Flomesh 软件负载白皮书

产品概述

Flomesh 软负载 (FLB) 是一款高效、稳定、易扩展的纯软件平台级应用负载均衡软件。它采用了“BGP+ECMP+eBPF+Pipy”的方案，提供了平台级的四层和七层负载均衡解决方案。

FLB 软负载可以在物理机、虚拟机、云主机、k8s 容器平台等多种环境中进行部署。从互联网边缘侧 (POP 点) 到互联网接入区 DMZ，再到容器集群入口 (Ingress & GatewayAPI)，FLB 都可以提供纯软件的四层和七层负载均衡能力。它不仅提供了媲美甚至超越硬件负载均衡的吞吐量、低延迟、高稳定性，还可以根据流量弹性的扩容和缩容，具有软件“起步小、易扩展”的特点，是一款“云原生、平台化”的软件负载均衡工具。

FLB 软负载可以快速部署在多种环境中，以拓展业务的接入能力，提升应用的交付能力。它是一款高性能、低资源消耗的软件负载均衡工具，无论是在容器环境中还是在低至 1C2G 的云主机中，FLB 均能很好地运行，并提供统一的低延迟、高吞吐、易管理的特性。

FLB 提供高度的定制化能力，用户可以通过定制管理控制接口 (Admin REST API) 来快速与 DevOps、云管等平台实现对接，提供多租户、自动化的负载均衡服务。此外，FLB 还提供了基于 PipyJS(PJS) 的二次开发扩展能力，用户可以使用 JS 语法快速定制四层和七层负载均衡的管理接口、路由策略、访问控制策略、负载均衡策略等。

第一章 软负载的演化和历史

在介绍 Flomesh 软负载之前，让我们简要回顾一下软负载的几个发展阶段、标志性技术架构、组件和特点。

第一代软负载：硬负载替代

软负载的发展始于千禧年后互联网的高速发展。当时，硬件负载均衡器成本高、难于扩展，成为互联网企业面临的问题。为解决这个问题，互联网工程师开始在开源软件中寻找可替代硬件负载均衡器的解决方案。经过快速发展，形成了“基于 LVS 的四层负载均衡+基于代理服务器的七层负载均衡”的标准方案。流行的七层代理服务器包括 Nginx、Haproxy 和 Varnish 等。我们将这种“LVS+Nginx”或“LVS+Haproxy”的方案称为第一代软负载。这个时代的另一个技术特征是，软负载的建设和使用方通常是互联网运维团队，他们自己开发配套的运维工具，围绕核心的“LVS+Nginx/Haproxy”形成了配套的工具链，包括自动化运维、监控和指标等。

第二代软负载：自服务化

第二代软负载是随着云计算的兴起而形成的。在云计算早期，云计算厂商进一步增强了第一代软负载方案，主要改进了多租户、自服务和弹性能力。其显著特征是，核心技术仍然采用“LVS+Nginx/Haproxy”的技术栈，而软负载的建设者为云平台产品团队，使用者则为云端租户。这种建设者和使用者分离的模式使得软负载的管理更加精细，同时实现了更高的自动化程度。

随着云计算厂商的发展，出现了面向私有云的商业化软件负载均衡软件，其中典型的例子包括 Nginx 公司开发的 Nginx Plus 和被 VMware 收购的 AVI Networks。自服务成为第二代软负载的最大特征之一，有时也被称为“自服务软负载”。

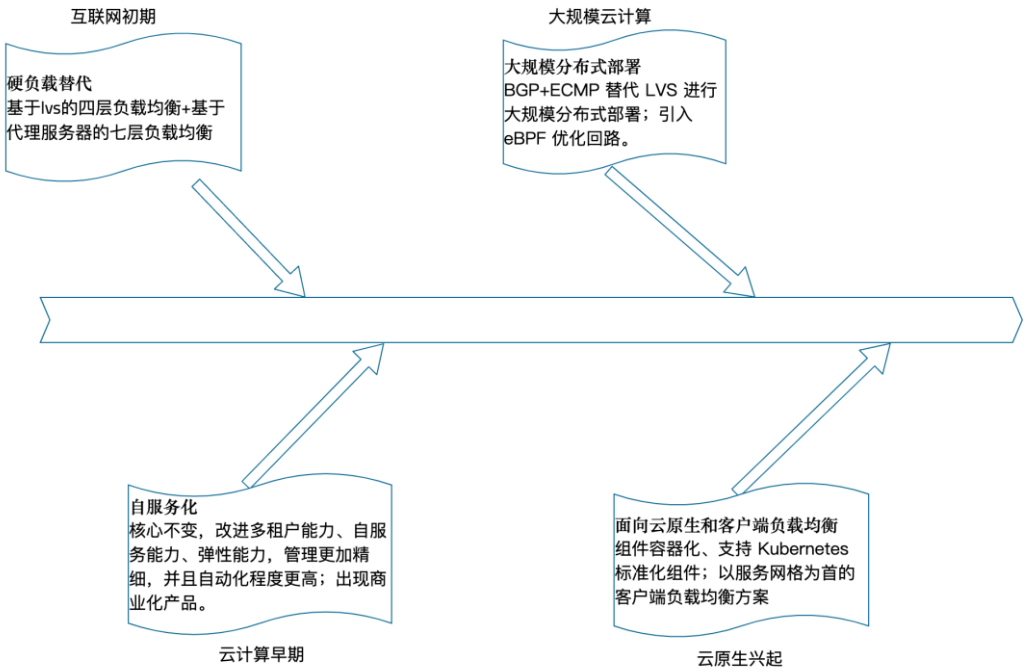
第三代软负载：大规模分布式部署

第三代软负载的出现是为了应对第二代软负载在面对大规模扩展时的限制。其典型的技术特征是在四层负载均衡中采用了 "BGP+ECMP" 的方案代替了 LVS，并引入了 eBPF 技术以优化回路等问题。典型的方案包括 Facebook 开源的 Katran 等。在七层

负载均衡中仍然采用了基于 Nginx/Haproxy 等代理的方案。由于引入了 BGP，第三代负载均衡可以在横向扩展上支持更大的规模，部署模式通常为大型分布式。因此，第三代负载均衡也被称为“分布式软负载”。

第四代软负载：云原生软负载

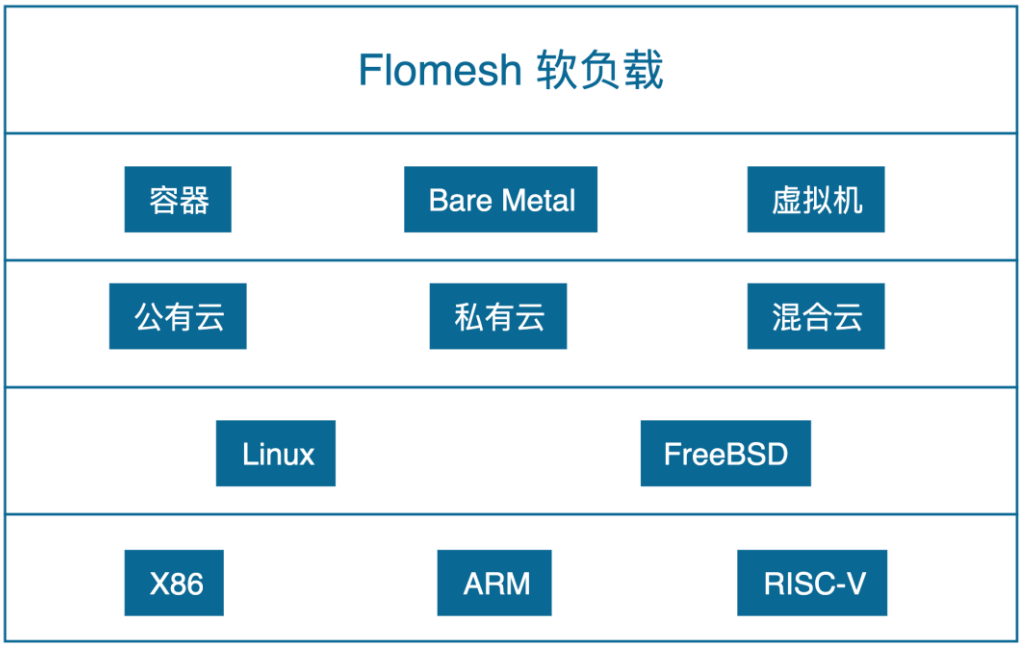
第四代软负载是随着容器平台广泛使用而兴起的。其典型的技术特征是容器化和客户端负载均衡的采用。容器化即包含了软负载各种组件本身的容器化，也包括对各种组件控制器广泛采用标准框架开发，如 kubernetes Ingress。客户端负载均衡的出现主要是为了应对微服务流行所带来的东西向流量的负载均衡。通过把负载均衡前置到服务调用方，形成客户端负载均衡的能力，这种能力更加灵活。这一代负载均衡的典型是服务网格中边车代理。这一代软负载也称为“云原生软负载”。



第二章 产品属性

2.1 产品定位

Flomesh 软负载是第四代软负载，旨在为云平台（包括公有云、私有云、混合云）提供 “负载均衡即服务” 。通过在公有云、私有云、混合云环境中使用 Flomesh 软负载，用户可以跨平台、跨集群、跨地域、跨供应商地获得负载均衡能力。这项能力不仅能够媲美甚至超越云平台自身的性能，同时也提供更加灵活的扩展性和统一的用户体验。



2.2 方案对比：私有云

相比于公有云或私有云运营团队自建的基于开源软件的负载均衡方案，Flomesh 具有更好的通用性和移植性，可以快速适配不同的云平台，以满足企业客户采用混合云和多云的发展需求。此外，Flomesh 软负载具有可扩展的特性，使得许多大型企业用户可以基于 Flomesh 软负载自建和定制专有的软负载平台。

2.3 方案对比：公有云

与特定云平台的负载均衡软件相比，如 AWS 的 NLB 和 ALB，Flomesh 软负载具有更好的性价比和更高的 ROI。在处理相同吞吐量的情况下，Flomesh 软负载的成本可以低至同等云服务成本的十分之一，甚至更低。同时，由于 Flomesh 软负载使用统一的技术栈适配了物理机、虚拟机、容器平台、微服务平台等多种场景，因此提供了更一致的管理体验和更有效的学习成本，实现了 “一次投入、到处使用” 的效果。

	AWS LB	Flomesh LB
计价模式	按请求量收费，超百万后，每百万额外收费	按照许可或者实例数收费
私有化部署	N	Y

私有化部署	Nginx	Flomesh LB
免费	Y	Y
开源	N	Y
二次开发	成本高	成本低

第三章 产品架构体系

3.1 控制台

Flomesh 软负载控制台基于 Strapi[官方网站] 开发，租户、配置、权限等统一存储于关系型数据库。Flomesh 控制台默认采用 PostgreSQL 或者 MySQL 作为存储。

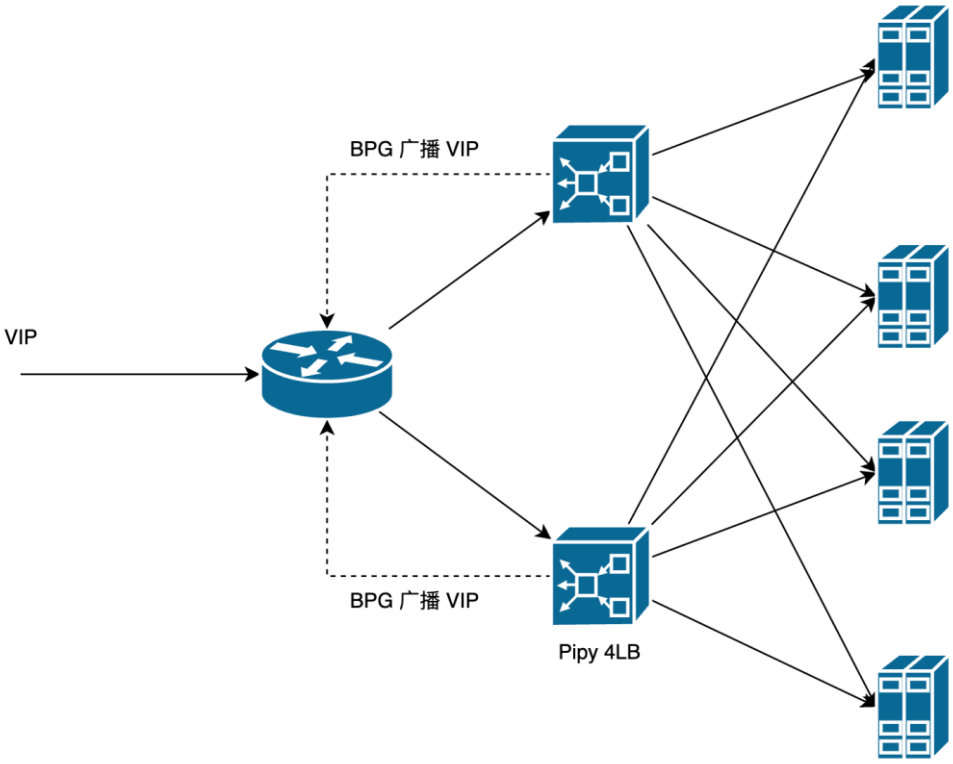
3.2 Pipy Repo（配置中心）

Flomesh 软负载采用 Pipy Proxy 作为七层代理，使用分布式的多个 Pipy 实例，并以 Pipy Repo 作为注册中心和配置中心。当新的 Pipy 实例启动时，它会注册到 Pipy

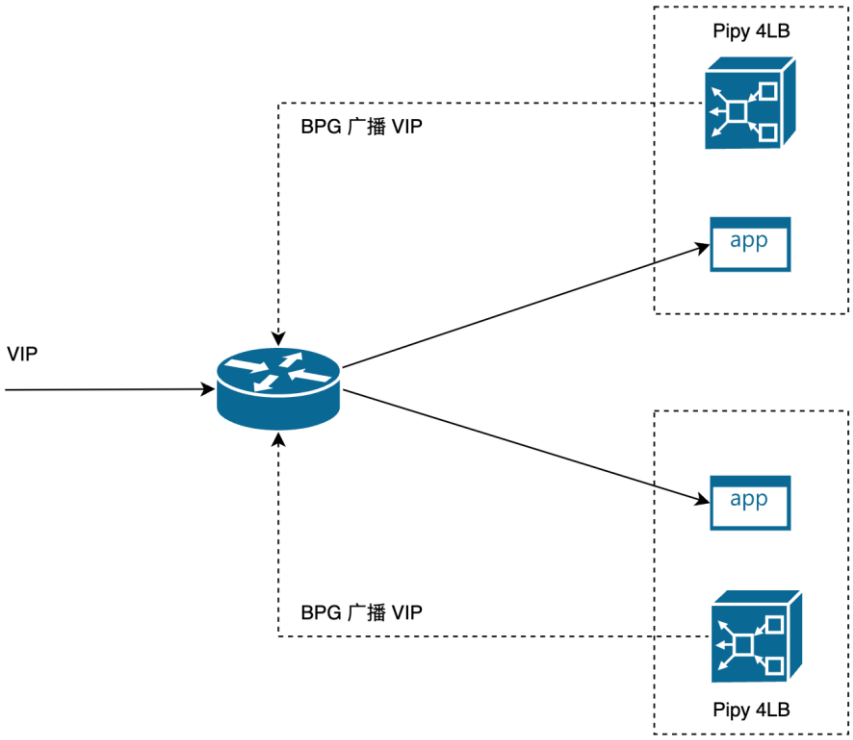
Repo 并获取属于自己实例的配置信息。因此，Pipy Repo 在 Flomesh 软负载中起到了注册中心和配置中心的作用。

3.3 Pipy 4LB

当 Pipy 4LB 采用集中部署模式时，一个 Pipy 4LB 集群会通过 BGP 协议向外宣告相同的 IP。路由器会基于这些宣告信息，使用 ECMP 策略将路由包转发到 Pipy 4LB 集群的不同节点。Pipy 会监听这些 IP 的指定端口，收到包后会根据负载均衡策略将其转发到上游服务器。

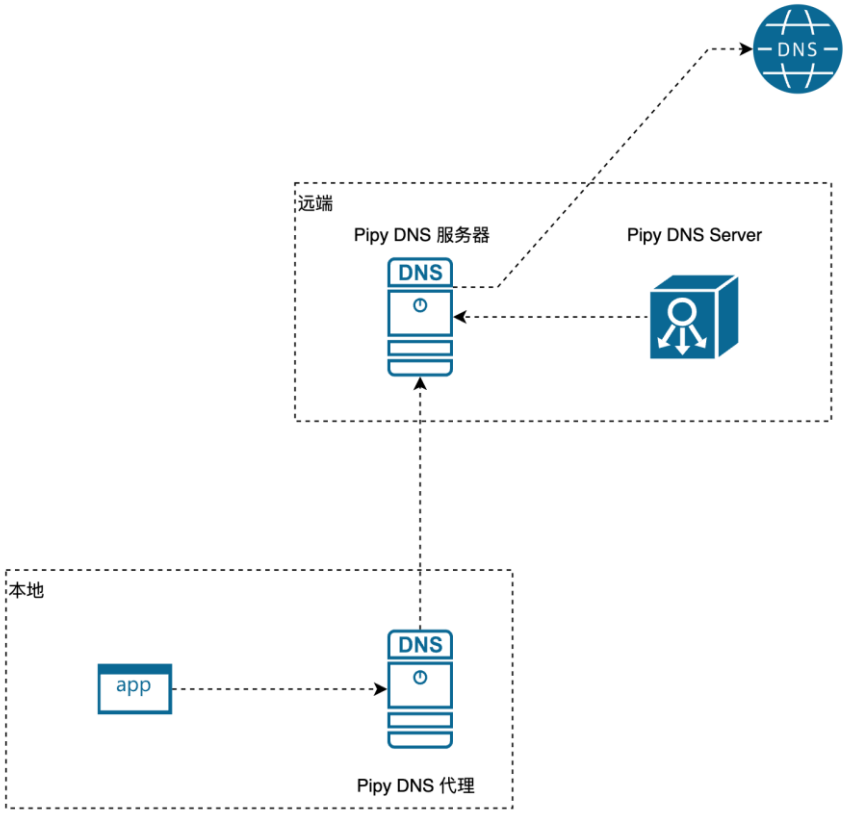


当 Pipy 4LB 采用边车模式部署的时候，Pipy 和业务进程会一同部署。其中，Pipy 负责向上级路由器宣告 IP，而业务进程则负责监听指定 IP 的端口。在这种模式下，流量不会经过 Pipy，而是直接到达业务进程。



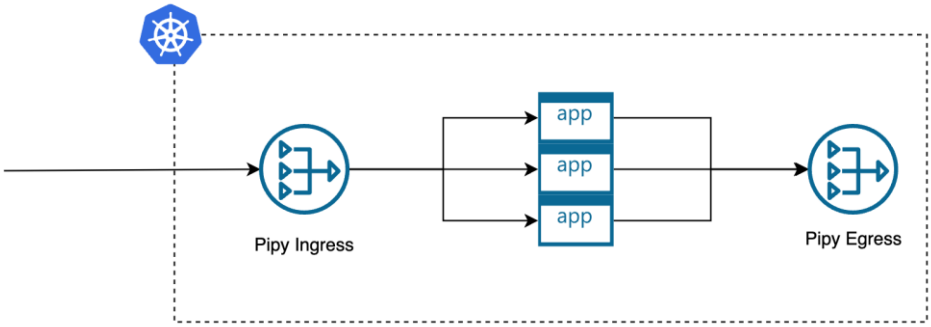
3.4 DNS 负载均衡（可选）

Pipy 可以作为 DNS 服务器或代理，同时工作在客户端和服务端。它可以响应客户端的 DNS 请求，并从控制器组件获取并维护应用的 IP 地址。Pipy 可以根据不同的策略为不同的客户端返回不同的解析记录，例如基于地理位置或健康状况进行负载均衡。



3.5 Ingress/Egress 控制器（可选）

当 Flomesh 软负载部署在 Kubernetes 容器平台上提供 Ingress 和 Egress 功能时，该控制器会被部署到 Kubernetes 指定命名空间。控制器负责通过 API Server 监听 Kubernetes 集群中各种资源的变化，并根据这些变化生成 Pipy 配置和动态脚本。这些配置和脚本会被推送到 Pipy Repo，并由 Pipy Ingress 实例获取、动态加载和自动执行。该控制器还支持 Gateway API。



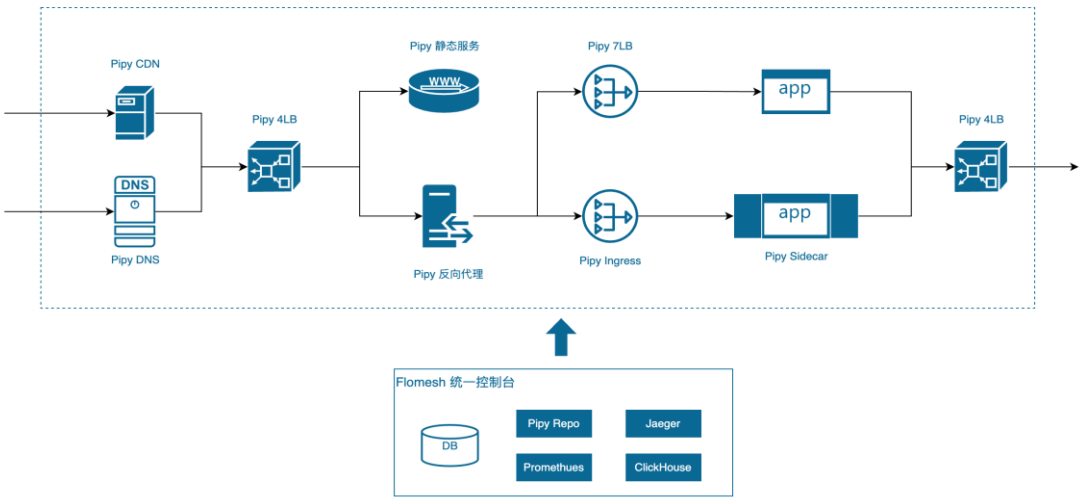
3.6 边车控制器（可选）

当 Flomesh 软负载提供客户端负载均衡（client loadbalance）时，Pipy 进程和业务进程采用边车方式部署。在应用程序启动时，Pipy 边车完成服务注册。当应用程序访问外部服务时，Flomesh 软负载会通过 iptables 或 eBPF 拦截请求并将其重定向到 Pipy 边车，Pipy 边车会完成服务发现并实现负载均衡。边车控制器还提供边车注入、边车启停、边车配置热加载等功能。

3.7 ELB 控制器（可选）

当 Flomesh 软负载部署在容器平台提供 ELB 服务时，ELB 控制器会监听 Kubernetes 中类型为 LoadBalancer 的服务，并据此向 Kubernetes 集群外的 BGP 路由器注册 ExternalIP。

第四章 组件拓扑



第五章 产品功能清单

Flomesh 软负载从功能角度分成几个大的方面：

1. 多租户管理
2. 技术组件管理
3. 四层负载均衡
4. 七层负载均衡
5. 客户端负载均衡
6. DNS 负载均衡
7. kubernetes Ingress 与 Egress
8. kubernetes ELB
9. API 负载均衡
10. 图形化用户控制台

5.1 多租户管理

组织管理。Flomesh 软负载是面向企业用户的多租户负载均衡即服务平台型产品。平

台上的用户以“组织”进行分组管理，任何一个用户都一定隶属于某一个“组织”。“组织”之间可以组成树形隶属关系。在该平台上，授权、资源访问控制等操作通常以“组织”为单位进行。例如“A 服务”被授权给“X 组织”时，那么所有隶属于“X 组织”的服务都可以访问“A 服务”。

- 项目管理。在 Flomesh 软负载管理平台上，所有的资源都按照“项目”进行分组的。这些资源包括注册中心、服务、API、负载均衡等，都隶属于不同的项目。在同一个项目内，资源之间的访问可以采用“宽松”访问模式，即同一个项目内的资源可以不受约束的互相访问；同一个项目内的资源之间的访问，也可以采用“严格”模式，即每个资源有自己独立的授权和访问控制规则。不同项目之间的资源，通常需要进行某种授权才能访问。
- 用户管理。Flomesh 软负载平台提供了常规管理系统的标准用户管理能力，包括用户注册、信息编辑等。在平台上，管理员可以对用户进行授权，授权后用户可以执行指定的操作。这些操作都会被记录下来，管理员可以进行审计。
- 角色管理。Flomesh 软负载平台采用基于角色的访问控制(RBAC，Role Based Access Control)模型，对所有资源的访问权限进行管理。所有资源的权限都是指定给特定的“角色”。管理员进一步将“角色”和用户绑定，使用户可以访问资源。
- 日志管理。在控制台上用户可以在自己权限范围内查看应用日志、应用全文日志（需要配置开启）、用户访问和操作记录。
- 事件管理。软负载平台自身可以产生各种事件，如告警、提示等。同时软负载平台也可以对接被管理的技术组件，接收事件，并在控制台进行集中展示和管理，如接收 k8s 的事件并进行处理。

5.2 技术组件管理

Flomesh 软负载平台由多个组件组成。这些组件，有些是 Flomesh 团队自研，有些是第三方组件用于满足特定的功能和管理需求。所有 Flomesh 团队自研的组件，均已开

源，在 https://github.com/flomesh-io 。这些组件有些是可以通过 Flomesh 软负载平台安装部署的；有些是可以通过配置参数集成使用的。根据组件的功能，组件可以分为如下几类。

- 4LB 集群。4LB 集群由多个 Pipy 组成，这些 Pipy 提供代理、路由、负载转发、BGP 宣告等核心功能。
- Tracing 类组件
  - Jaeger
- 指标类组件
  - Prometheus
- 数据库组件
  - MySQL
  - PostgreSQL
- NoSQL 组件
  - Clickhouse
  - ElasticSearch

5.3 四层负载均衡

Flomesh 负载均衡提供多租户的四层负载均衡功能，包括 IP 高可用、TCP/UDP 包转发、负载均衡、代理、隧道等核心功能。对于不同子网间流量，采用 BGP+ECMP 的 IP 高可用和负载均衡方案；对于相同子网内的流量，采用 eBFP 的 IP 高可用和负载均衡方案。当需要在负载均衡过程中施加高级管控策略，如访问控制、动态故障迁移等，采用 Pipy Proxy 进行包转发。

- IP 高可用。当原地址和目的地址处于不同子网或不同交换机 LEAF 时，Flomesh 负载均衡采用 BGP+ECMP 的方式提供 IP 高可用功能。在这种模式下，集中部署的 Pipy（或随目的地址进程部署的 Pipy sidecar）会向 BGP 路

由器进行 BGP 宣告，声明目标地址的进程拥有指定的 VIP。根据这些 BGP 声明和 BGP 路由策略，BGP 路由器将在多个目标节点之间进行负载均衡。同时，Pipy 也会监控目标进程和端口状态，当目标端口失效时（例如目标进程意外退出），Pipy 会进行 VIP 宣告并从 BGP 路由器中摘除这些失效的目的地址路由。当原地址和目的地址在同一子网时，请求不会经过 BGP 路由器。此时，Flomesh 软负载采用基于 eBPF 技术的方式，将请求的 VIP 地址基于负载均衡策略（例如轮询）转换为真实的目的地址，从而实现 IP 高可用。

- BGP 宣告。Flomesh 核心组件 Pipy 加入了 BGP 宣告模块，可以将 Pipy 部署在中央位置，或者随着目标地址进程一起做 sidecar 部署，实现 BGP 宣告。在中央模式下，Flomesh 软负载通常监听某个注册中心（例如 Eureka）。当服务的多个提供者注册其 IP 地址时，Flomesh 软负载会将这些 IP 地址宣告给 BGP 路由器作为同一个 VIP，这样经过 BGP 路由器的数据包会基于 ECMP 协议，在多个目标地址之间轮询建立连接并发送数据包。当以 Pipy sidecar 方式部署时，Pipy 的地址进程会与真实 IP 地址一起启动并获得真实 IP 地址。Pipy 会根据预先配置的 VIP 向 BGP 路由器宣告这个 VIP，BGP 路由器据此向目标地址轮询建立连接并发送数据包。当目标地址失效时，例如目标地址的进程退出，Pipy 会监测到这种变动，进行 BGP 摘除宣告，从 VIP 列表中移除该目标地址。
- 四层代理。在四层负载均衡过程中，当需要进行策略管理时，例如身份识别、访问控制、链路加密等，Flomesh 软负载会拦截流量并使其经过 Pipy Proxy，在 Pipy Proxy 中执行所需的策略。需要注意的是，如果不需要策略管理，则不会进行流量拦截，包也不会经过 Pipy proxy。在四层层面，Flomesh 软负载支持 TCP 代理和 socks 代理。其中，socks 代理支持 socks4 和 socks5 协议。
- 四层隧道。在某些情况下，原地址和目的地址可能处于不同的网络中，甚至跨地域、跨数据中心，甚至需要穿透防火墙。此时，Flomesh 软负载会在原地址

和目的地址之间建立隧道，在隧道中进行包的转发、路由和负载均衡。Flomesh 软负载支持基于 HTTP/1.1、HTTP/2 和 gRPC 协议的隧道技术。同时，Flomesh 软负载支持在一个隧道中的多路 TCP 连接，采用多路复用技术。因此，隧道技术不仅可用于穿透防火墙，还可降低底层 TCP 连接所占用的网元资源。该技术通常用于 IoT 长连接低带宽数据传输场景中，通过在隧道中多路复用 TCP 连接，可以减轻路由器等底层网络设备因大量 TCP 连接而带来的资源占用。

- 策略管理。在负载均衡的过程中，通常需要对流量进行某些策略管理，例如身份识别、访问控制、限制带宽、限制连接建立频率等。此时，Flomesh 软负载采用基于流量拦截技术，使流量经过 Pipy Proxy，在 Pipy Proxy 中执行具体的策略。当策略为全局策略时，例如建立连接的频率，Flomesh 软负载会采用中央化的统计和计数组件进行计数统计，Pipy Proxy 通过长连接和该计数组件实时通讯以执行规则检测。该中央组件也是一个独立的 Pipy。当多个策略需要作用于一个连接时，例如先进行 IP 访问控制再进行带宽限制，这些策略会以链式方式作用，这些策略被包装成 plugin，并按照链式顺序进行执行。

5.4 七层负载均衡

Flomesh 软负载提供多租户、多集群、多协议的七层负载均衡能力。这些能力基于 Pipy Proxy 构建，通过控制器和 Kubernetes 等云平台集成，呈现的形态包括 Ingress/Egress、GatewayAPI、API Gateway、Kubernetes ELB、Sidecar Proxy 等。

Flomesh 软负载七层负载均衡（7LB）支持 HTTP/1.x、HTTP/2、DNS、Redis、MQTT、Dubbo、gRPC、Thrift、TCP 短报文等协议。基于对这些协议的解析，提供了典型的负载均衡功能，包括路由（CBR）、转发、代理、报文改写、负载均衡、Failover 等。此外，通过扩展插件，还实现了内容过滤（WAF 过滤）、身份识别、访

问控制等安全扩展功能。同时，Flomesh 还支持服务发现、熔断降级、灰度发布等典型的服务治理功能。

通常，七层负载均衡和四层负载均衡会搭配使用。为了降低配置和管理的复杂度，Flomesh 软负载提供了在同一个 Pipy 中同时运行四层负载均衡和七层负载均衡的模式。

此外，七层负载均衡还支持以插件方式实现模块化策略执行。各种策略可以以插件方式配置或动态注入到 Pipy 中。通过 Pipy 热重载功能，可以在不重启 Pipy 的情况下动态生效，以提供更高的服务水平协议 (SLA) 和服务质量 (QoS)，避免因为重新加载配置导致的服务中断。

Flomesh 还使用了 PipyJS 技术，该技术支持使用 JavaScript 实现完全的插件功能。在这种情况下，可以在不重新部署或升级 Pipy 的情况下，部署新的插件。这进一步提高了服务的 QoS 和 SLA，同时也降低了升级和管理 Pipy 可执行文件的复杂度。

5.5 客户端负载均衡

客户端负载均衡是一种常用的微服务技术手段。服务调用方可以通过 SDK 或 sidecar 进程对多个服务提供者地址按照负载均衡策略进行访问。一个典型的用例是 HashiCorp 的 Consul 微服务框架中的 Consul Agent，这是一种标准的 sidecar 模式。Flomesh 软负载提供了类似 Consul Agent 原理的客户端负载均衡能力。无论微服务调用方进程运行在物理机、虚拟机还是容器中，Flomesh 软负载都可以手动或自动地注入 Pipy sidecar。当服务调用方发起请求时，请求会被 Flomesh 软负载拦截后进入 Pipy sidecar proxy 完成负载均衡。除了负载均衡功能外，Pipy sidecar 还可以通过模块方式提供多种非负载均衡的技术能力，如拨测、强制策略检查（类 OPA）、服务注册等功能。这些非负载均衡模块随 Flomesh 软负载产品提供，但默认不能通过 Flomesh 软负载控制台进行操作，需要通过专业服务进行配置和管理。

客户端负载均衡除了提供负载均衡功能外，还有很多其他优点。例如在 failover 高可

用场景和多集群服务发现场景中也经常使用。Flomesh 软负载基于客户端负载均衡技术，提供了进程级的 failover 能力。

对于多集群服务发现场景，Flomesh 软负载提供了两种方案：一种是基于 iptables 和 Pipy sidecar proxy 的方案，另一种是基于 eBPF 和 Pipy Proxy 的方案。这两种方案各有优劣，需要根据使用环境和场景进行选择。如果您需要详细的决策方法，请联系 Flomesh 的专家技术团队。在采购 Flomesh 软负载产品的情况下，该类远程专家服务是包含在产品服务范围内的，无需额外付费。

5.6 DNS 负载均衡

DNS 负载均衡是一种常用的负载均衡方案，其基本原理是一个域名可以对应多个 IP 地址，DNS 服务器对每次域名查询返回不同的 IP 解析结果，从而将客户端请求引导到不同的 IP 地址上去。Flomesh 软负载的核心组件 Pipy 中包含了 DNS 模块，通过 PipyJS 可以快速开发灵活的 DNS 解析方案。在 Kubernetes 环境中，默认提供基础的 DNS 服务发现能力，该能力通过 Pipy sidecar 或 Pipy Proxy 方式提供。在工作原理上，Flomesh 软负载会采用 iptables 或 eBPF 技术对 DNS 查询进行拦截，并将其定向到 Pipy DNS 模块（通过 53 端口暴露）。当业务进程进行 DNS 查询时，会首先被 Pipy 处理，处理逻辑根据具体的场景决定，或者是根据用户特定需求定制 PipyJS 脚本。对于 Pipy 处理的 DNS 请求，Pipy 会将其发往上游 DNS 服务器，在典型的 Kubernetes 环境中是 CoreDNS 服务。这里的 DNS 查询逻辑最简单的情况是将域名解析成一个服务的多个服务提供者 IP 地址，其作用和效果类似于 Kubernetes 中 Service 的 ClusterIP。但是作为 sidecar 的 Pipy DNS 由于和业务进程共享网络命名空间，因此其查询和解析速度要优于基于 iptables 或者 ipvs 的 ClusterIP 方案。同时，由于少了经过 ClusterIP 到真实 IP 的 NAT 或者代理过程，也会使速度更快。

除了用于 Kubernetes 的容器环境，Flomesh 软负载的 DNS 负载均衡也被用在非容器环境中。这些环境包括 Kubernetes 环境，其典型的应用场景包括：

1. 面向互联网的 GTM（Global Traffic Management）DNS 服务。该服务针对特定的域名，提供动态和智能解析。例如，针对不同来源 IP 地址的 DNS 查询，返回地理位置最近的入口 IP；或者针对预先设定的 failover 策略，返回针对特定域名的备用 IP 地址。
2. 面向局域网的 LTM（Local Traffic Management）DNS 服务。随着云计算的高速发展，企业用户越来越多地开始规划自己的网络，同时规划自己的 DNS 内网解析。针对不同的数据中心、不同的网络分区、不同业务分组都有一套管理严格的 DNS 分配和解析策略。这些解析策略一方面为没有实例的物理机、虚拟机、容器分配了对应的域名，同时也为不同的子网和业务线分配了不同的域。每个域都有自己专属的域名解析能力和流量出入口。这种情况随着微服务的发展变得流行，也变得更加依赖动态 DNS 解析。Pipy DNS 模块和基于 PipyJS 的脚本能力为动态 DNS 解析提供了基础。由于这种功能具有很强的客户定制化特征，目前该类功能是通过 Flomesh 软负载的专家服务提供。
3. 面向容器环境的 DNS 服务发现。该能力如本节第二段所述，主要用于 Kubernetes 环境中的 DNS 服务发现场景。Flomesh 软负载通过拦截业务进程的 DNS 查询请求到 PipyNS，实现了动态域名解析和服务发现。

5.7 Kubernetes Ingress 和 Egress

在容器平台环境中，负载均衡的主要形态包括 Ingress 和 Egress。这些形态的接口规范包括 Ingress、GatewayAPI 和 Egress/EgressGateway CRD。

Flomesh 软负载提供了针对 Kubernetes 环境中负载均衡的各种控制器。这些控制器主要采用 Go 语言开发，其相关实现都在 Flomesh 的开源仓库 https://github.com/flomesh-io/ErieCanal 中。

这些封装好的控制器可以通过 Kubernetes 工具链进行快速的部署和使用。其用户体验遵循了 Kubernetes 社区的用户习惯。

在 Flomesh 软负载控制台上，这些功能在 Ingress 菜单中。这里提供了图形化的配置

和管理能力。

5.8 Kubernetes ELB

在 Kubernetes 的标准实现中，定义了类型为 LB 的服务（Service），但是默认并没有实现为其分配 ExternalIP。各种公有云平台都有自己的实现。开源社区也有典型的实现，如 MetalLB。Flomesh 软负载也提供了自己的 ELB 实现。该实现分为两种模式：

一种模式是在扁平网络中，Flomesh 软负载通过 BGP 宣告的方式，直接对 BGP 路由器宣告某个特定的 ExternalIP。

另一种模式是在容器平台之外有 Flomesh 四层负载均衡集群。此时 Flomesh ELB 会自动在集群外的 Flomesh 四层负载均衡集群上注册 ExternalIP 的转发规则，形成“集群外四层负载均衡 + Ingress 七层负载均衡”的网络解决方案。

针对传统的非 BGP 网络，Flomesh 软负载提供定制化的基于 ARP 的 ELB 服务。目前，该服务不在 Flomesh 软负载标准产品中，是一种可选的 Flomesh 专家定制服务。

5.9 针对应用接口（API）的负载均衡

针对 REST API 的用户，Flomesh 软负载提供了完整的 API 级负载均衡能力。用户可以定义一个 API，例如某个 “域名 + 路径 + HTTP 方法” 的组合，并且可以使用这个定义来对该 API 的流量进行负载均衡和策略管理。在这种模式下，Flomesh 软负载实现了标准 API Gateway 的能力。

包括 API 负载均衡，Flomesh 软负载支持了多种层级的负载均衡：

- 面向互联网入口的负载均衡，包括 DNS 负载均衡、4 层负载均衡、7 层反向代理和负载均衡
- 面向子网入口的负载均衡，用于构建 4 层和 7 层的网络边界
- 面向主机和实例的负载均衡，类似传统负载均衡硬件的使用方式和功能
- 面向容器平台的负载均衡，覆盖包括 Ingress、Egress、多集群服务管理（MCS）
- 面向微服务的负载均衡，包括 DNS 服务发现、客户端负载均衡

- 面向应用接口（API）的负载均衡

5.10 图形化用户控制台

Flomesh 软负载提供了图形控制台，并已经在 GitHub 上开源了其实现：https://github.com/flomesh-io/traffic-guru。

大多数的 Flomesh 软负载功能都可以通过图形控制台来进行可视化操作，但是，也有少量功能需要根据客户需求进行定制化部署和后台管理。

除了提供多租户的软负载管理能力，Flomesh 软负载图形控制台还提供了可以灵活定制的各种 Dashboard。用户可以根据需要定制自己的 Dashboard。如果默认的 Dashboard 无法满足需求，用户可以联系 Flomesh 软负载专家支持团队进行定制。

Flomesh 软负载控制台默认运行在容器环境中，并且可以通过 Helm chart 进行部署。控制台还提供了基本的 Kubernetes 管理能力，例如查看各种 Kubernetes 资源的状态和配置等。此外，控制台还包括一个基本的可以执行 kubectl 的 web terminal，方便管理员在浏览器中操作和管理 Flomesh 软负载及其相关组件。