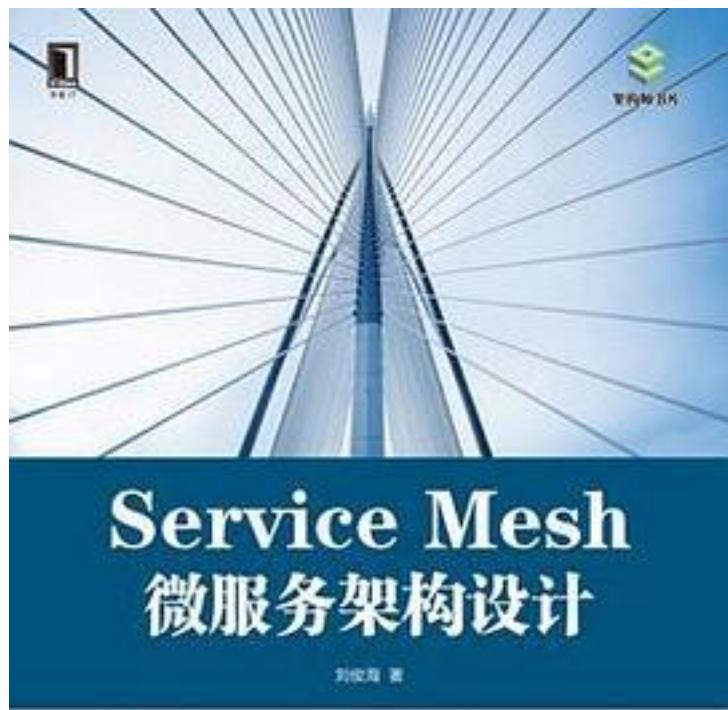


# Service Mesh微服务架构设计



ARCHITECTURE DESIGN OF SERVICE MESH AND MICRO SERVICES

追求架构清晰易懂，从设计与工程化视角剖析Service Mesh，深植为渠一线。  
实践真实的落地。  
涵盖微服务实践应用、Istio/Etcd的架构设计与实现、Service Mesh工程化设计思想与实践等。

机械工业出版社  
CHINA MACHINE PRESS

[Service Mesh微服务架构设计 下载链接1](#)

著者:刘俊海

出版者:机械工业出版社

出版时间:2019-9

装帧:平装

isbn:9787111636847

全书分为3部分：第一部分是基础篇，首先从微服务架构的挑战讲起，接下来剖析service mesh产生的背景，service mesh当前的现状以及主流的一些开源项目。第二部分是实战篇，深入讲解如何从零开

始构建一个生产环境可用的service

mesh系统，包含技术选型、架构设计和技术难度深入分析等。其中高性能、高可用、高扩展性方面的一些设计和考量都会深入阐述。第三部分是应用篇，实例分析service mesh对服务治理带来的便利和影响。

通过阅读本书，读者不仅能深入了解service  
mesh对微服务领域的影响，而且还可以了解service

mesh架构和设计的全过程，全书也包含高性能、高可用、高扩展性、服务治理等多个重要主题。

作者介绍：

刘俊海

好未来高级架构师，曾在滴滴、百度等知名互联网公司任职，超过8年C/C++开发和架构设计经验；精通服务框架和业务高可用技术，多年亿级流量环境下高并发和高可用实战经验，精通微服务架构和微服务基础设施，近期关注Service Mesh。

目录: 前言

第一篇 基础篇

第1章 微服务架构	2
1.1 为什么需要微服务	2
1.1.1 传统单体服务的问题	2
1.1.2 微服务的定义	3
1.1.3 微服务与康威定律	3
1.1.4 微服务的收益	4
1.2 微服务架构的挑战	4
1.2.1 服务拆分	4
1.2.2 开发挑战	5
1.2.3 测试挑战	5
1.2.4 运维挑战	6
1.3 微服务化的具体时机	6
1.4 微服务化开展前的准备工作	8
1.4.1 微服务开发框架	8
1.4.2 微服务标准化	15
1.4.3 持续集成与发布	17
1.5 微服务实施	17
1.5.1 微服务拆分	17
1.5.2 微服务通信	19
1.5.3 微服务稳定性保障	20
1.6 本章小结	25
第2章 微服务治理	26
2.1 微服务治理基础	26
2.1.1 服务治理由来	26
2.1.2 服务治理的目标与愿景	27
2.1.3 服务治理的工作范畴	28
2.1.4 服务治理闭环体系	29
2.2 正向服务治理	29
2.2.1 效率治理	30
2.2.2 稳定性治理	31
2.3 效果治理	34
2.4 可见可观测	35

2.4.1 服务可见性	35
2.4.2 变更可见性	36
2.4.3 可观测性	36
2.5 量化分析体系	41
2.5.1 稳定性风险度量	41
2.5.2 基于多维度监控的故障定位	42
2.5.3 风险分析	43
2.6 线上治理	43
2.6.1 线上预案体系	43
2.6.2 基于Metric的预案自动触发	44
2.6.3 治理参数动态调整	44
2.7 线下治理	47
2.7.1 链路稳定性治理	47
2.7.2 架构与资源治理	50
2.8 服务治理演进	50
2.8.1 远程Proxy方式	51
2.8.2 基于智能客户端的服务框架	52
2.8.3 本地Proxy	52
2.9 理想的服务治理架构	53
2.10 本章小结	54
第3章 下一代微服务框架Service Mesh概要	55
3.1 Service Mesh基础	55
3.1.1 什么是Service Mesh	55
3.1.2 Service Mesh的基本模式	56
3.2 Service Mesh的发展历程	58
3.3 Service Mesh项目Linkerd	60
3.3.1 Linkerd演进	60
3.3.2 Linkerd路由机制	62
3.3.3 Linkerd 2.0核心架构	63
3.4 Service Mesh项目Istio	64
3.4.1 Envoy	64
3.4.2 Istio	66
3.5 Service Mesh其他解决方案	67
3.5.1 国外其他Service Mesh项目	67
3.5.2 Service Mesh在中国的发展	68
3.6 Service Mesh云上产品	69
3.6.1 AWS App Mesh	69
3.6.2 Azure Service Fabric Mesh	69
3.6.3 Google Cloud Service Mesh	70
3.6.4 SuperGloo	70
3.7 Service Mesh标准化	71
3.8 本章小结	71
第二篇 架构篇	
第4章 Envoy架构剖析	74
4.1 Envoy整体架构	74
4.1.1 关键设计约束	74
4.1.2 设计原则	75
4.1.3 整体架构	76
4.2 Envoy网络模型	78
4.2.1 Envoy事件调度模型	78
4.2.2 Envoy线程模型	81
4.2.3 线程本地存储机制	81
4.3 Envoy扩展模型	84
4.3.1 插件扩展机制	84

4.3.2 网络相关插件	86
4.3.3 其他扩展插件	88
4.4 Envoy数据平面API	88
4.4.1 XDS协议语义	88
4.4.2 XDS协议通信	90
4.5 Envoy启动管理	91
4.5.1 正常启动	92
4.5.2 热重启	94
4.6 Envoy与Nginx架构层面的对比	95
4.6.1 功能与定位	96
4.6.2 网络模型	96
4.6.3 连接处理	97
4.6.4 插件机制	98
4.6.5 配置管理	99
4.6.6 内存管理	99
4.6.7 部署与运维	100
4.6.8 观测与诊断	100
4.7 本章小结	100
第5章 Istio架构剖析	101
5.1 Istio整体架构	101
5.1.1 数据平面组件	102
5.1.2 控制平面组件	103
5.2 Istio的Kubernetes基础	104
5.2.1 Kubernetes综述	104
5.2.2 Kubernetes网络访问模型	107
5.2.3 Kubernetes API管理	110
5.2.4 Istio与Kubernetes的相互关系	111
5.3 Istio流量控制模型	112
5.3.1 流量管理API	112
5.3.2 Istio Mesh模型	116
5.4 Mixer模型	118
5.4.1 Mixer基本概念	119
5.4.2 Mixer通用配置模型	119
5.4.3 Mixer架构演进以及对性能的影响	121
5.5 Istio安全	122
5.5.1 Istio安全基础	122
5.5.2 Istio认证架构	123
5.6 Istio配置处理框架	124
5.6.1 配置验证	125
5.6.2 配置变更处理和分发	125
5.7 本章小结	125
第6章 Istio控制流设计	126
6.1 Envoy生命周期管理	126
6.1.1 Envoy注入	126
6.1.2 Envoy启动管理	128
6.1.3 Envoy配置和运行状态监控	131
6.2 Istio配置变更管理	133
6.2.1 通用模型和机制	133
6.2.2 Kubernetes具体实现	137
6.3 控制平面和数据平面的XDS交互	138
6.3.1 控制平面的gRPC Server启动	139
6.3.2 Envoy的XDS请求	140
6.3.3 Istio XDS配置下发	140
6.3.4 Envoy的XDS消息接收	143

6.4 XDS配置生成	143
6.4.1 可见性	143
6.4.2 配置生成机制	145
6.4.3 XDS配置生成实现	147
6.5 XDS配置的Envoy处理	149
6.5.1 XDS配置变更的判断	149
6.5.2 CDS配置的延迟处理	150
6.5.3 集群和节点配置处理	152
6.5.4 路由配置处理	153
6.5.5 监听器配置处理	153
6.6 本章小结	155
第7章 Istio数据流设计	156
7.1 Iptables	156
7.1.1 Iptables的基本原理	156
7.1.2 Iptables在Istio中的使用	158
7.2 监听管理	158
7.2.1 监听器建立	158
7.2.2 监听器和工作线程绑定	159
7.3 连接管理	160
7.3.1 监听器匹配	160
7.3.2 协议过滤器匹配	161
7.3.3 创建新连接	161
7.4 网络I/O和缓冲区管理	162
7.4.1 传输层数据读取	162
7.4.2 插件处理	163
7.5 Thrift协议处理	164
7.5.1 Thrift插件的整体架构	164
7.5.2 协议解析	165
7.5.3 协议相关的插件机制	166
7.6 HTTP请求处理	168
7.6.1 HTTP请求处理流程	168
7.6.2 协议解析	169
7.6.3 路由管理	171
7.6.4 HTTP过滤链处理	174
7.6.5 负载均衡	176
7.6.6 连接池实现	179
7.7 本章小结	182
第8章 Istio微服务治理	183
8.1 链路稳定性治理	183
8.1.1 超时机制	183
8.1.2 重试机制和重试策略	185
8.1.3 节点熔断和健康检查机制	188
8.1.4 资源限制机制	189
8.1.5 全局限流机制	190
8.2 链路可观测性	190
8.2.1 Envoy分布式跟踪支持	190
8.2.2 Envoy Metric支持	194
8.2.3 Envoy Log支持	198
8.3 本章小结	200
第9章 Service Mesh架构的工程化设计	201
9.1 复用和解耦	201
9.2 架构扩展机制	203
9.2.1 服务注册中心插件机制	203
9.2.2 Envoy Filter插件机制	203

9.3 性能设计	204
9.3.1 基于TLS的无锁设计	204
9.3.2 多级缓存机制	205
9.3.3 批量更新机制	205
9.4 架构设计的权衡	206
9.5 API和SDK设计	207
9.5.1 声明式API设计	207
9.5.2 代码自动生成机制	207
9.6 配置管理	208
9.6.1 基于Protobuf 3的配置Scheme描述	208
9.6.2 配置动态加载机制	210
9.7 本章小结	210
第10章 Service Mesh与云原生架构	211
10.1 Service Mesh和Serverless	211
10.1.1 Serverless基础	211
10.1.2 Knative	213
10.2 东西向和南北向通信的统一	215
10.3 云原生时代的Service Mesh	216
10.4 Service Mesh现状和展望	217
10.5 本章小结	218
附录 Service Mesh迁移的要点与原则	219
• • • • (收起)	

[Service Mesh微服务架构设计 下载链接1](#)

## 标签

mesh

架构

编程

service

软件开发

计算机

架构设计

微服务

## 评论

年纪大了， service mesh 这都是啥...

---

初读，只会了解下Service mesh的大体情况

---

一本说明书

---

基础设施总是很难绕开数据平面、控制平面、管理平面。应用需要将业务需求和非业务需求解耦，业务只需要关注业务相关的功能需求，开发、测试和运维的效率才能大大提高。

---

对于微服务和治理这块，体系和方法论的总结很不错，关于 envoy 和 istio 的细节部分就源码来凑吧，多打的一颗星给体系和方法论

---

第一部分的微服务和微服务治理讲的挺好的，微服务涉及的方方面面都提到，还算比较影响深刻吧。后面的部分由于时间有限，粗略一翻。总体一般吧。

---

K8s+service mesh，进一步提高了资源弹性伸缩的效率，对开发进一步屏蔽了底层的细节，更加高效，缺点就是又要学习了~

---

[Service Mesh微服务架构设计 下载链接1](#)

## 书评

---

[Service Mesh微服务架构设计 下载链接1](#)