服务网格不是一个新概念，**在云原生时代，服务网格成为了将运行在Kubernetes之上的微服务连接成为容器化平台的一种实现方式。**如果没有服务网格，则每个微服务都需要配置接收（或发送）来自其他微服务的流量。服务网格完全改变了这一点。

**有了服务网格，微服务就能够以可靠，安全和可控制的方式相互通信，而不必进行手动配置，也不必花费大量时间和精力来维护微服务之间的连接。**

Kubernetes和服务网格是相互协作的，同时使用服务网格可以很轻松地实现更复杂的容器化架构。

有很多方法可以在Kubernetes之上搭建服务网格，在本文中，我们将比较一些常见的可用于建立服务网格的工具，以查看哪种工具更适合自己的业务场景。

[**AWS App Mesh**](https://aws.amazon.com/app-mesh/)

**现在许多基于Kubernetes的应用程序和微服务都在Amazon Web Services环境中运行，因此很难不谈AWS App Mesh。顾名思义，AWS App Mesh是Amazon自研的服务网格，旨在为Amazon服务创建服务网格层。**

作为Amazon产品，AWS App Mesh选择与Envoy结合来作为其服务代理。AWS App Mesh通过创建虚拟服务，可以实现在同一名称空间( namespace )内服务间的连接通信。这是因为，AWS环境中的每个微服务都可以找到该虚拟服务，利用其可将通信引导至其他微服务。

**AWS App Mesh可以与其他服务（如EKS，Fargate和EC2）的无缝集成是其最大的优势，但是在使用App Mesh方面也存在一些限制，例如，你不能迁移到App Mesh的外部环境或在多云环境中使用此服务。**

App Mesh还借助CloudWatch和AWS X-Ray来管理服务网格，这就意味着你可以在AWS 主仪表板上控制管理服务网格。尽管App Mesh不支持授权规则，但也支持诸如mTLS支持和负载均衡之类的安全功能。

[**Istio**](https://istio.io/)

Istio是Kubernetes中最受欢迎的服务网格工具。它最初是为Lyft开发的，但后来成为Lyft，Google和IBM的联合开发项目。考虑到Google是Kubernetes背后的公司，Istio能够支持Kubernetes中的多种部署类型，也并不奇怪。

与App Mesh相似，Istio也可以使用Envoy作为其服务代理，但并不仅限于Envoy作为唯一的入口控制器( ingress controller )。Istio的独特之处在于它提供了巨大的灵活性，而没有通常的复杂性。也可以将Istio用于其他容器化平台，但是它与Kubernetes的无缝集成使其成为有用的工具。

**例如，Istio支持网格扩展和多集群网格，这两个功能都是App Mesh和许多其他服务网格工具所没有的。Istio也可以控制处理流量访问和负载均衡，甚至还支持故障注入和延迟注入。**

使用Istio的唯一缺点是你可能会对它提供的功能感到不知所措。如果你有足够的资源使用Istio处理服务网格层，那么Istio可以利用其功能帮助简化组织最复杂的微服务体系结构。

[**Linkerd**](https://linkerd.io/2/overview/)

在Linkerd v2.x版本发布之前，Linkerd已是一种非常流行的服务网格工具。Kubernetes社区已经很好地集成了Linkerd新版本。并且在2020年4月中旬，Linkerd稳定的2.7.1版本已经发布。

**Linkerd完全是作为独立的服务网格工具构建的，因此它不依赖Envoy等第三方工具进行管理，内部使用linkerd-proxy作为服务代理。**

Linkerd最近的升级，还包括仪表板改进和针对金丝雀部署的流量拆分功能的可视化。这使其成为实时监视和编排金丝雀或蓝/绿部署的绝佳工具。

Linkerd在保持独立性的同时，还与入口控制器( ingress controller )保持高度兼容性。实际上，Linkerd能够与你使用的任何入口控制器一起使用，从而使其在这方面最灵活。要使服务网格与你的应用程序集成在一起，只需要一个简单的 linkerd inject命令即可。

Linkerd2也经过高度优化，安装仅需60秒。如果你正在寻找一种具有最佳性能的服务网格工具，那么可以尝试一下Linkerd2。作为非侵入性服务网格工具，Linkerd部署后不需要太多优化。开箱即用的配置足以支持复杂的微服务架构，并且能够防止重大攻击。Linkerd通过mTLS加密来增强应用程序安全性

**Linkerd的不足是，它目前可能不支持在多云环境和多集群网格中创建。**

但是，Linkerd也是专门为Kubernetes开发的工具，当用作Kubernetes实例的服务网格层时，它的功能不会因此降低。此外，它还可以与[OpenCensus](https://opencensus.io/" \t "_blank)配合使用，使跟踪和管理变得非常容易。

[**Kuma**](https://kuma.io/)

**Kuma使用Envoy作为服务代理，并支持任何入口控制器。Kuma与Consul Connect非常相似（我们将在下文介绍），但具有一些令人耳目一新的功能。**

Kuma不仅可以投入生产，而且还具有功能强大的服务网格工具所期望的功能。它支持与OpenTracing兼容的所有后端，并允许你使用外部CA证书。

不幸的是，此工具仍缺少某些功能。

**目前，在Kuma中无法进行基于路径( path-based )或基于请求头( header-based )的流量拆分。也不支持流量访问控制和流量指标等功能。这些功能可能会在以后的更新中引入，但是就目前而言，你必须手动进行代理模板处理才能解决这些工具的不足。**

尽管如此，Kuma看起来很有希望成为服务网格工具。它尚未达到其1.0.0版本（当前为0.4.0），但是该工具背后的开发人员听取了社区的意见，并乐于促使其成为功能强大的服务网格工具。

[**Consul Connect**](https://www.consul.io/docs/connect/index.html)

HashiCorp的Consul Connect也是一个服务网格工具。Consul Connect可以与Envoy等服务代理一起使用。它还可以与任何入口控制器一起使用，使其成为最容易集成到现有Kubernetes集群中的一种服务网格工具。

**Consul Connect可在任何Consul环境中无缝运行。不幸的是，它只能在 Consul 环境中工作。**

Consul Connect服务网格工具提供了许多方便的功能，还可以与HashiCorp其他产品一起使用。

Consul Connect支持从TCP到gRPC的所有内容，可与Kubernetes，VM和Nomads一起使用。Consul Connect完全支持网格扩展，因此你可以在一个跨多个云服务和集群的环境，使用它作为服务网格层来支持组织内的微服务。

Consul Connect需要改进的一方面是监视。但是，你可以集成Prometheus和Grafana之类的工具来监视数据。你只需要从服务代理中提取数据即可，而不是直接从Consul Connect中提取数据。

[**Envoy Proxy**](https://www.envoyproxy.io/)

以上这些服务网格工具都可以采用Envoy作为其服务代理。与其他代理工具相比，Envoy确实具有一些优势，负载均衡是所有这些工具中最突出的优势。

**Envoy自动重试，本地负载均衡和请求屏蔽，使你可以配置流量负载均衡以实现最佳性能。**

另一方面，可观察性使Envoy成为能够支持功能强大的网络的理想解决方案。

**总结**

以上这些服务网格工具的主要目标是：创建一种云体系结构，在该体系结构中，微服务能够以可靠，安全的方式相互通信。好消息是，无论使用哪种工具，你都将能够实现这一目标。