 /
OpenStack Nova/Placement 核心评审员

Alex 是英特尔的云软件工程师和 Envoy 的维护者。目前，他专注于 Service Mesh 数据平面和 Envoy 社区。过去，他在 OpenStack 和 IaaS 方面也有 10 年的经验。

谢之皓
英特尔云软件工程师

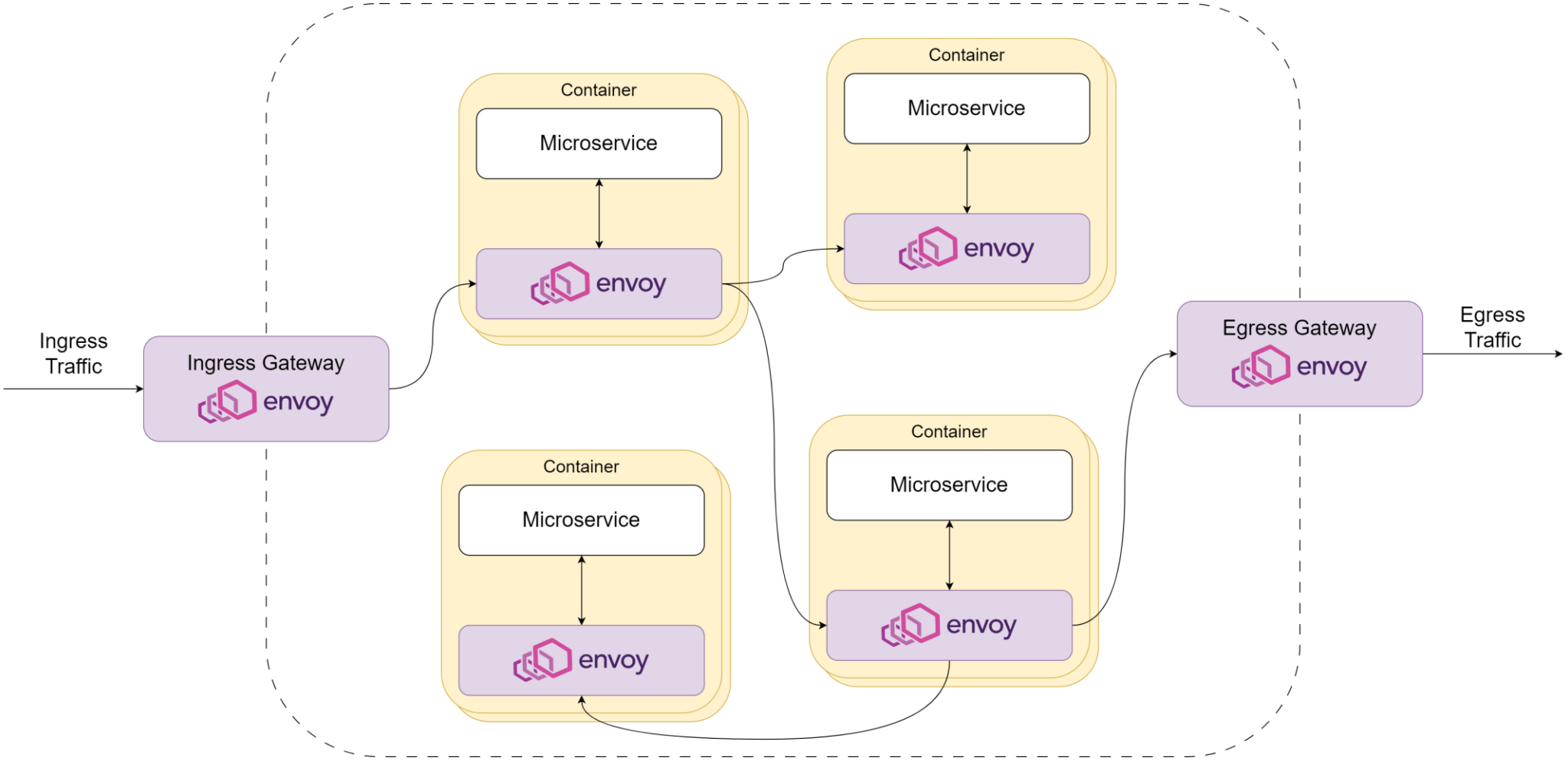
Zhihao 是英特尔服务网格团队的云软件工程师。他负责 Envoy 项目，重点优化网络、负载均衡、路由和访问控制的性能。

议程

- 背景
- Envoy 的 I/O 架构
- 将 io_uring 集成到 Envoy 的 I/O 架构中
- API 与基准测试
- 状态更新
- 自由提问

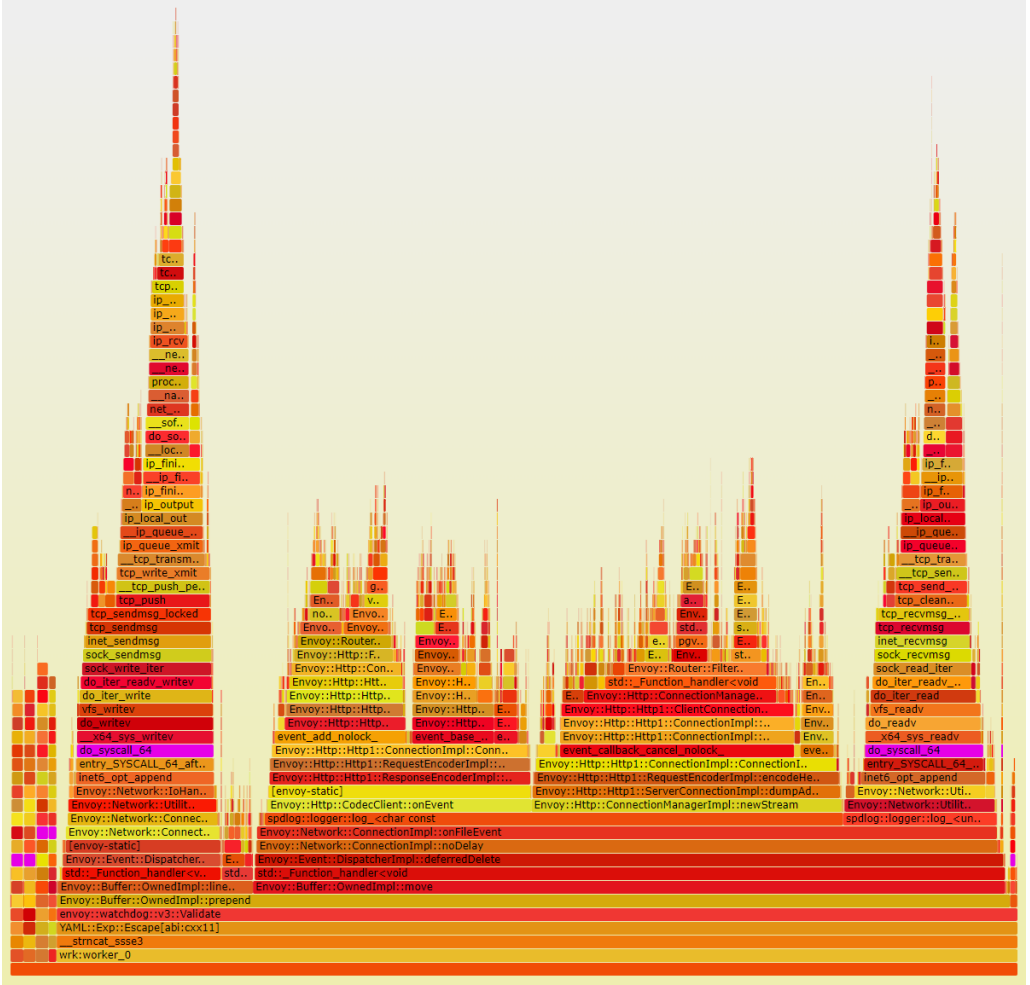
背景

Envoy 无处不在


Ingress Traffic
Ingress Gateway
envoy
Container
Microservice
envoy
Container
Microservice
envoy
Container
Microservice
envoy
Container
Microservice
envoy
Egress Gateway
envoy
Egress Traffic

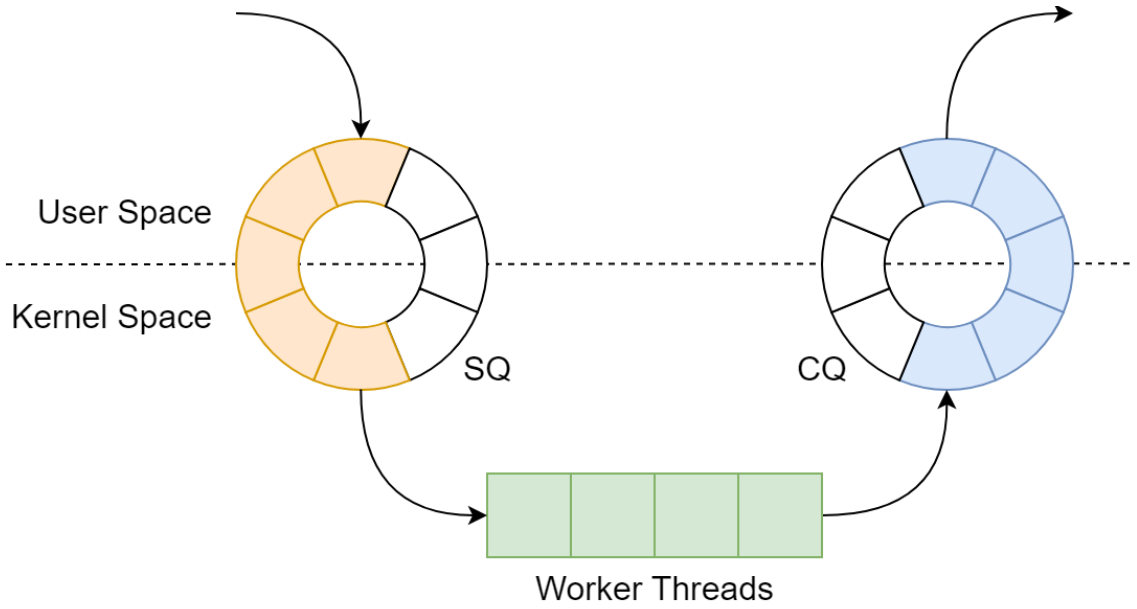
我们之前的发现

- 在一个简单的 HTTP 路由用例中，将近 30% 的 CPU 使用率分配给了与读写相关的系统调用
- 您可以在 Envoy 中使用 VCL，但使用 VCL 需要外部 VPP 进程的支持，这涉及到大量的调整校对工作



集成 io_uring 的好处

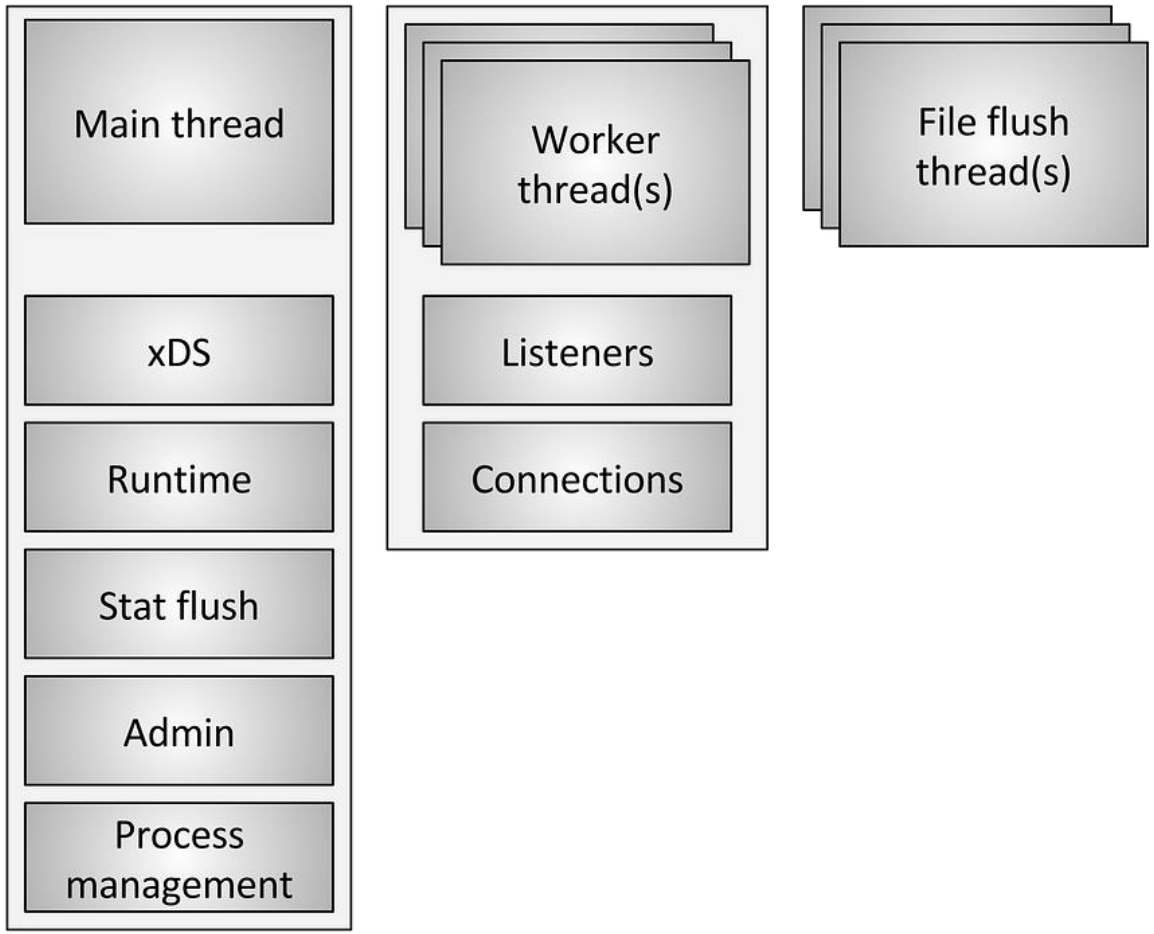
- io_uring 是一种新的异步 I/O 应用程序接口，由 2 个环形缓冲区负责提交 (提交队列) 和完成 (完成队列) 的交换
- 更少的系统调用，更少的内存拷贝
- io_uring 以完全异步的方式运行，这与 Envoy 现有的 I/O 模型不同



Envoy 的 I/O 架构

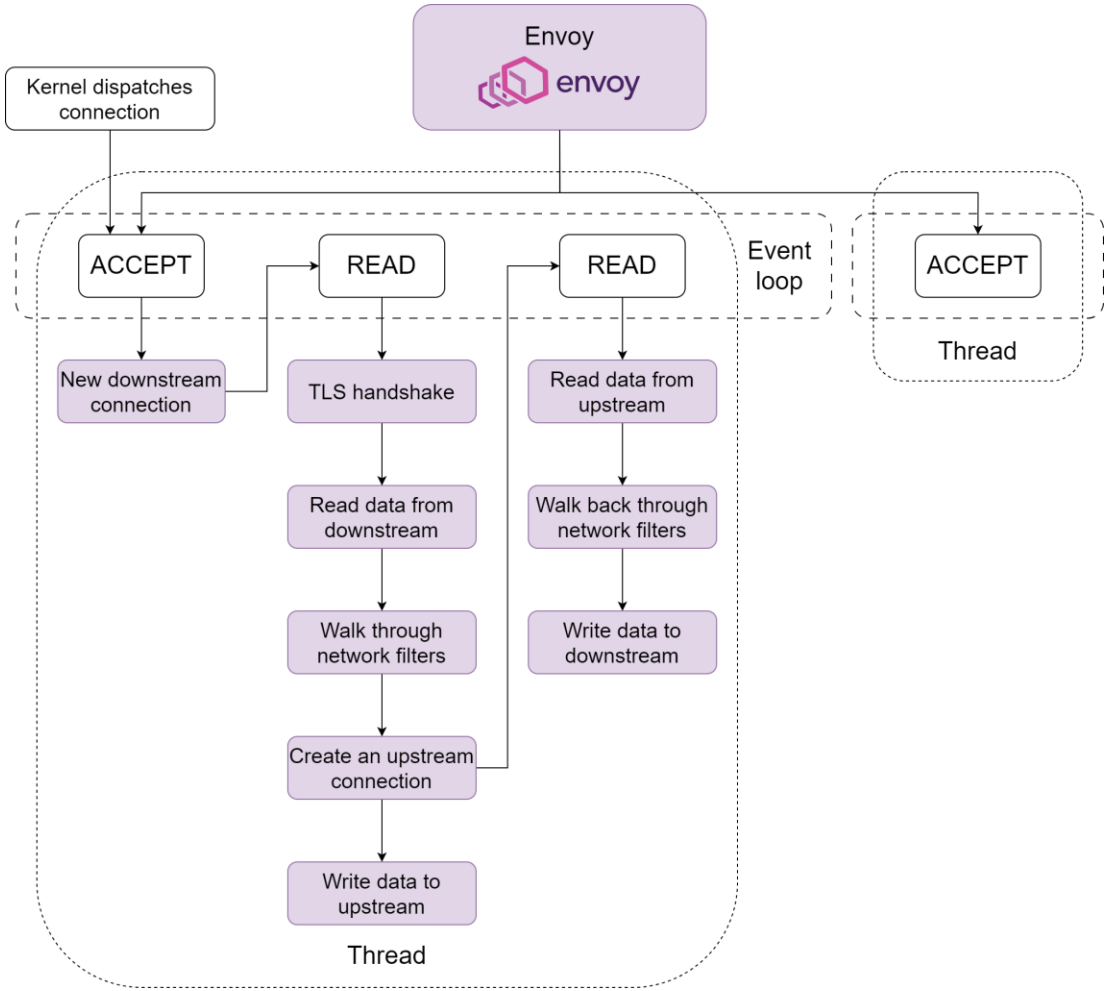
线程模型

- 单进程，多线程
- 主线程负责控制路径
- 工作线程负责数据路径
  - 连接由工作线程处理



I/O 模型

- 内核选择一个线程以分派新连接，连接将在整个生命周期中绑定到该线程上
- 每个线程都运行一个事件循环，用于 I/O 复用
  - 基于 libevent
- 基本上，在整个生命周期中，连接只绑定到一个线程上
  - 套接字注册到当前线程的事件循环中




KubeCon | CloudNativeCon

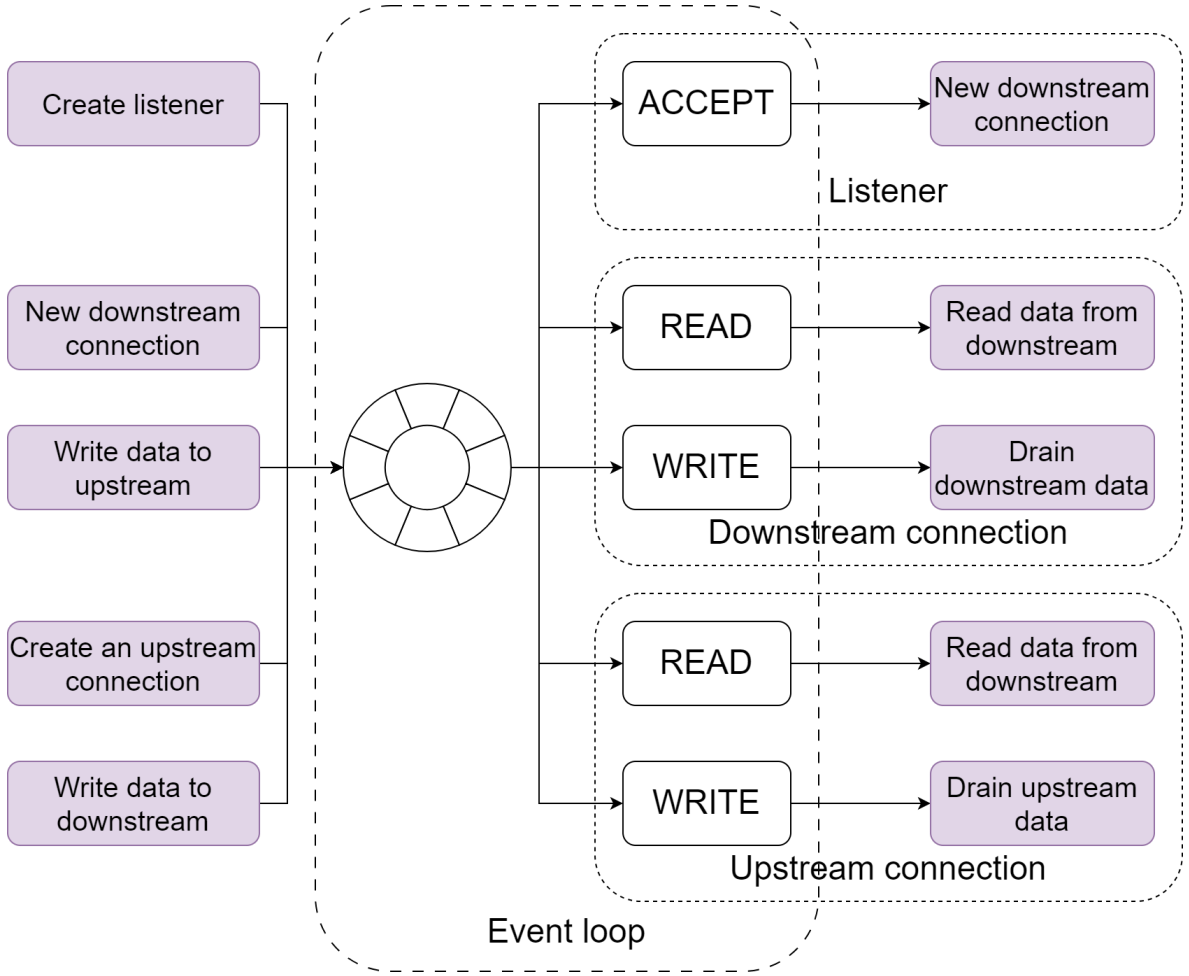
OPEN SOURCE SUMMIT

China 2023

将 io_uring 集成到 Envoy 的 I/O 架构中

将 io_uring 集成到事件循环中

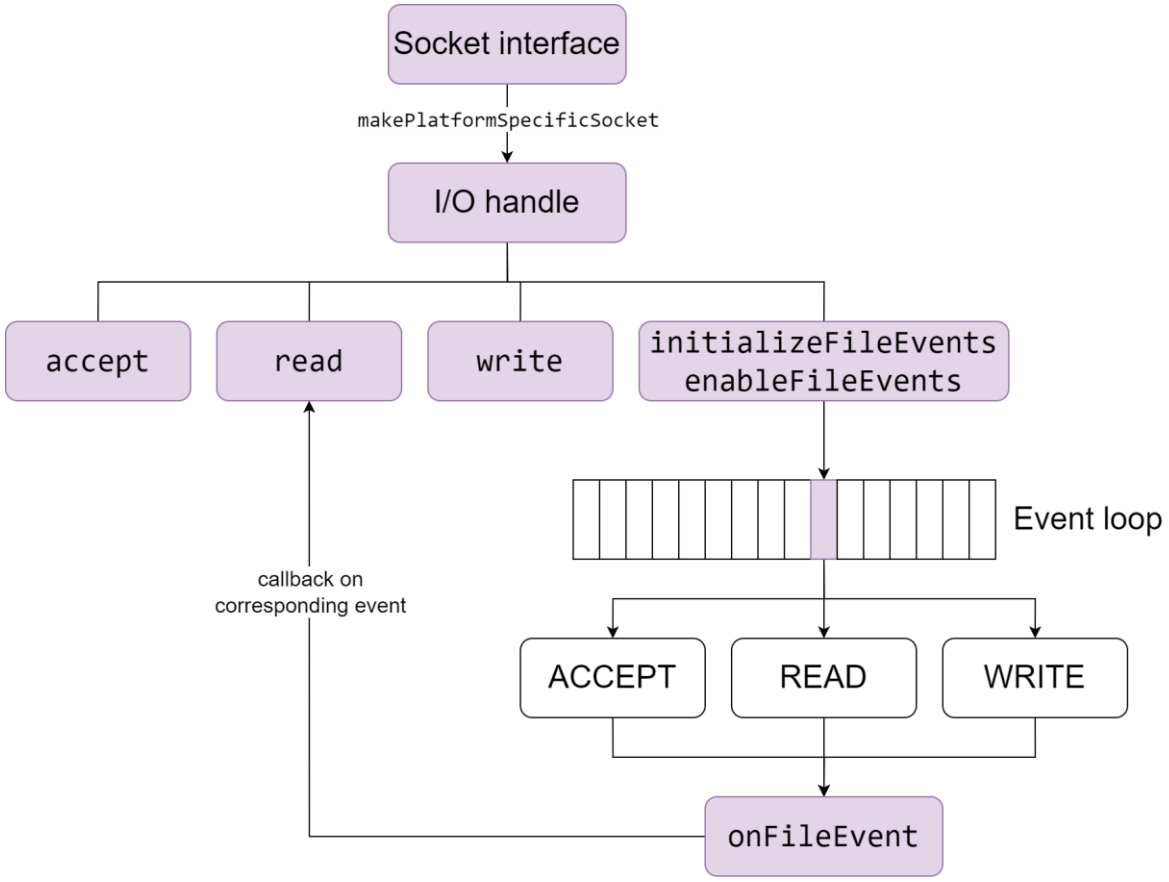
- libevent 负责每个线程中的事件循环，而这些线程并非原生支持 io_uring
- liburing 支持 eventfd，只要有提交完成并添加到完成队列，我们便知道在完成队列中有完成队列事件需要处理
- 使用 eventfd 来桥接 libevent 事件循环和 io_uring



https://github.com/libevent/libevent/issues/1019

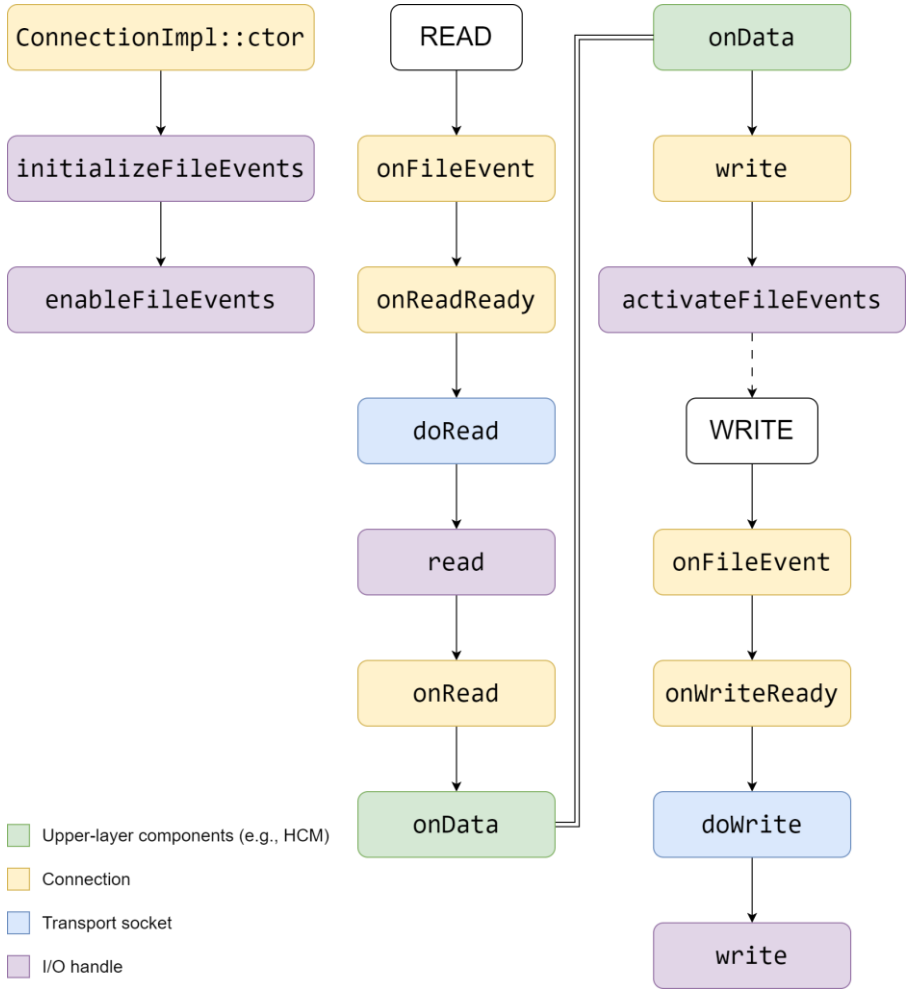
套接字接口

- 套接字接口决定使用哪个 I/O 句柄
  - 默认
  - VCL
  - 用户空间
- 我们修改了默认的套接字接口以支持 io_uring
  - 检测 Linux 内核功能以启用 io_uring
  - 为每个线程初始化 io_uring



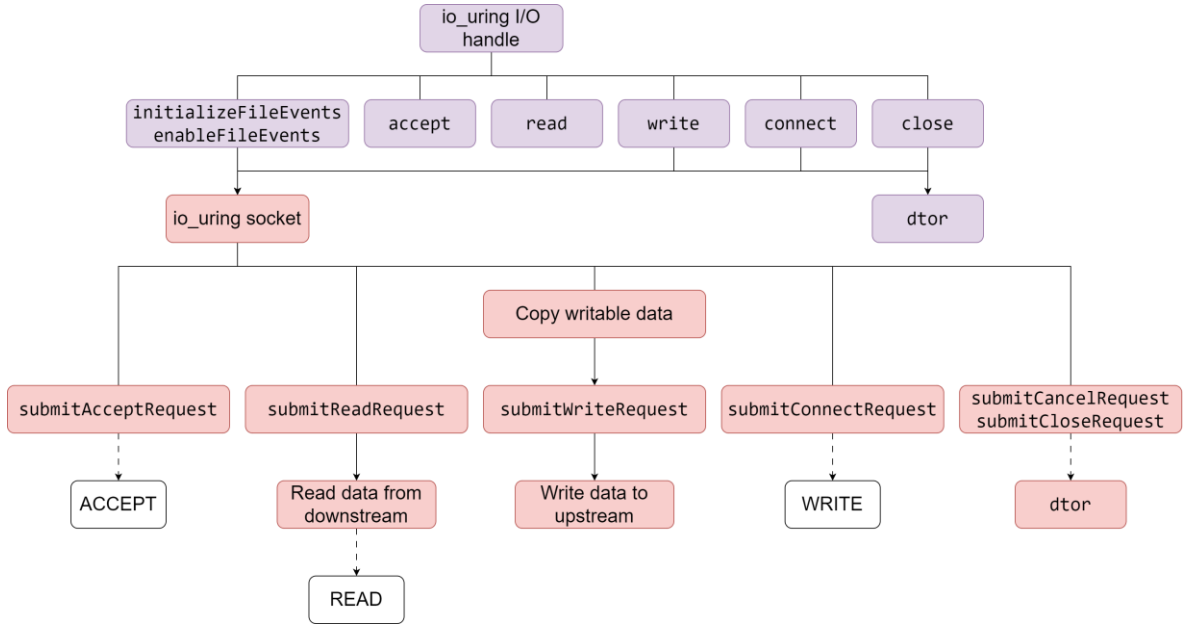
I/O 句柄

- I/O 句柄是 I/O API 的抽象
- 工作流程比较复杂，但上层组件可以忽略套接字的内部细节
- 套接字 API 与 io_uring API 的线程模型不同
  - 例如，套接字 API 中的写入会同步返回完成写入的字节数，但这一过程在 io_uring 中是异步完成的



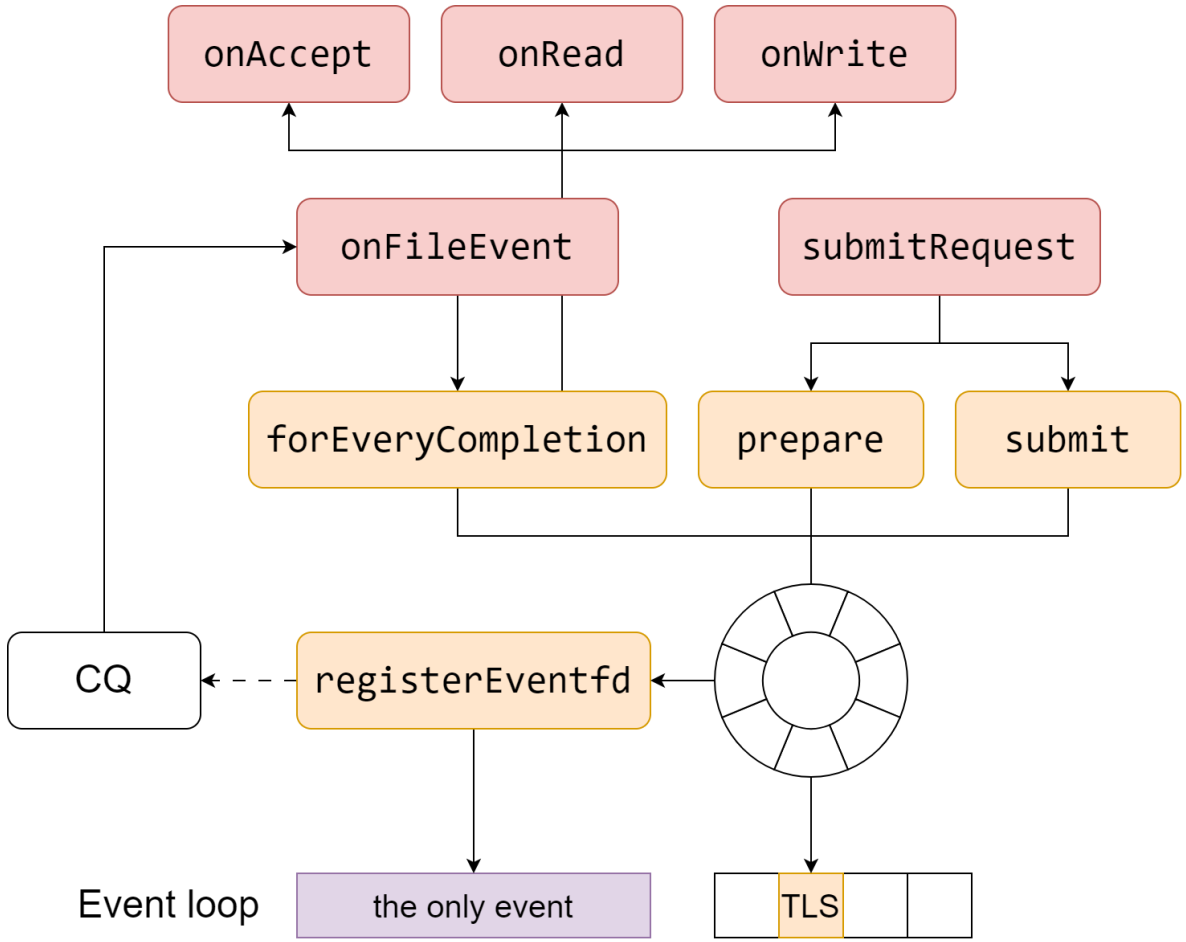
io_uring 套接字

- io_uring I/O 句柄与 I/O 句柄接口保持一致，并与 io_uring 套接字通信
- io_uring 套接字管理其内部套接字的生命周期，有着相比 I/O 句柄更长的生命周期，并负责向 io_uring 提交请求和从 io_uring 完成处理

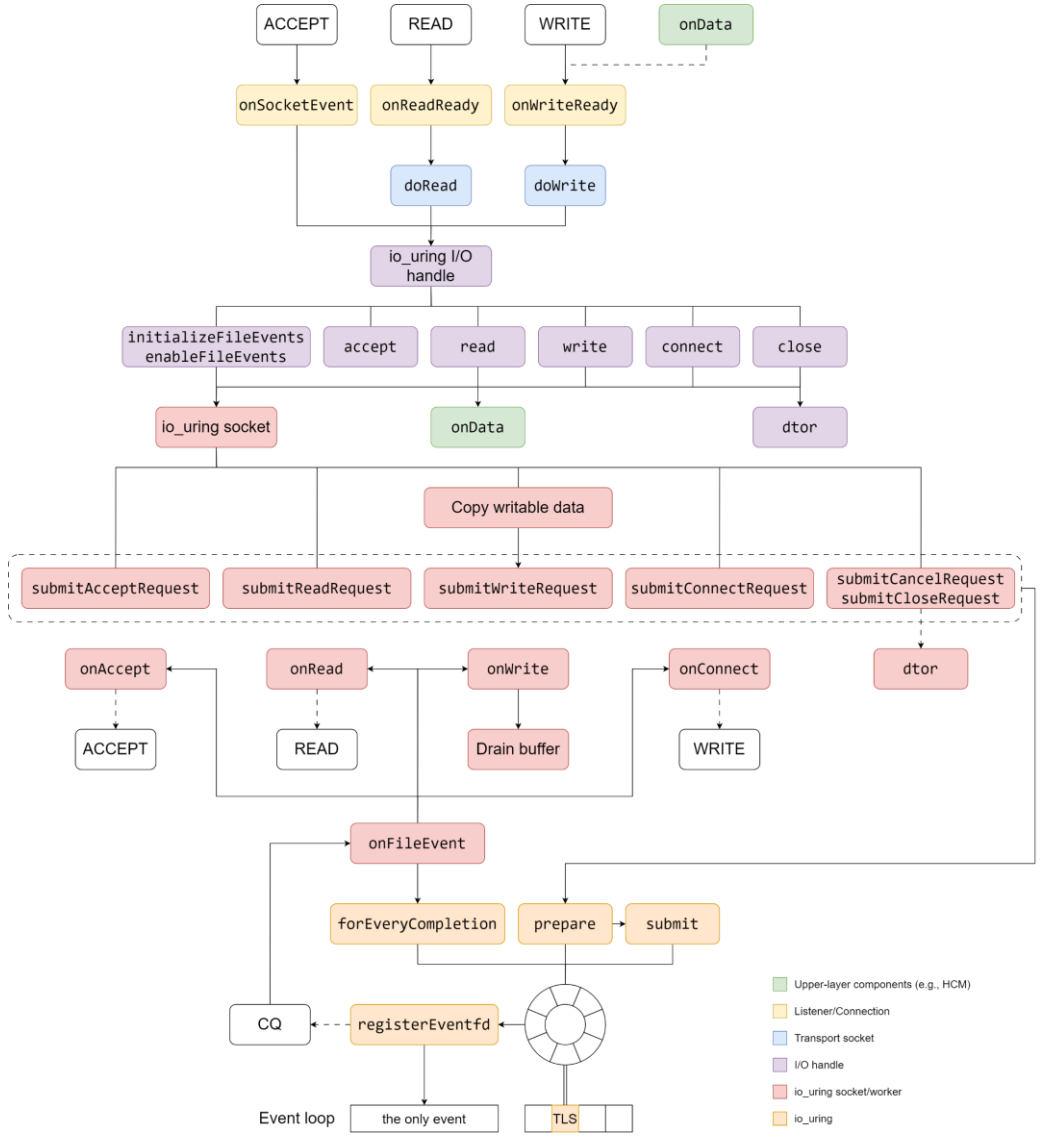


io_uring worker

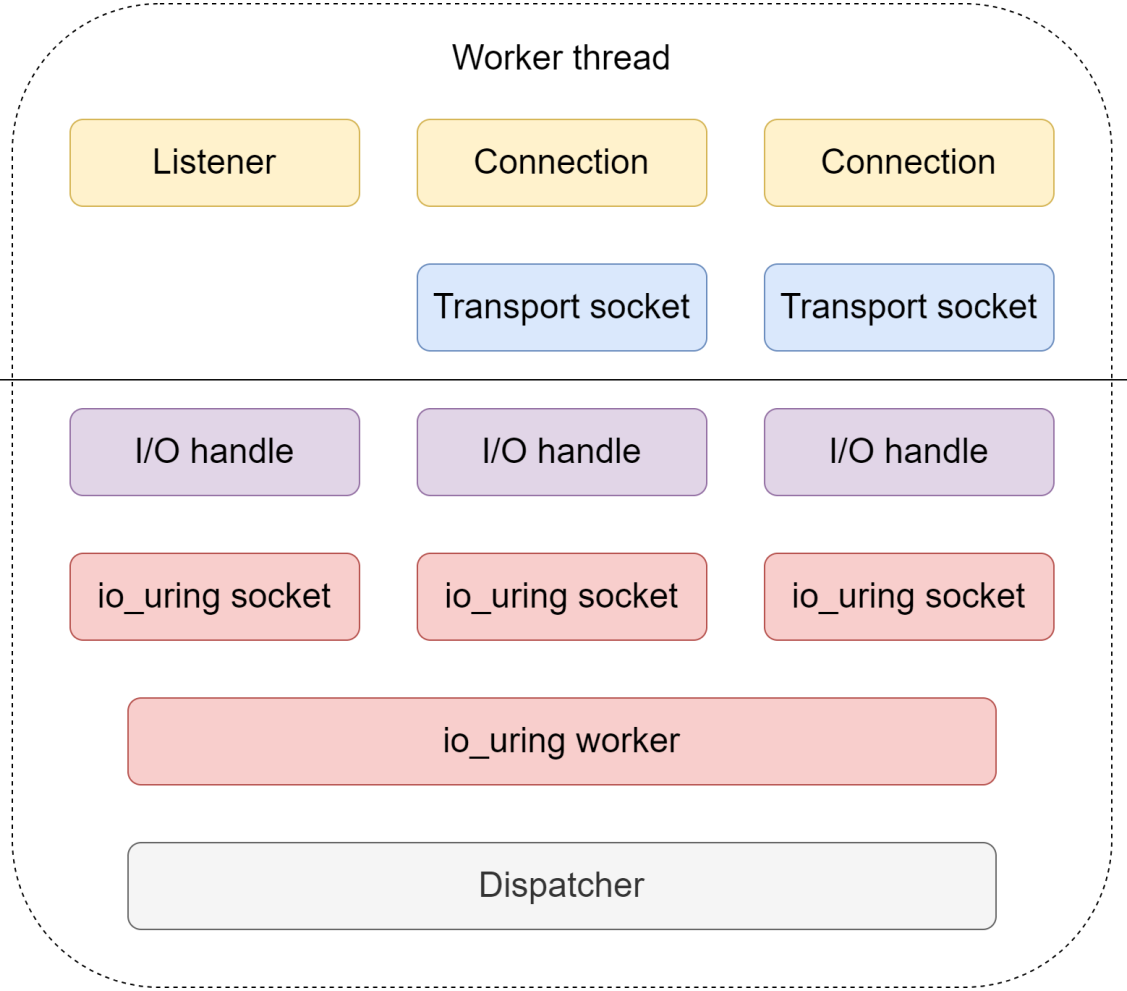
- 是一种 io_uring 事件循环
- 每个线程都有一个 io_uring worker 和一个由 TLS 保证的 io_uring 实例，这一实例注册事件循环中的唯一事件
- io_uring worker 管理线程中的 io_uring 套接字并进行通信



把所有东西放在一起...


ACCEPT
READ
WRITE
onData
onSocketEvent
onReadReady
onWriteReady
doRead
doWrite
io_uring I/O handle
initializeFileEvents
enableFileEvents
accept
read
write
connect
close
io_uring socket
onData
dtor
Copy writable data
submitAcceptRequest
submitReadRequest
submitWriteRequest
submitConnectRequest
submitCancelRequest
submitCloseRequest
onAccept
onRead
onWrite
onConnect
dtor
ACCEPT
READ
Drain buffer
WRITE
onFileEvent
forEveryCompletion
prepare
submit
CQ
registerEventfd
Event loop
the only event
TLS
Upper-layer components (e.g., HCM)
Listener/Connection
Transport socket
I/O handle
io_uring socket/worker
io_uring

…在一个更高的层次上



The upper layer does not aware of io_uring

I/O handles mimic socket API behavior

io_uring sockets manage sockets lifetime and requests

io_uring worker creates per-thread io_uring instance, and is responsible for request submission and completion calling back

io_uring worker registers itself as an event in the dispatcher

Design doc at https://docs.google.com/document/d/17M15J6vh7ZzoJPCY65j6HNoLSqNEoWnQ3Oa9oKIru0o

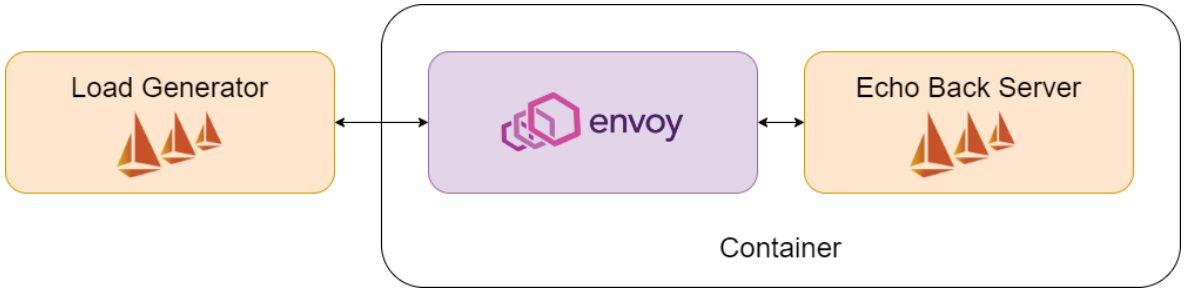
API 与基准测试

如何在 Envoy 中启用 io_uring


```
default_socket_interface: "…default_socket_interface"
bootstrap_extensions:
- name: …default_socket_interface
  typed_config:
    "@type": …DefaultSocketInterface
    enable_io_uring: true
    io_uring_size: 300
    accept_size: 5
    read_buffer_size: 8192
    write_timeout_ms: 1000
    use_submission_queue_polling: false
```

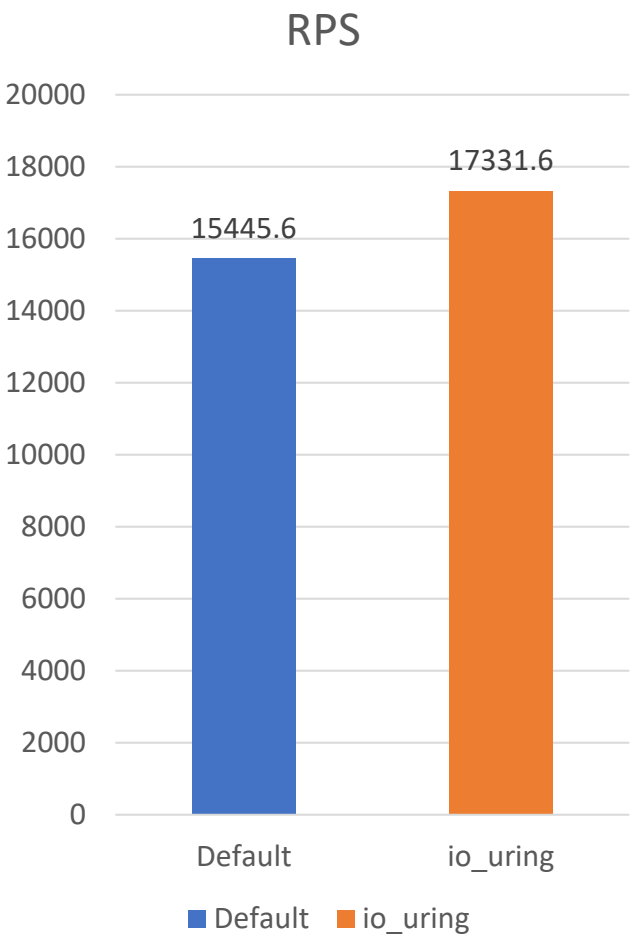
工作负载和配置

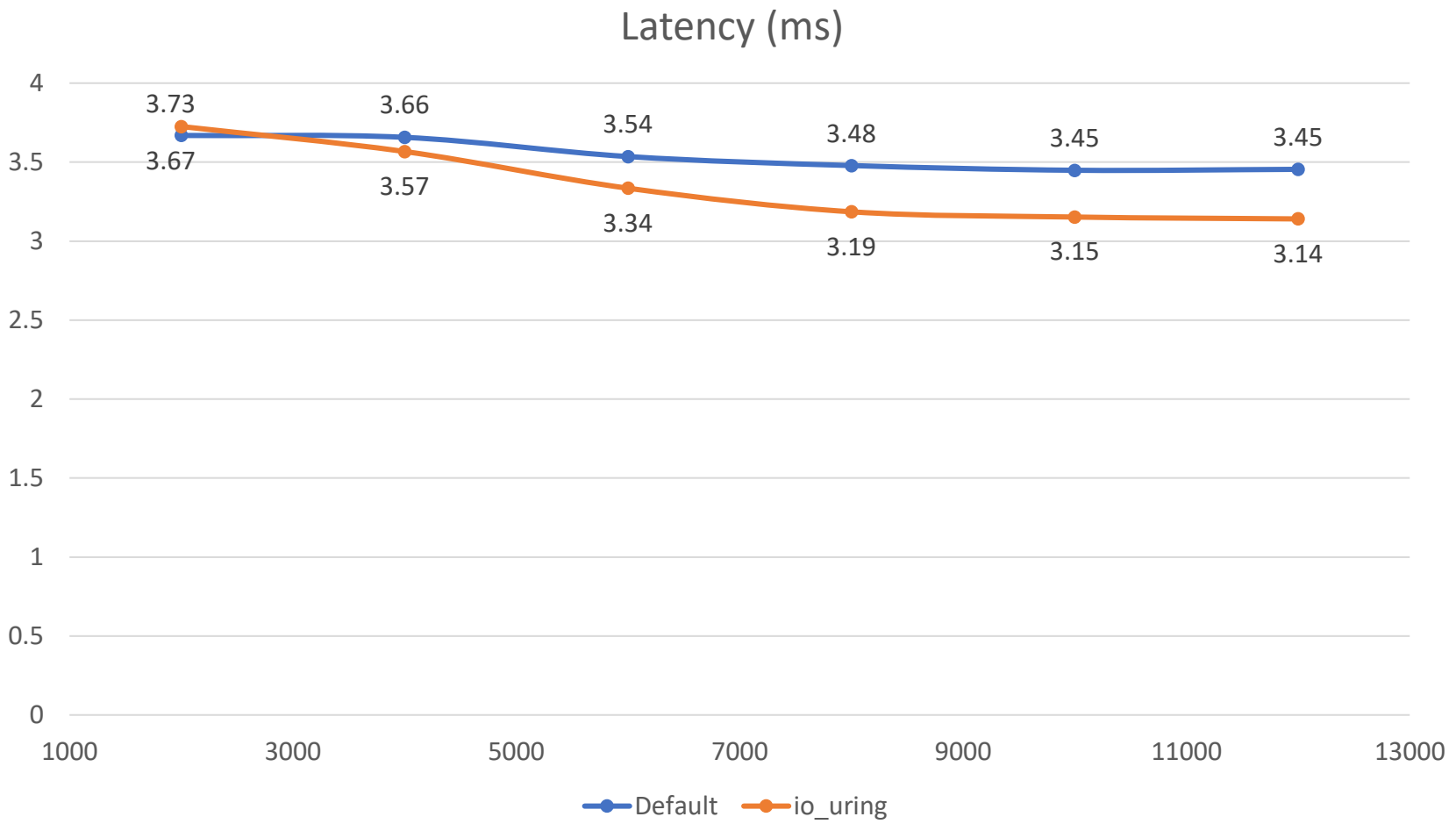
- Envoy 6050489 / 1.27.0
  - 单线程
- Fortio load 1.34.1
  - 64 个连接，不限制 QPS，4kB payload POST
  - 64 个连接, QPS 由 2000 上升到 12000，4kB payload POST
- Fortio server 1.59.0
  - 返回 payload



Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex

性能





Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex

状态更新

社区状态

- 所有导致不稳定状态的问题都已得到解决，在第 3 次迭代中，CI 全部通过
  - Envoy 有着多达 800 个的集成测试，在第 1 次迭代中，90% 的测试都失败了，其中大部分是段错误和超时
  - 得益于 io_uring 套接字的抽象性，我们可以隔离实现并分别测试它们的功能
  - 有些测试在 io_uring 上无法运行，需要重写
- 开始社区合并进程。目前已将 30% 的代码合并到代码库中 🎉
  - 我们的目标是在第 1 个完整的实现中支持 TCP

Pull Request at https://github.com/envoyproxy/envoy/pull/28824

接下来

- 社区
  - 继续合并其余代码
  - 有一些已知的问题需要解决
- 性能
  - 参数调整校对
  - SQ 轮询，固定缓冲区，零拷贝以及最新的内核中的新功能
- 替换调度器
  - 将 libevent 替换成 io_uring

自由提问

Notices and Disclaimers

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.
Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.
Your costs and results may vary.
Intel technologies may require enabled hardware, software or service activation.
© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.