Service Mesh Meetup #7　成都站

Service Mesh是下一代SDN吗？

从通信的角度看Service Mesh的发展

赵化冰

中兴通讯 软件专家/Istio Committer

2019.10.26

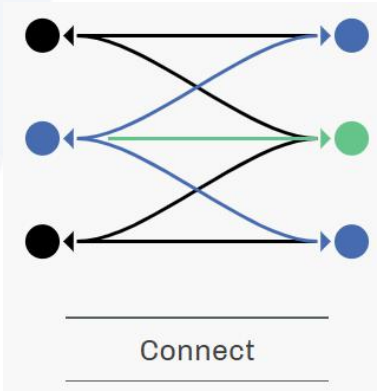
CLOUD NATIVE COMPUTING FOUNDATION　蚂蚁金服 ANT FINANCIAL　ServiceMesher

什么是Service Mesh？ - by Willian Morgan(Buoyant)

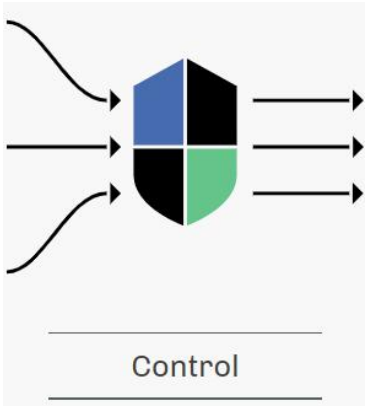
A service mesh is a dedicated infrastructure layer for handling service-to-service communication. It’s responsible for the reliable delivery of requests through the complex topology of services that comprise a modern, cloud native application. In practice, the service mesh is typically implemented as an array of lightweight network proxies that are deployed alongside application code, without the application needing to be aware.

服务网格是一个**基础设施层**，用于处理服务间通信。云原生应用有着复杂的服务拓 扑，服务网格负责在这些拓扑中**实现请求的可靠传递**。在实践中，服务网格通常实 现为一组**轻量级网络代理**，它们与应用程序一起部署，但**对应用程序透明**。

什么是Service Mesh？ - by Istio

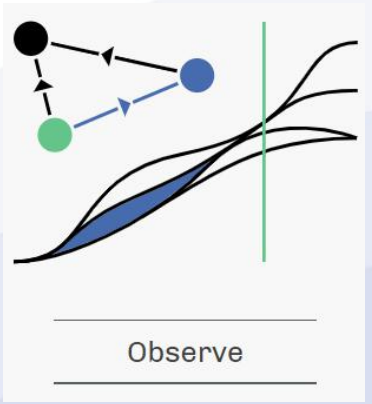


服务发现
负载均衡
流量控制
...



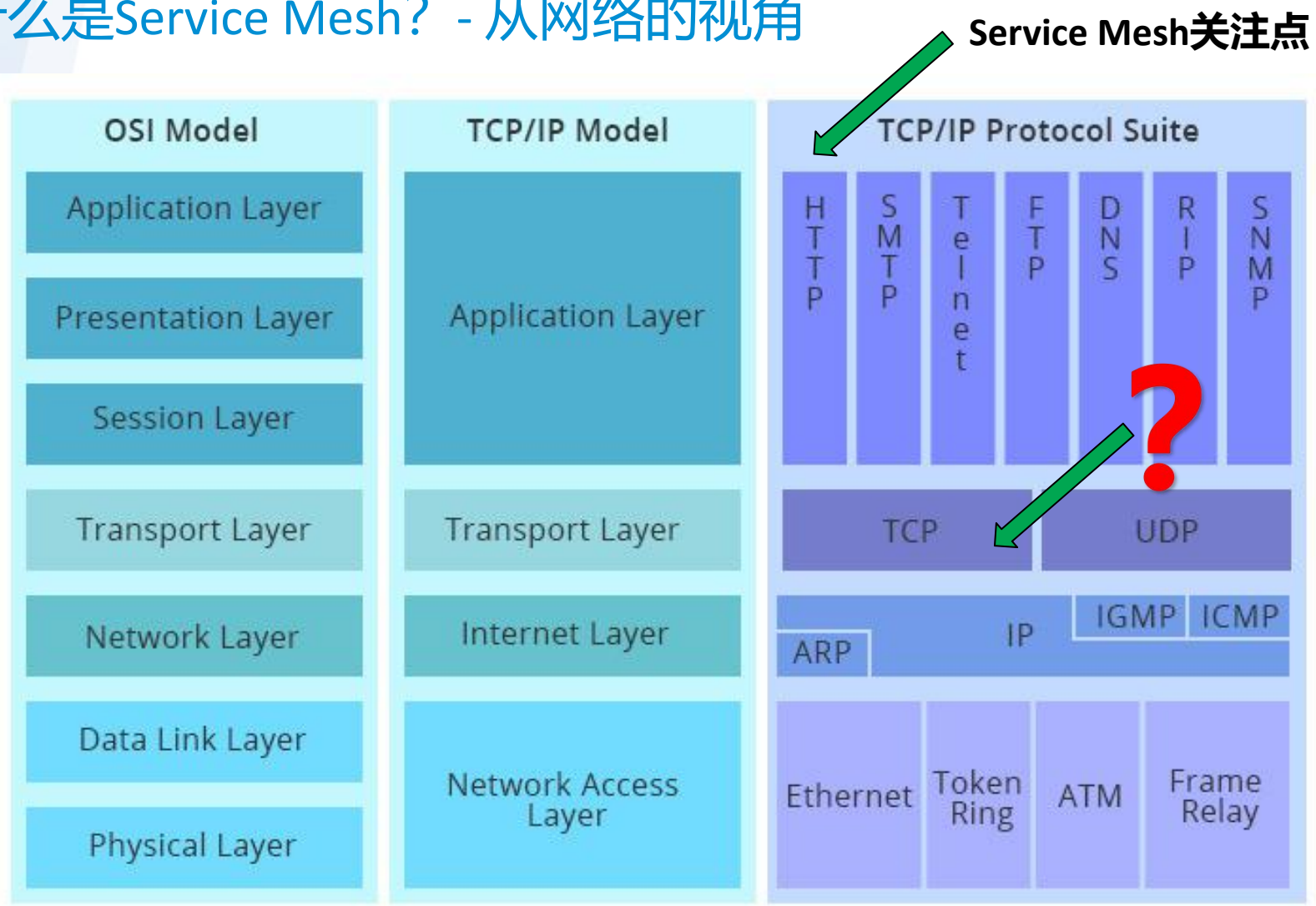
黑白名单
限流
...

身份认证
通信加密
权限控制
...



调用追踪
指标收集
...

什么是Service Mesh？ - 从网络的视角



网络视角：

Service Mesh是一个主要针对七层的网络解决方案，解决的是服务间的连通问题

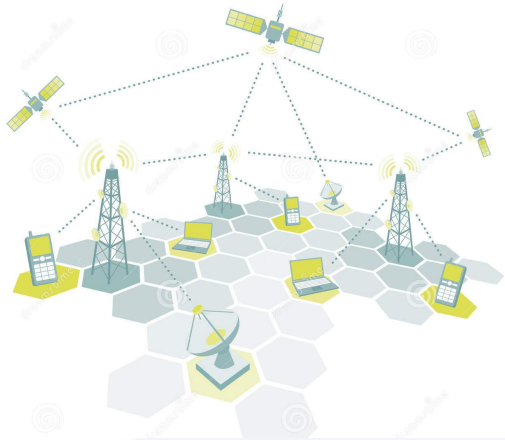
Service Mesh是下一代的SDN吗?

通信网络和微服务系统面临类似的问题:

通信网络

- 互不兼容的专有设备
- 基于IP的通信缺乏质量保证
- 低效的业务部署和配置

...



微服务系统

- 互不兼容的代码库
- 不可靠的远程方法调用
- 低效的服务运维

...

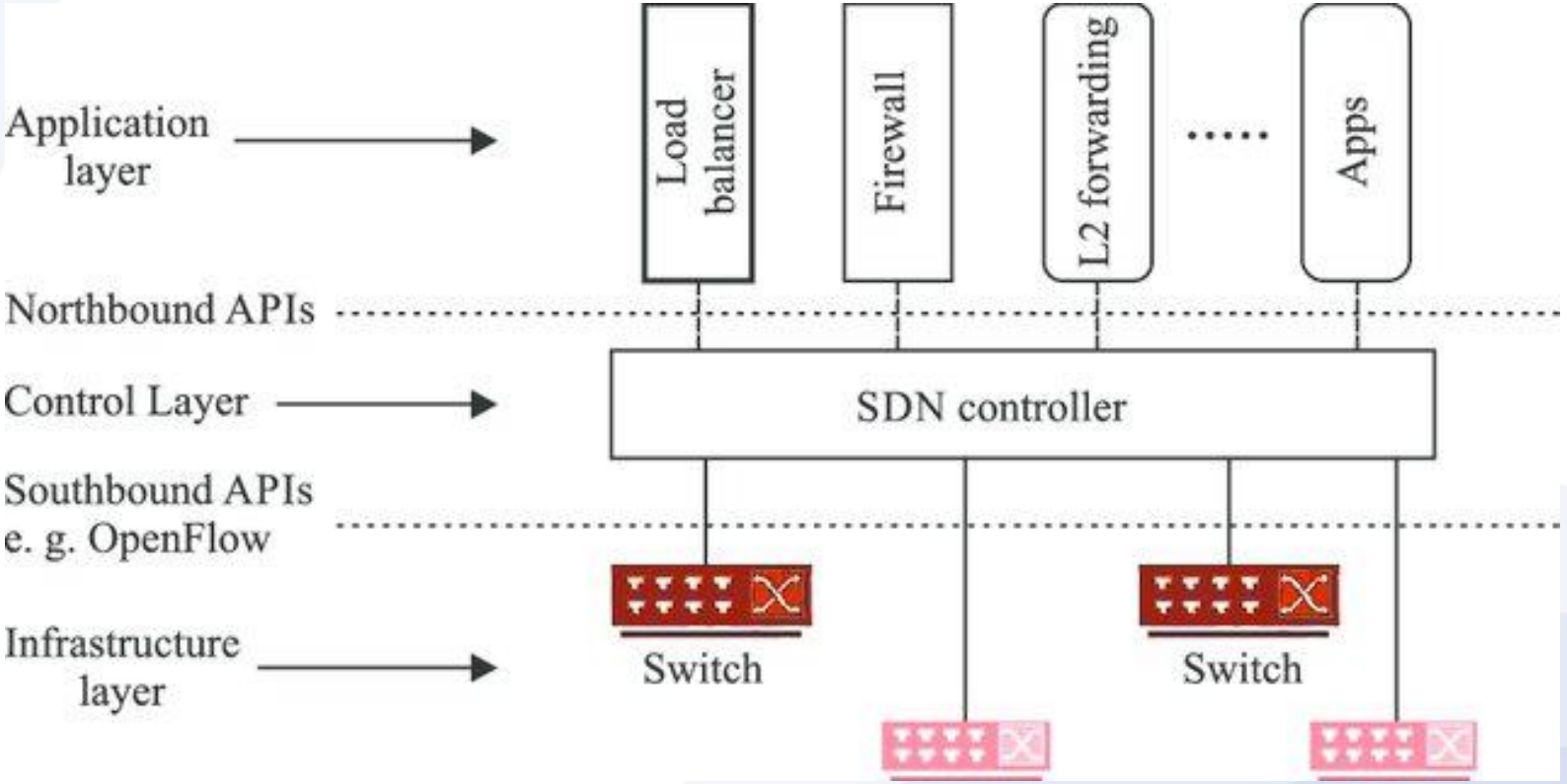
Service Mesh是下一代的SDN吗?

SDN：　　主要关注1到4层
Service Mesh：　主要关注4到7层

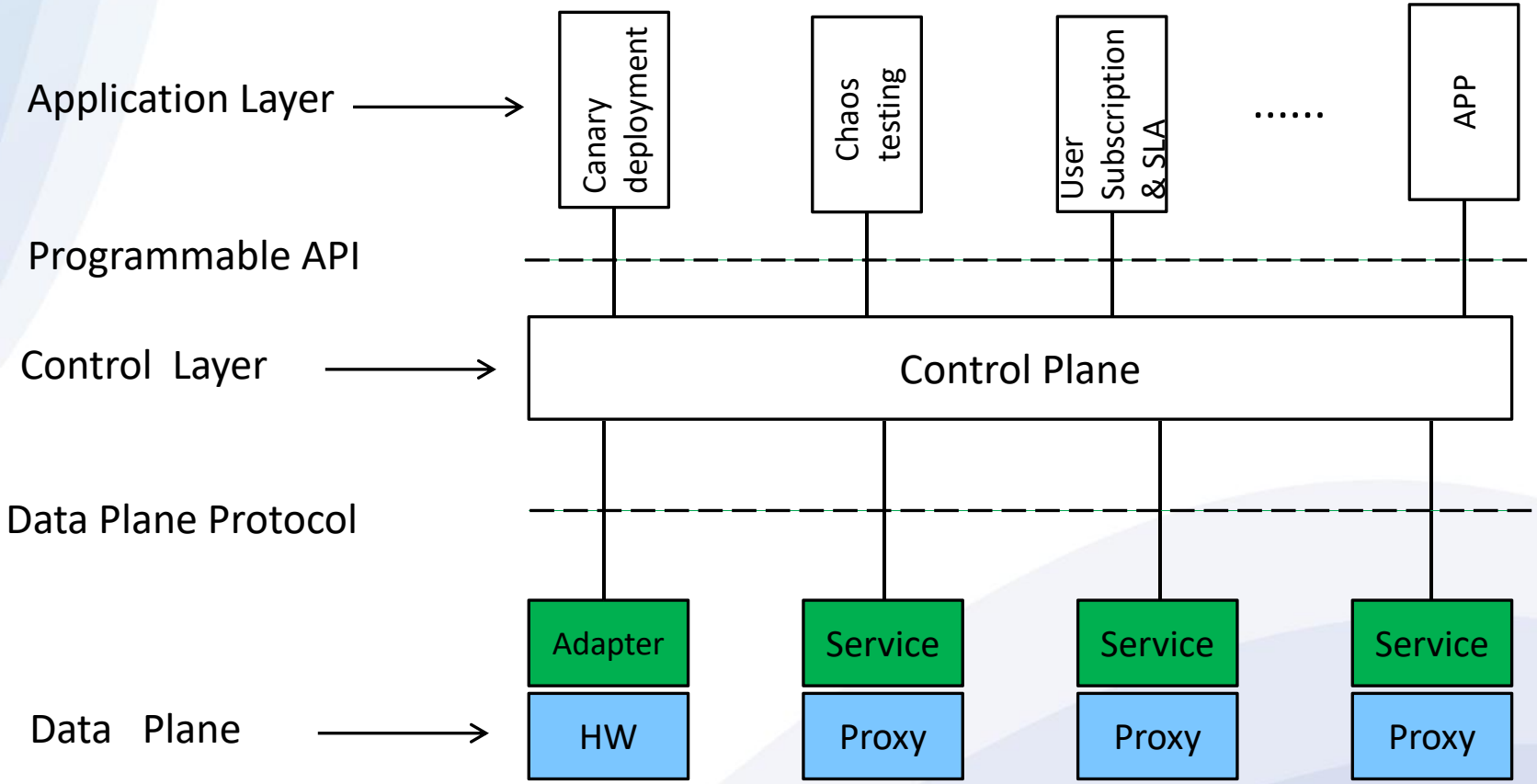
Network Layer	Look at	Solution
Layer 7+	Message Body	Service Mesh
Layer 7	HTTP Header	Service Mesh
Layer 4	TCP Port	SDN/Service Mesh?
Layer 3	IP Address/Protocol	SDN
Layer 2	MAC/VLAN	SDN
Layer 1	Input Port	SDN

类似的问题，不同的网络层次，通信网络的解决方案能为Service Mesh提供哪些借鉴?

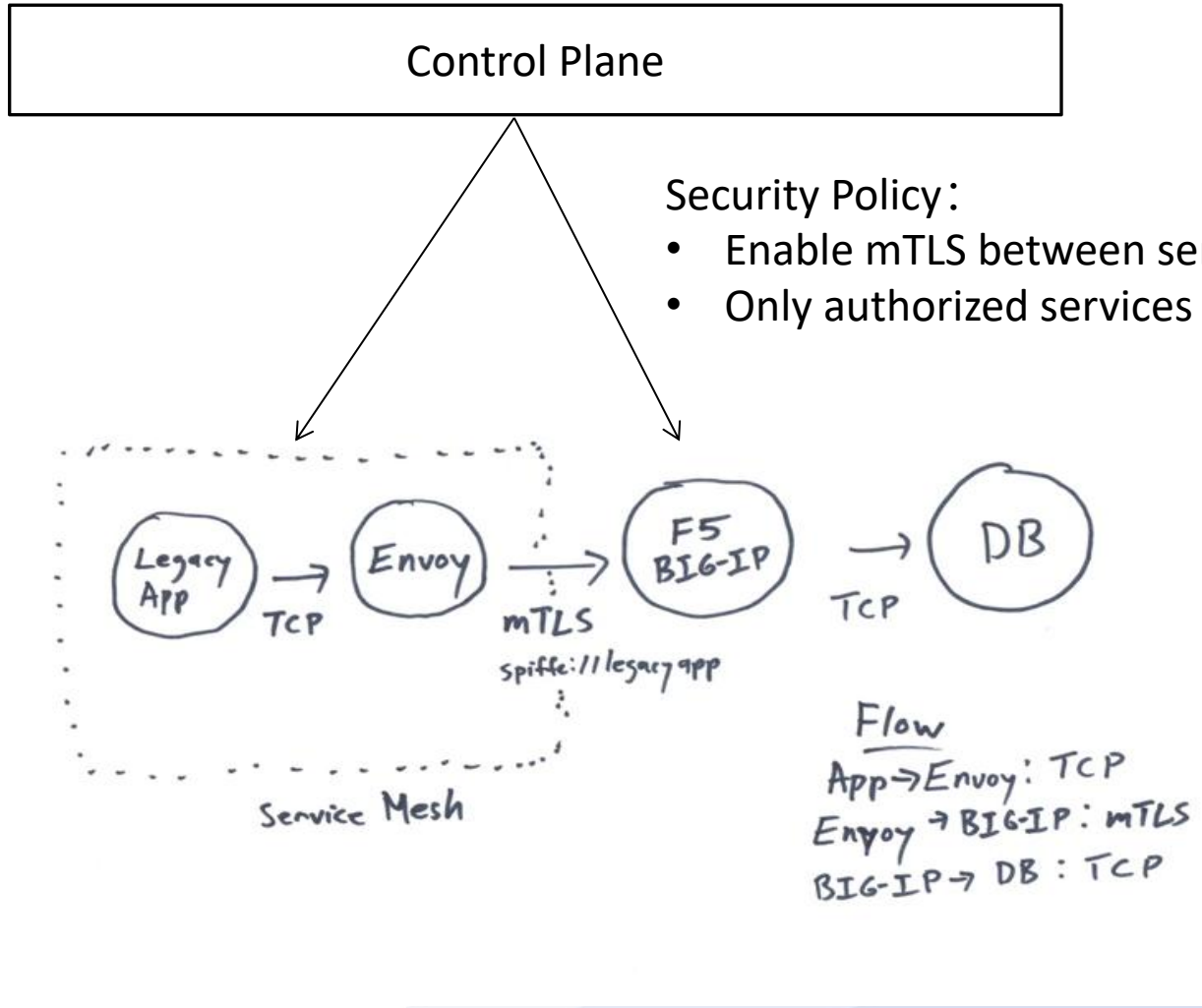
通信网络的解决方案：SDN



微服务应用层通信的解决方案-Service Mesh

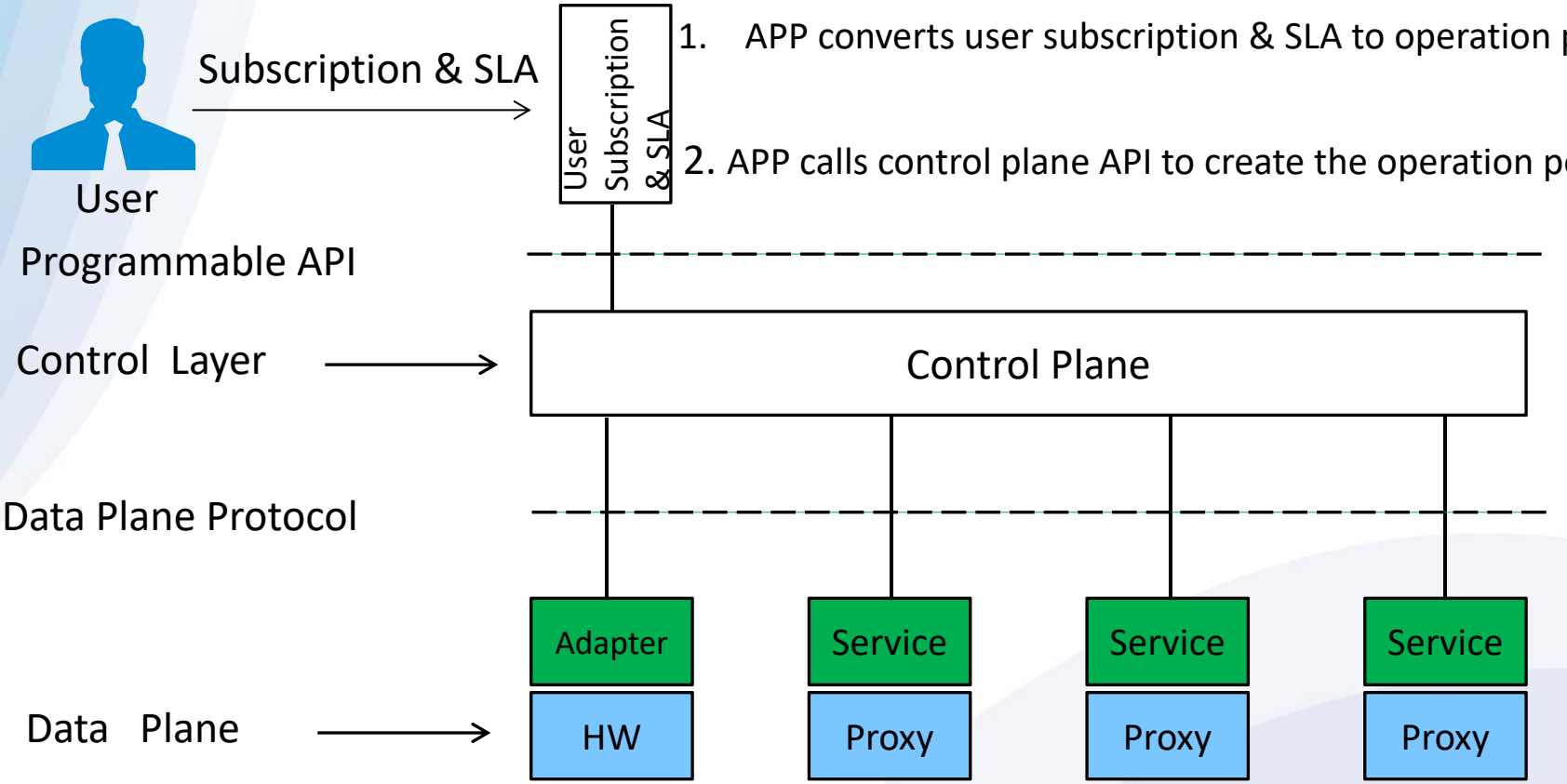


通过Service Mesh控制面统一管理F5和Envoy



https://aspenmesh.io/2019/03/expanding-service-mesh-without-envoy/

App Example: User Subscription and SLA Management



3. Control plane distributes the operation policy to data plane via data plane protocol

4. Data plane follows the rules to handle user traffic(For example: Rate limiting, Service priority, etc.)

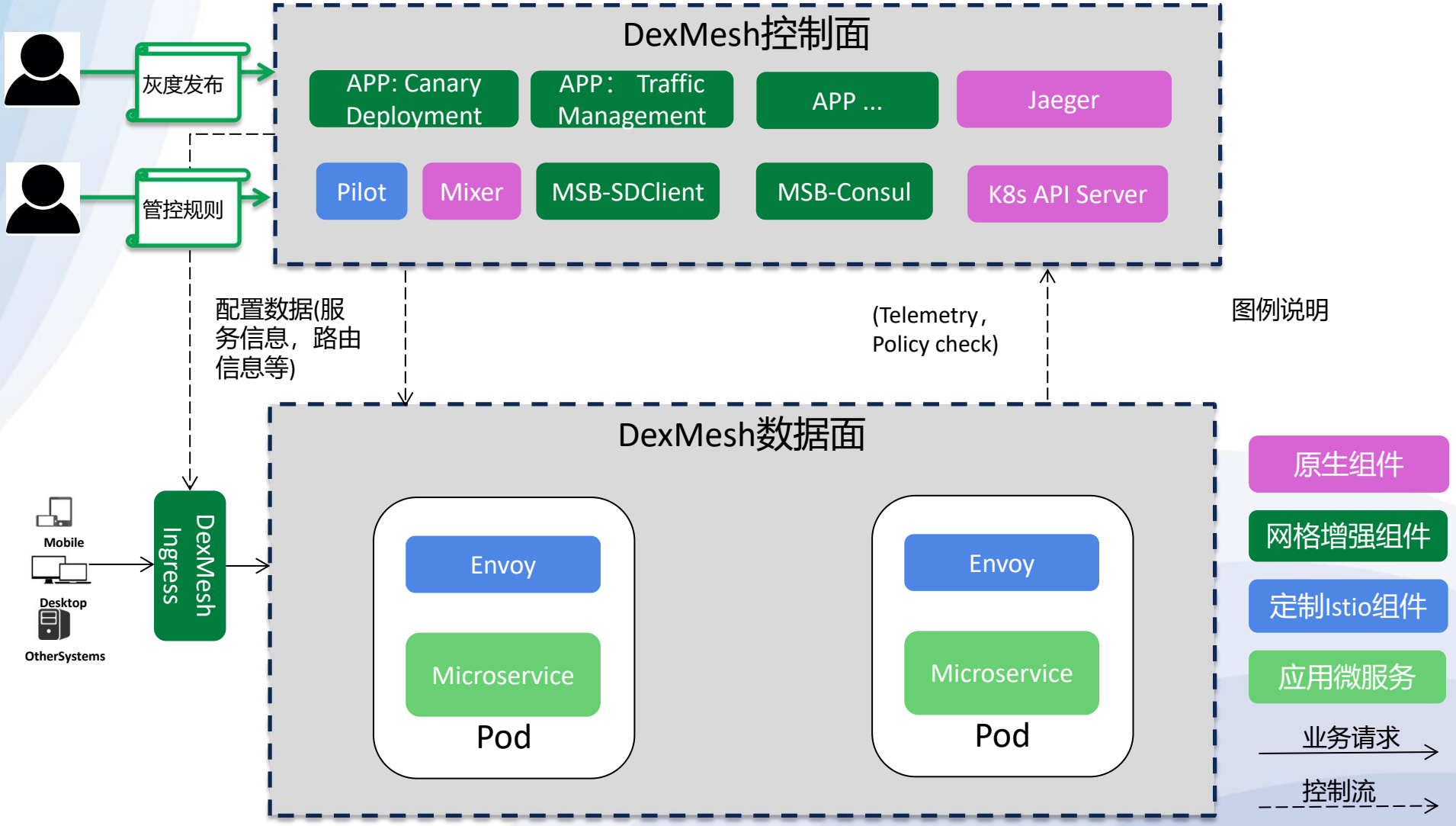
总结：他山之石，可以攻玉

- 解决类似的问题：运维和通信的问题
- 相似的解决方案：数据面+控制面+应用
- 不同的协议层次：SDN 2-4层，Service Mesh 主要为7层

SDN对Service Mesh发展的启发：

- **北向接口**
  - 面向业务和运维
  - 具有较高的抽象层次，比较容易提取统一的控制面标准？
  - 主要面向layer 7及以上？
  - SMI能否统一控制面标准？如何避免成为最小公分母，扩展支持其它协议？
- **南向接口**
  - 面向流量和路由配置
  - xDS v2将统一数据面标准？
  - xDS接口包含有较多实现相关内容：Listener, Filter, 能否可以成为一个通用的接口协议？是否会出现Envoy之外的大量数据面实现？
  - 建议：对xDS接口进行改进，去掉实现相关内容
- **Service Mesh的发展**
  - 控制面对数据面软硬件的统一控制能力？
  - 通过控制面API接入各种丰富的应用场景 - 下一个热点？

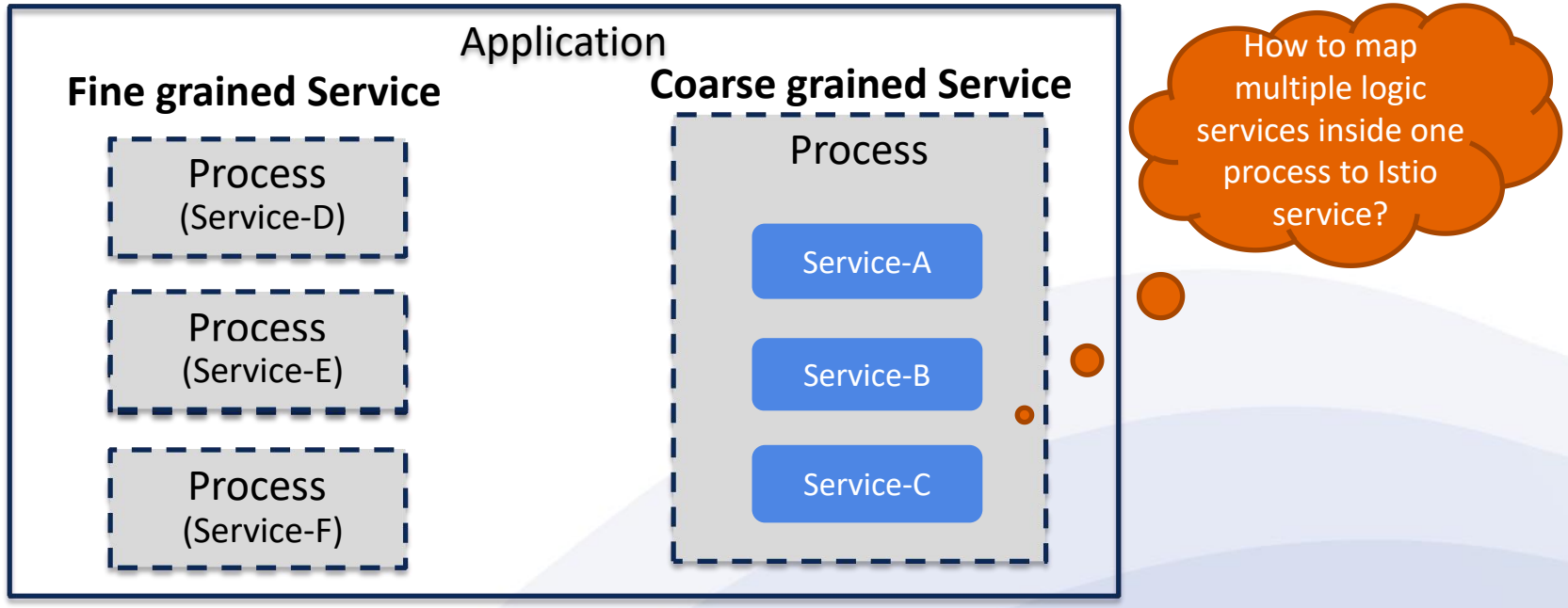
总体架构-高层视图



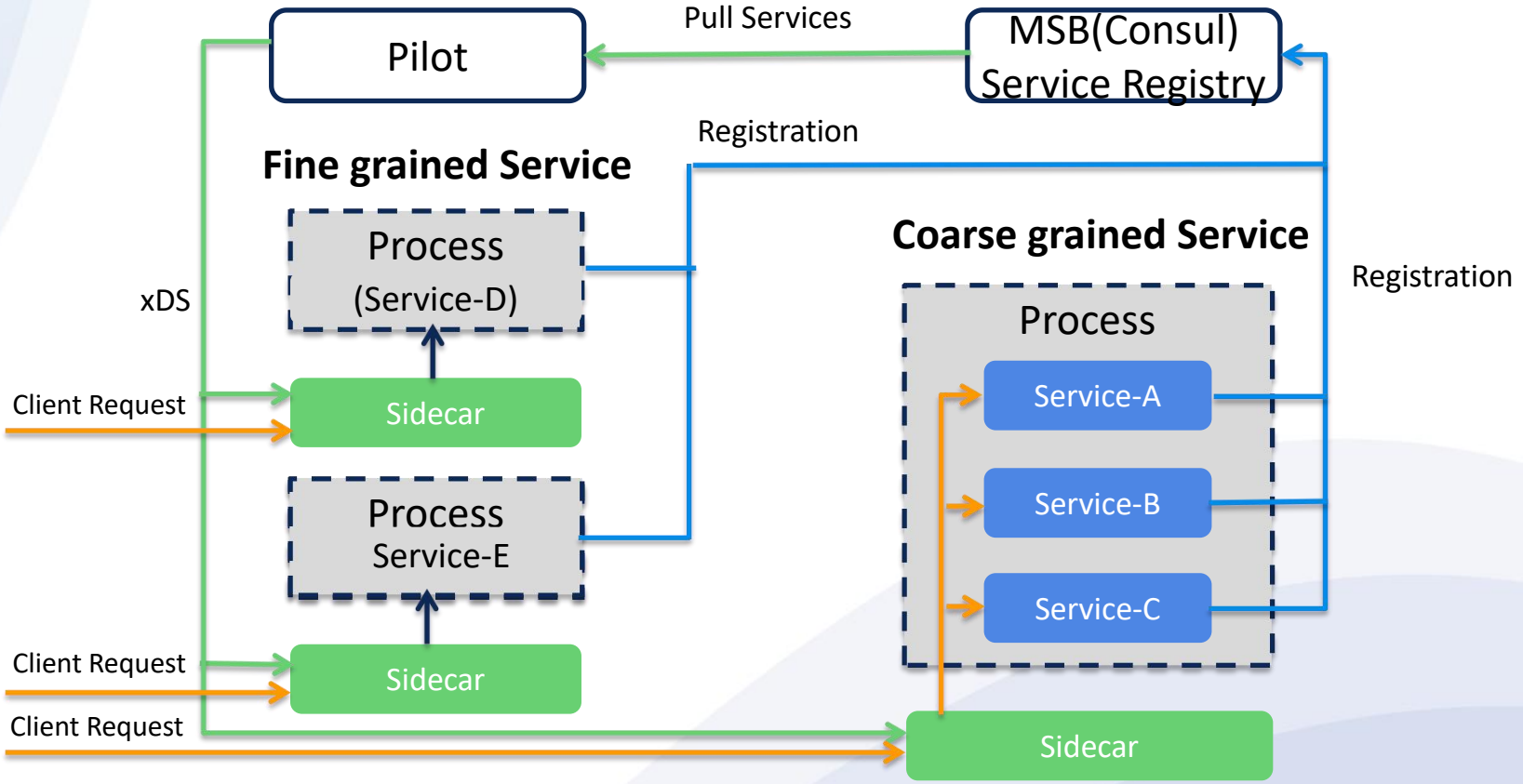
产品化增强-服务注册发现

出于历史原因，我们使用了Kubernetes来部署服务，但并未使用Kubernetes内置的服务发现功能。

An example application: Combination of “fine Grained” and “Coarse Grained“ Services



产品化增强-兼容粗粒度的SOA类应用



Consul Registry遇到的坑：

1.0版本的Consul Registry只能算PoC（原型验证），远未达到产品要求

CPU占用率超高不下 (Pilot+Consul 占用冲高到 400%)

- TIME_WAIT Sockets 太多导致FD耗光

Consul Registry优化

- 增加数据缓存，减少无谓的Consul Catalog API调用
- 将Polling改为Watch，大幅降低Consul服务数据变化后的同步时延

优化效果

- 200个服务的规模下，CPU占用率降低了一个数量级
- 服务数据变化同步时延从分钟级降低到秒级
- Consul调用导致的TIME_WAIT Sockets数量减少到个位级

产品化增强-Ingress API Gateway

K8S Ingress

Load balancing
SSL termination
Virtual hosting

提供七层网关能力，但和服务网格是割裂的

Istio Gateway

Load balancing
SSL termination
Virtual hosting
Advanced traffic routing
Fault injection
Other benifits brought by Istio

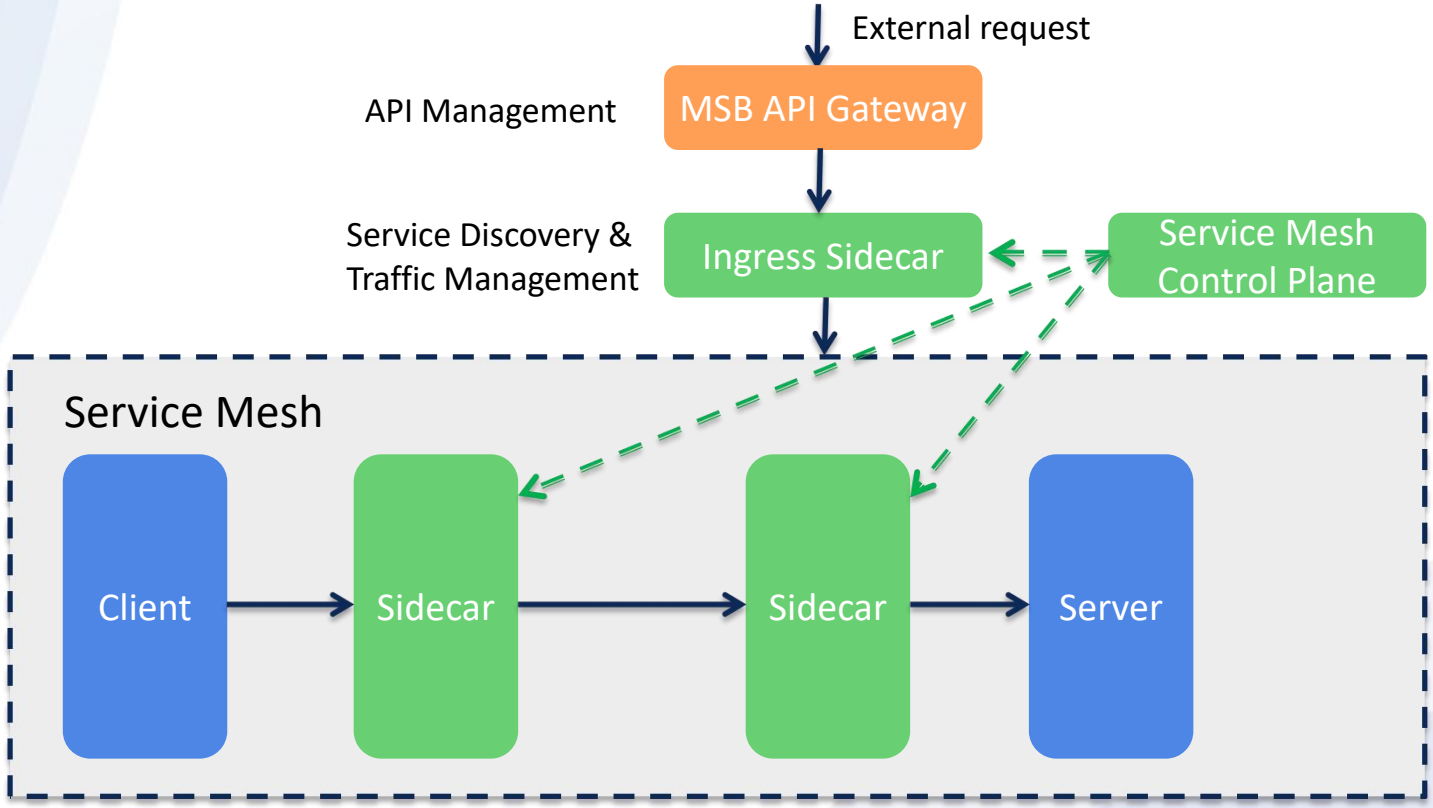
提供七层网关和网格能力，但缺少API管理能力

API Gateway

Load balancing
SSL termination
Virtual hosting
Traffic routing
API lifecycle management
API access monitoring
API access authorization
API key management
API Billing and Rate limiting
Other business logic ...

提供API管理能力，缺少服务网格能力

在DexMesh场景下Mesh和API Gateway的分工与协同



API Gateway：应用网关逻辑

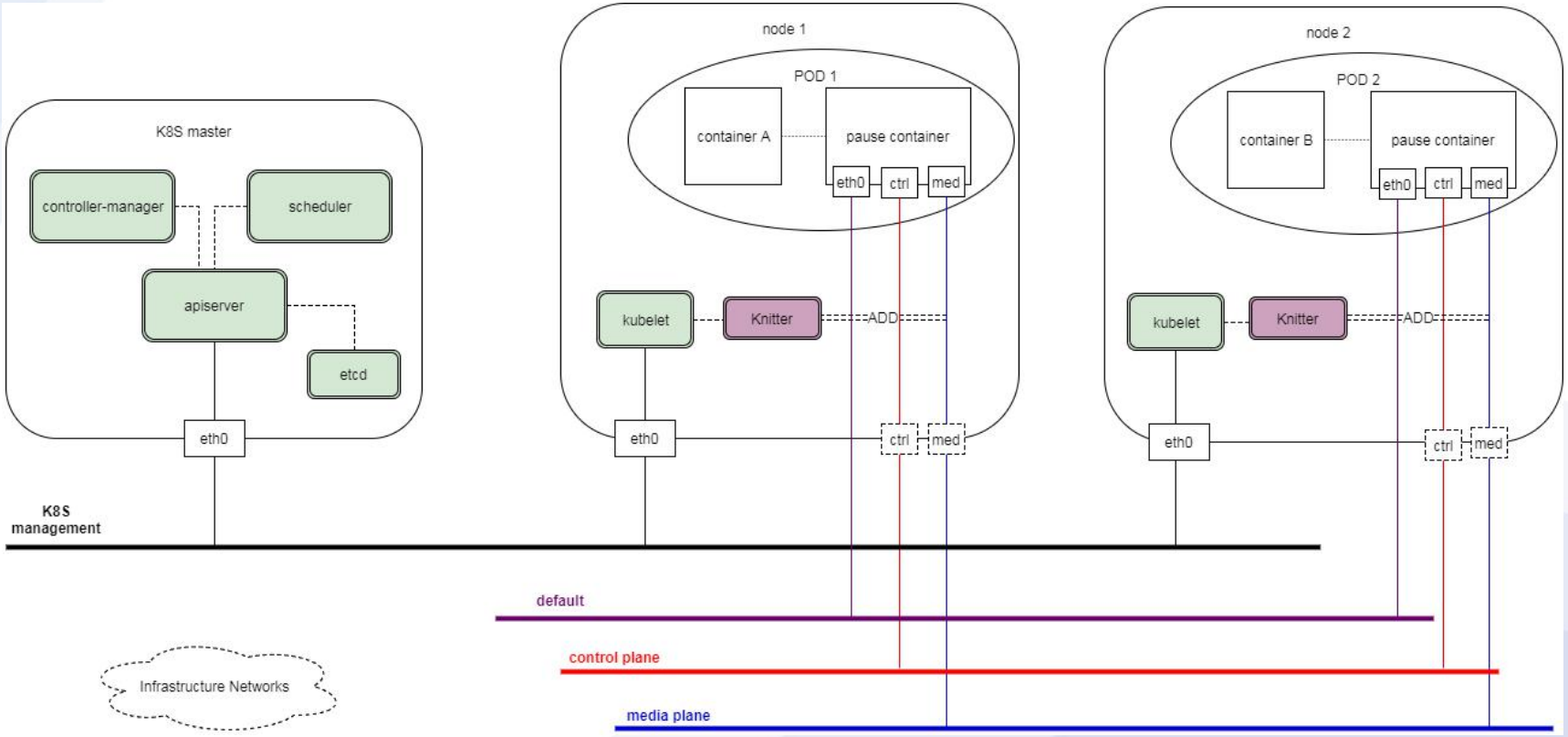
- 使用不同端口为不同租户提供访问入口
- 租户间的隔离和访问控制
- 用户层面的访问控制
- 按用户的API访问限流
- API访问日志和计费

Service Mesh：统一的微服务通信管理

- 服务发现
- 负载均衡
- 重试，断路器
- 故障注入
- 分布式调用跟踪
- Metrics 收集

产品化增强-支持多网络平面

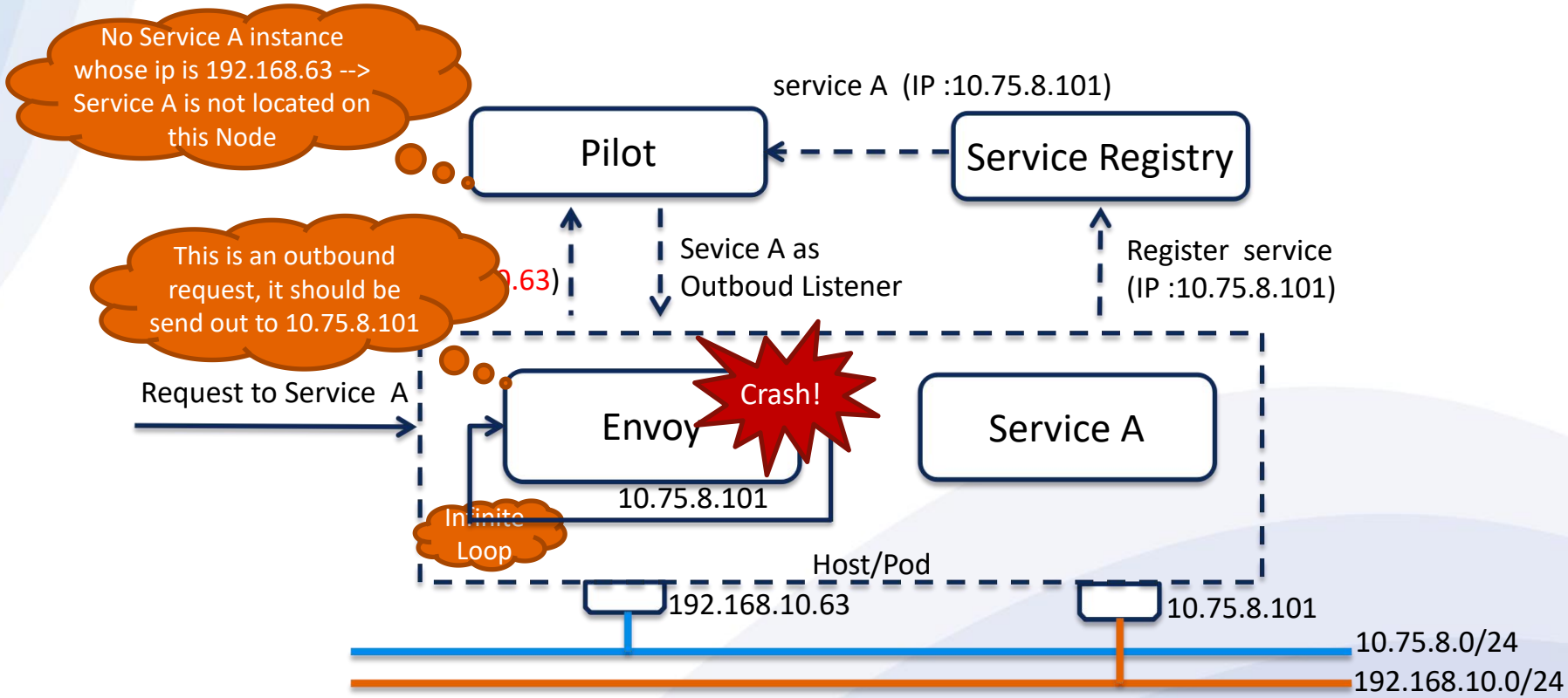
电信系统一般会有多个网络平面的，主要原因包括：**避免不同功能的网络之间的相互影响； 网络设计冗余，增强系统网络的健壮性； 为不同的网络提供不同的SLA；通过网络隔离提高安全性；通过叠加多个网络增加系统带宽**



上图中的Kubernets集群使用了Knitter网络插件，部署了四个网络平面

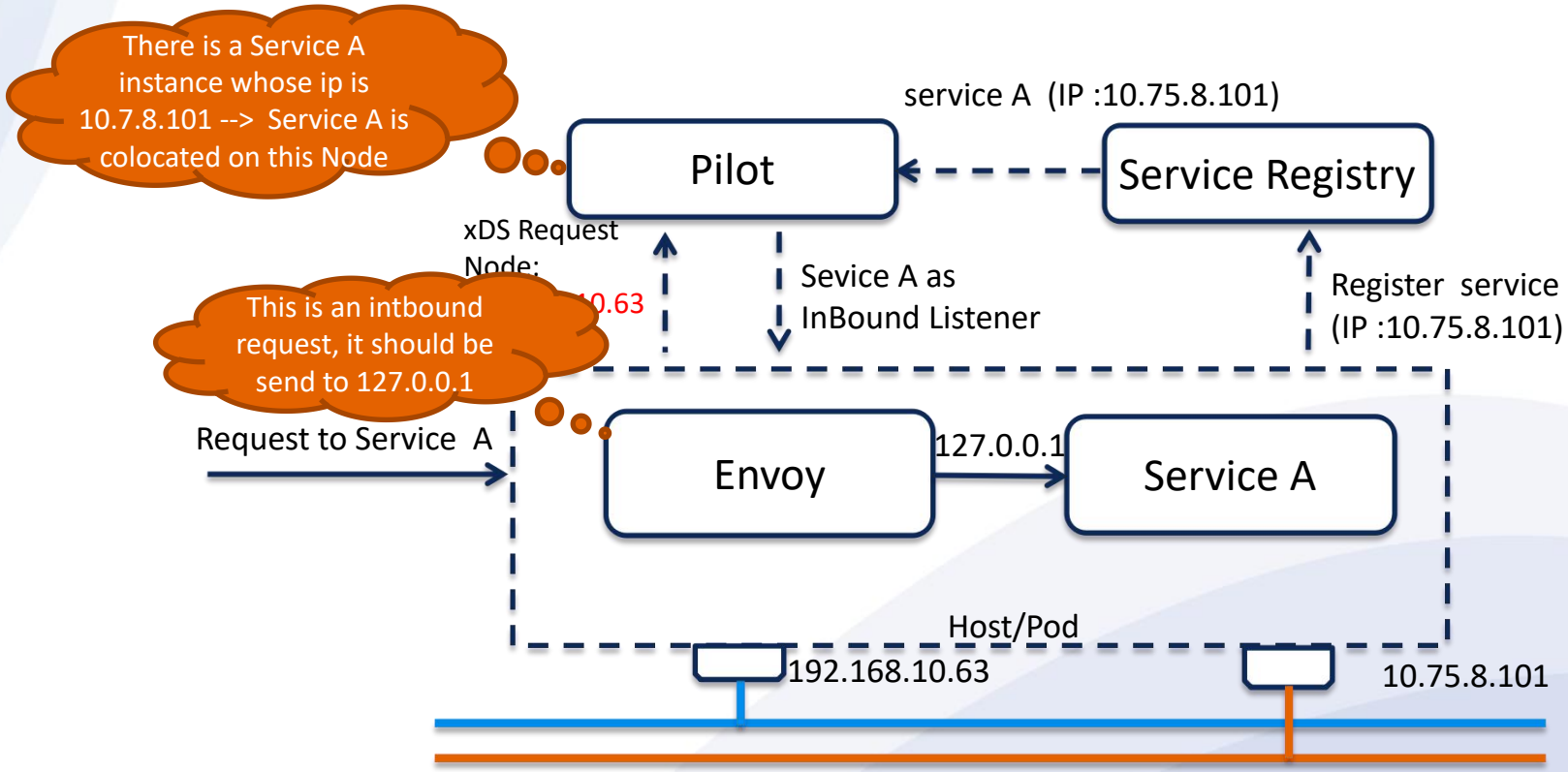
产品化增强-支持多网络平面

Istio1.0中不支持多网络平面，当服务地址和Envoy地址分别位于两个网络上时，会导致转发请求时发生死循环，导致socket耗尽，Envoy不停重启。



产品化增强-支持多网络平面

我们对Istio的代码进行了改造，增加了多网络平面支持。

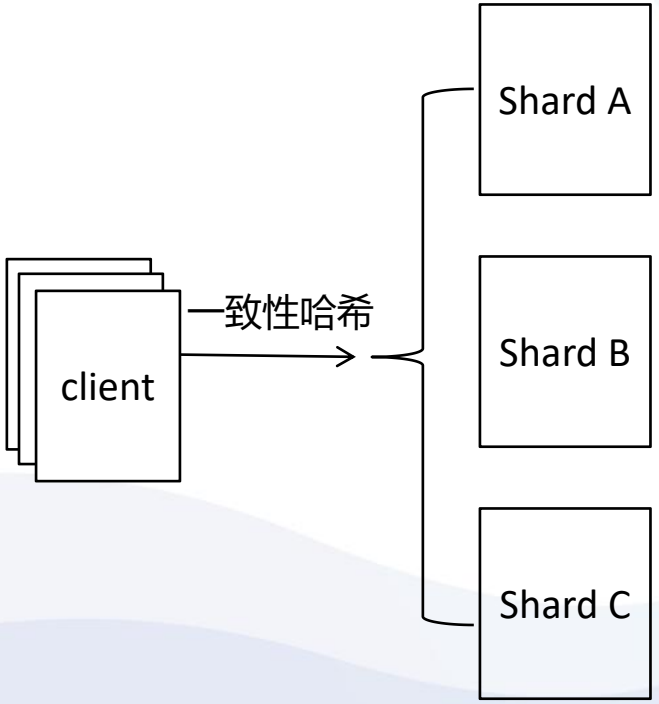


产品化增强-TCP Service的处理

是否需要将TCP Service纳入Service Mesh管控?

- **收益**
  - TCP Service可以享受流量管理，可见性，策略控制等Istio承诺的益处

- **成本**
  - Istio不理解TCP上的应用层协议，其对TCP Service的缺省处理会影响应用层逻辑
    -例子： Envoy的LB算法不能处理应用后端集群的Sharding
  - Istio中和HTTP Service 端口冲突会的TCP Service请求会被Envoy直接丢弃
    - 要求对应用进行改造，避免端口冲突



建议

- 将TCP纳入Service Mesh管控还不成熟，成本远大于收益
- Service Mesh应主要关注L7，而不是L4

产品化增强-TCP Service的处理

在Service Mesh中 Bypass TCP流量，让TCP请求跳过Service Mesh的处理，缺省发送到原始请求目的地。

- 方案一：通过IPtables bypass TCP流量
  通过IP段或者端口范围区分HTTP和其他TCP流量
  - 需要对应用进行改造

- 方案二：在Envoy中 bypass TCP 流量
  -不需要对应用进行改造，但Envoy要具备区分TCP和HTTP流量的能力，需对Envoy进行改造

改造方案：

- Envoy侧：通过一个自定义的envoy listener filter区分HTTP和非HTTP的TCP流量

- Pilot侧：修改Pilot下发的LDS配置，将TCP流量转到一个指定的filter chain 处理，通过tcp_proxy filter将TCP请求发往Passthroughfilter，以达到bypass TCP流量的目的。

产品化增强-TCP Service的处理-Istio 1.3中的改进

Istio 1.3改进：提供了HTTP Inspector，并且可以支持按照协议对filterchain进行Match

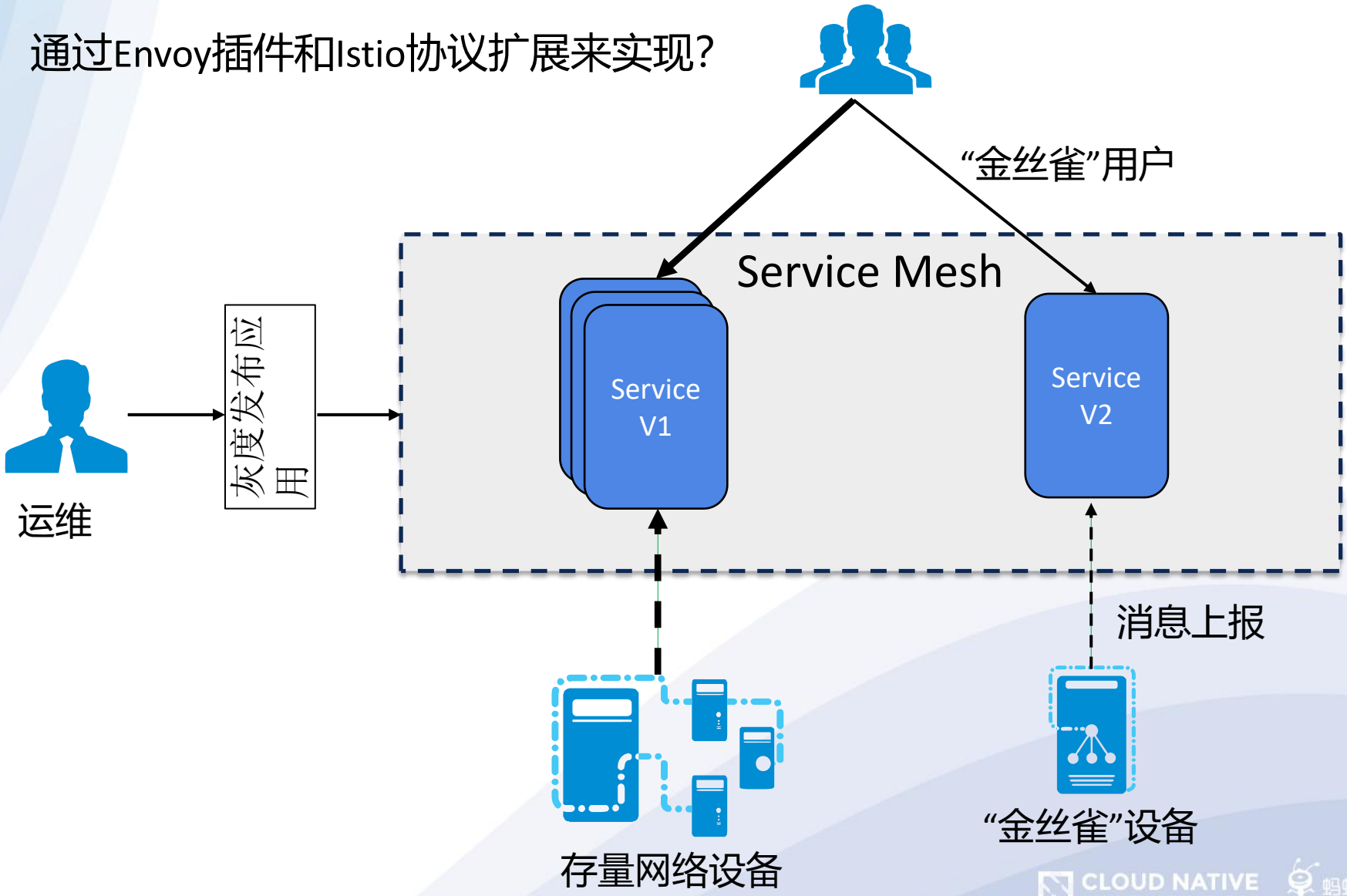
可在一个TCP请求没有对应的TCP服务，并且端口和HTTP服务没有冲突的情况下，TCP请求会被缺省发送到原始目的地。

遗留问题：
端口冲突的情况下，TCP请求将会被丢弃，导致客户端请求失败。

长期演进方案：通过自定义Envoy Filter扩展Service Mesh对应用层协议的处理能力

产品化增强：异步通信的流量管理

通过Envoy插件和Istio协议扩展来实现？



产品化增强-其他

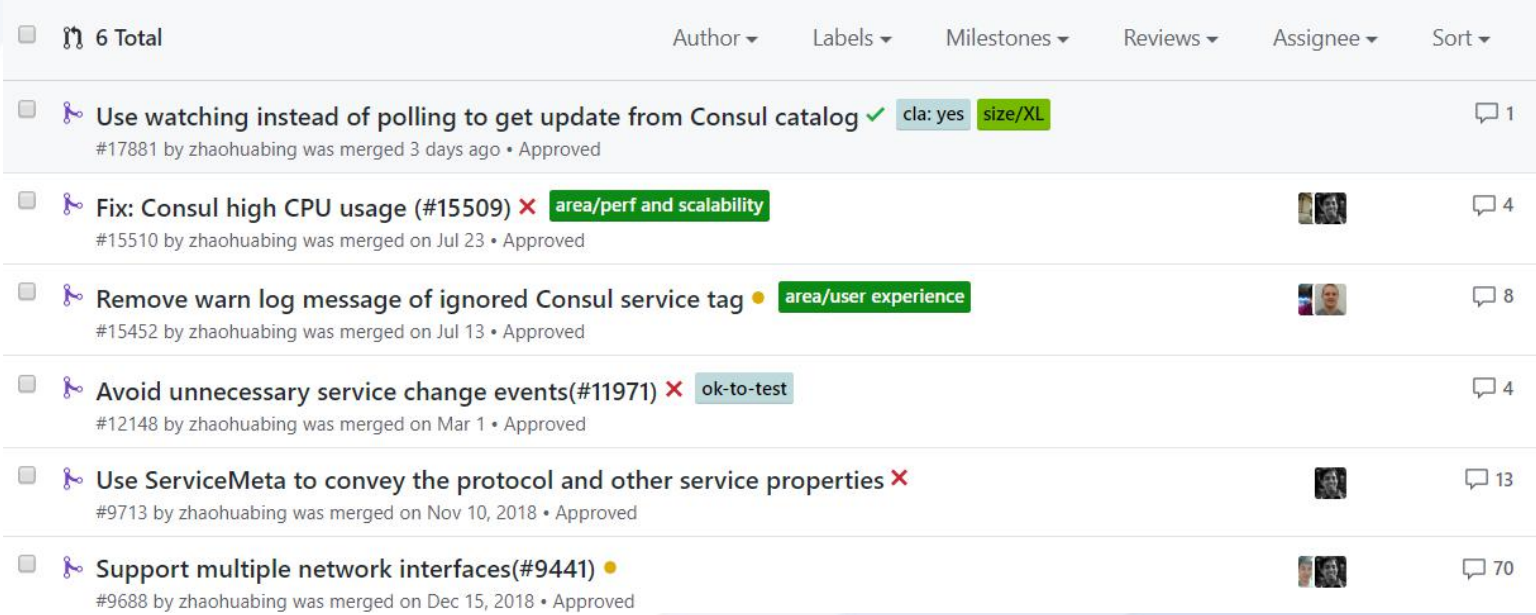
- APP：灰度发布、流量控制，更多的APP待业务场景触发
- IPV6支持增强
- 在Istio中集成方法级的调用跟踪
- 在Istio中集成Kafka调用跟踪

上游开源社区参与情况

所有通用的故障修复、性能优化和新特性都提交PR合入了上游社区。包括：

- Consul Registry性能和资源占用优化
- 多网络平面支持

在产品化过程中对Istio的改进会持续向社区进行贡献！



Service Mesh中文社区 https://www.servicemesher.com

个人博客 https://zhaohuabing.com

https://medium.com/@zhaohuabing

Github https://github.com/zhaohuabing

关注 ServiceMesher 公众号

扫码加入微信交流群
请备注姓名-公司信息

CLOUD NATIVE COMPUTING FOUNDATION 蚂蚁金服 ANT FINANCIAL ServiceMesher