【微服务架构 1】

单体应用时代

在一个进程里面运行，不管是否做集群

规模大后，分库分表

好处：开发简单、集中管理、没有重复开发、不会有分布式的损耗

缺点：不好维护、升级困难、无法适应快速迭代

程序太庞大

技术的发展，都是业务推动的

微服务架构 Microservice Architecture

是一种程序架构设计的风格，把各种业务分离成单独的服务，在独立的进程运行，起到解耦的作用

跨进程通信

第三方存储(queue\db\nosql等)

特点：没有门槛，被动式的

网络协议通信（webapi\wcf）

特点：数据序列化彻底

.net remoting rpc

客户端、服务端都是.net平台，性能高

规范：微服务中的任何服务都要做集群，一台不靠谱，增加承载能力、避免单一故障

Nginx不适用于微服务，因为集群伸缩的时候，要求能够主动发现，而nginx无法做到，它需要重启才能发现；

另外nginx也无法做健康检查

Consul：替代nginx

负载均衡、服务注册与发现、健康检查

Polly：

降级：流量大，物理资源不够用时，降低或关闭某些低优先级的服务（双十一）

polly通过预设优先级，平时正常返回，特殊时期能自动停止低优先级的服务

限流：限制单位时间内的请求频率（秒杀）

缓存：配置某些服务的缓存

重试：调用服务可能失败，比如网络异常，服务正在重启时，就重试

超时：防止因为某个服务挂了，导致别的服务调用被堵塞

熔断：重试超过一定次数，在接下来某个时间内，不再去调用

Ocelot网关：

微服务中最重要的

Polly的功能可在这个环节实现

验证授权：

统一认证

Docker：快速部署

分布式事务