### 网关高性能之道

###### 异步化的设计

* + 路由转发请求异步化：
    - 在我们网关内部转向下游服务的时候，用于提升吞吐量；
  + 接收服务响应异步化
    - 双异步模式好处：比较适合于下游服务性能不是很高的场景（500-2000ms），非常合适的；
    - 双异步模式缺点：下游服务性能很好（1ms - 3ms），频繁的上下文切换；
  + 插件（Fliter）异步化
    - 比如在插件中处理很耗时的操作（认证授权服务，调用第三方的PRC服务），用于提升吞吐量

###### 吞吐量为王：

* + 在某些特定的业务场景下：会有一些流量的洪峰突然瞬时打到我们的入口网关；
  + 我们需要再boss-work之后加一个缓冲区：
    - disruptor
    - mpmc

###### 线程数设定：

* + 网关服务完全是一个CPU密集型的服务类型：假设我们服务器是8C/16G
    - CPU密集型：core + 1 (N) = 8 ~ 8 + N
    - IO密集型： core / (1 - 阻塞系数[0.7-0.9]) = 80
  + CPU亲和性：用操作系统的CPU核，与一个线程做强绑定

###### 用尽缓存：

* + 在尽可能能用到缓存的地方，都使用缓存（内存：map、list、queue）

###### 串行化设计：

* + 在耗时很小，性能要求非常高的场景下：往往串行执行逻辑，会比并行执行效率更高；
  + 比较合适并行设计的场合：业务逻辑处理的时候，比如有远程RPC调用，很耗时的操作（任务没有依赖关系）