46Kubernetes 系列（四十三）Kubernetes 平台 Ingress 流程管理

Kubernetes 系列（四十三）Kubernetes 平台 Ingress 流程管理

在Kubernetes平台上，流量可能来自内部源(在集群内的工作负载之间)，也可能来自外部源(从外部客户机到集群中的工作负载)。后者通常被称为南北交通，是安全风险的主要来源。管理入口流量至关重要。

为了完成这项任务，一代又一代的工具出现了。在这里，我将讨论带来这些工具的问题、可用选项和它们的局限性。需要注意的是，本文只关注入口流量，并假设到达集群边缘的流量已经通过了基本的安全检查，如Web应用程序防火墙(WAF)和网络防火墙。另外，在我们关于流量管理的上下文中，术语入口(以及入口控制器和入口规则)不应该与[NetworkPolicy]上下文中(https://kubernetes.io/docs/concepts/services-networking/network-policies/#networkpolicy-resource)的"作为策略类型的 入口" 相混淆。

Kubernetes生态系统中目前有四种工具家族，用于管理进入集群的流量:

- Kubernetes Service (LoadBalancer类型)
- Kubernetes Ingress
- Ingress CRD
- Gateway API

我按时间顺序列出了它们。最早和最简单的是Kubernetes Service对象。它只允许外部流量跨界进入集群，而不需要真正管理流量。

Kubernetes Service

Kubernetes 服务资源有四个 ServiceTypes (ClusterIP、NodePort、LoadBalancer和ExternalName)。只有LoadBalancer类型是在集群外部公开工作负载的实现方法。相比之下，ClusterIP和ExternalName类型在集群中起作用。NodePort通过在每个节点上打开一个入口点来暴露工作负载，这是沙盒环境之外的 不推荐 。

具有 LoadBalancer 类型的 Service 资源将流量从外部客户端引导到内部Pods。要实现这一点，需要在底层进行一些活动。首先， 云控制器管理器 从云提供者提供一个负载均衡器设备。在前端，这个负载均衡器保护一个从客户端可路由的IP地址。后端以每个Kubernetes节点上的节点端口为目标。然后，在节点上，kube-proxy监听节点端口，并从那里将传入的流量转发到节点本身上运行的目标Pod。 业务控制器 自动同步负载均衡器和节点端口之间的配置。这种类型的服务资源在传输层操作，与应用程序层消息无关。它在集群边界上打开一个"门"，但不检查哪些内容(无论是无害的还是恶意的)被带入了集群。在生产中，我们很少单独使用这个选项。

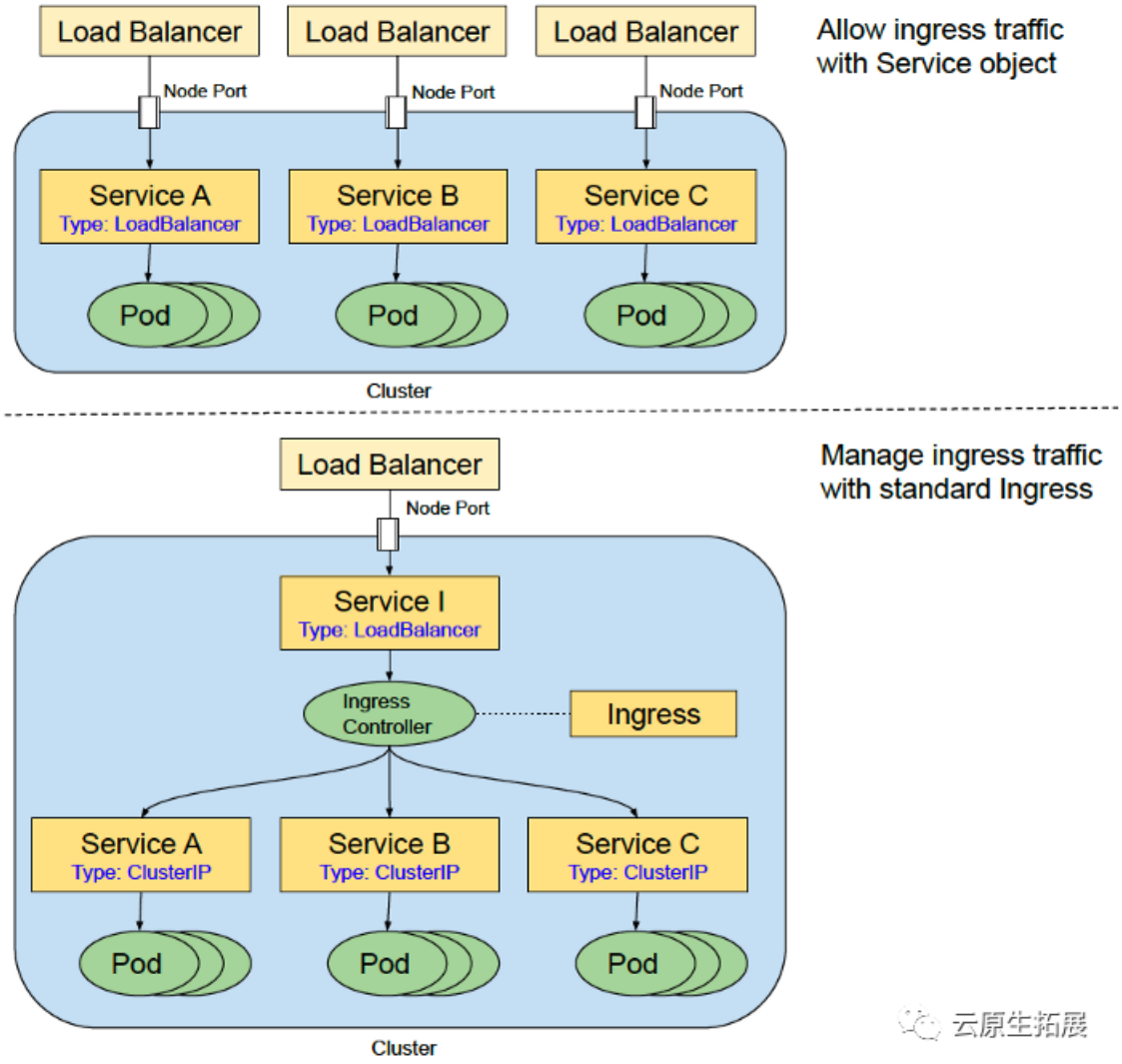
为了安全，我们需要的不仅仅是永远敞开的大门。我们需要检查和控制穿过大门进入集群的不可信的流量。例如，我们经常想要终止TLS来检查web请求试图去哪里。在集群中，我们可能还希望只将请求路由到其允许的目的地。在某些情况下，我们可能需要委托第三方在登机口授权web请求。所有这些应用层的繁重工作都需要它们专用的计算资源——入口控制器。

标准 Kubernetes Ingress

标准Kubernetes Ingress 是一种更好的公开HTTP工作负载的方法，因为它具有控制流量的能力。在Kubernetes中， Ingress 资源是定义流量控制规则的抽象资源类型(即接口)。一个 Ingress 控制器是满足规则的实现。是Ingress控制器像勤奋的看门人一样完成给定的任务。这些任务可能包括监视传入的流量、监视规则更新以及针对流量执行规则。

通过Ingress资源承担流量控制，我们可以使用 关注点分离 修改 Ingress路径 。我们现在只保留一个LoadBalancer Service对象，它具有更大的作用。它充当所有入站流量的通用网关，目的是到达集群内部的不同工作负载。这项任务是关键的和涉及的。幸运的是，我们可以将Service对象的管理委托给入口控制器。负载均衡器设备允许入口控制器在进入集群的过程中仔细检查和路由传入请求。Ingress控制器的供应商支持基于诸如 URL路径 和 SNI 等属性的广泛路由功能。另一方面，租户工作负载现在只需要在集群中公开它们自己，期望Ingress控制器将请求路由到它们的路径。他们的生活变得更容易。ClusterIP类型的服务对象可以满足需求。

Service Object VS Ingress



通过Ingress资源，我们将多个打开的门合并为一个共享的门。因此，我们将所有进入的流量都通过这个网关，并集中监视和控制通过的流量。Load balancer Service 对象为入口铺平了路径。Ingress控制器在该通道中充当安全警卫。Ingress资源类似于守卫的"工作指令"。这种关注点分离还有助于定义团队之间的职责边界。在多租户模型中，开发应用程序(例如计费服务)是租户的责任(例如计费功能的开发人员)。他们不设计如何对外公开他们的服务。Ingress、它的实现和路由配置都被认为是所有租户的基础设施服务。它们属于平台团队的责任范围。

在创建集群时，平台团队需要安装所需的Ingress控制器。在某些情况下，租户可能需要Ingress的多个实例，以及Ingress控制器的多个实例。同样，安装所有Ingress控制器并从每个Ingress资源对象正确引用它们是平台团队的责任。

平台团队还承担为其租户选择Ingress控制器实现的角色。Kubernetes只在其本地Ingress资源中定义标准接口，将控制器实现留给每个技术开发人员。无论哪个开发人员，他们的设计通常都遵循控制器模式。作为一个例子，查看这个文档(How NGINX Ingress Controller Works | NGINX Ingress Controller)，了解NGINX如何实现它的Ingress控制器。Ingress控制器的作用类似于web服务器上的反向代理。Kubernetes编译了一个实现提供程序列表。从列表中，我们可以找到如下的控制器:

- 应用程序交付控制器(ADC)解决方案的传统参与者: F5， HA Proxy 和 Citrix；
- 由公共云提供商:Azure的[AGIC] (https://docs.microsoft.com/en-us/azure/application-gateway/tutorial-ingress-controller-add-on-existing?toc=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Faks%2Ftoc.json&bc=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fbread%2Ftoc.json), [GCP 的 GKEIngress)(https://cloud.google.com/kubernetes-engine/docs/concepts/ingress)和(AWS 上的 ALB Ingress 控制器) (https://kubernetes-sigs.github.io/aws-load-balancer-controller/v1.1/)
- 基于NGINX: NGINX控制器和Kong Ingress控制器
- 建立在Envoy(一个现代开源代理项目)之上: Contour, Ambassador, EnRoute, Gloo

除了第三方选项，Kubernetes社区自己也在NGINX上维护了一个不错的Ingress控制器实现。人们不应该将这个社区版本的

Ingress控制器(称为Ingress - NGINX)与F5团队命名的另一个类似的NGINX版本的实现混淆。关于Ingress控制器选项的更多信息，LearnK8s网站维护了一个综合表，作为社区的最大努力。所以对这些信息持保留态度。

标准的Ingress API花了五年的时间才完成。随着它的发展，许多人发现了它的局限性。首先，标准的Ingress资源根本不处理非http服务。为了公开任意TCP端口，Ingress文档建议退回到具有LoadBalancer类型的Service对象，将我们送回原点，因为缺乏控制能力。第二，即使使用HTTP工作负载，标准Ingress资源可以表达的路由规则仍然很简单。超出基本框架的特性需要只有提供者才能理解的注释块。此外，Ingress资源集中了所有承租者的路由配置，而在实际操作中，单个承租者通常希望从各自的名称空间管理自己的路由。

Ingress CRD

原生的 Kubernetes Ingress用作标准的API模式。这样做是以可扩展性为代价的:开发人员必须牺牲灵活性来在他们自己的接口中公开高级功能。使用专有的 注解 的解决方法很麻烦。伴随着注释扩展的是可维护性的恶化。自从标准Ingress开始，许多Ingress控制器开发人员(如 Gloo ， Contour ， Traefik 和 HAProxy)在大量注释的痛点上意见一致，并共享切换到 crd (自定义资源定义)的动机。

一些公司的Ingress CRD产品向前迈出了一步。因此，它们的用户可以使用开发人员定义的属性将其Ingress描述为自定义资源(CR)。以 Traefik Labs 为例，他们有命名为 IngressRouteTCP 和 IngressRoute 的crd。前者为任何TCP流量提供路由，这是标准Ingress中所缺少的功能。后者为web流量构建了许多超越基线路由功能的特性，例如将路由配置委托给租户，以及引用来自不同名称空间的秘密。这些CRs与其他Traefik CRs 一起工作，例如 Middleware 和 TLSOption ，在他们的产品中交付一套完整的特性。

在Ingress CRD家族中，我们可以基于任何想要的流量管理特性进行选择，只要有支持这些特性的提供商。有许多特征需要寻找。通常需要考虑的是:

• 其他协议支持:HTTP/2, gRPC, TCP, WebSocket

• 动态更新配置，不中断流量

• 高可用性、吞吐量、负载均衡算法、健康检查、会话保持

• 流量控制:限速、重试、断路、请求重写、分流、镜像、金丝雀和蓝绿色释放

• 与服务网格的兼容性

在货比三家之前，平台团队应该调查他们的涉众，并列出租户需要的详细功能。在供应商方面，Ingress控制器业务的许多竞争者也扩大了他们在CRD市场的足迹。以下是一些开发人员及其路由crd名称的列表:

• Traefik Labs: IngressRoute、IngressRouteTCP

• Kong: KongIngress

• F5 Nginx: VirtualServer and VirtualServerRoute

• Countour: HTTPProxy

• Gloo Edge: Gateway ， Virtual Service

• Ambassador: Host 和 Mapping

• Istio: Gateway ，VirtualService和DestinationRule

为了阐明与服务网格的兼容性，Istio的Gateway是一个很好的例子。Istio的网关是作为Istio服务网格的一部分提供的Ingress CRD。网关CRD与其他Istio CRD结合以启用服务网格特性，例如可跟踪性。如果您的集群运行Istio作为服务网格是一个硬性要求，那么它的Gateway CRD自然是您选择的Ingress。即使其他入口控制器或CRD可能工作，Istio Ingress Gateway在Istio服务网格上提供统一的配置体验。也就是说，另一个服务网格提供商Linkerd持有相反的设计哲学:他们的开发人员故意选择不实现他们自己的Ingress crd。相反，他们的目标是使他们的产品与一系列常用的输入控制器或crd兼容。

这些crd非常灵活，它们的用例远远超出了为web客户端服务的范围。例如，许多人将它们配置为API网关，这是微服务体系结构中API网关模式的核心组件。入口CRD作为API网关的使用是如此广泛，以至于Gateway这个术语比ingress更容易被理解为反映这些功能丰富的CRD所做的事情的通用术语。正如上面的列表所显示的，一些CRD开发人员已经将他们的产品标记为"网关"，以区别于Ingress控制器家族。

Gateway API

CRD 促进了特性开发，但是它们的生态系统已经发展到这样一个地步:许多供应商实现了不同的crd，但具有基本相似的特性，根据治理社区(SIG)，这是碎片化的标志。SIG Network社区主动使用Gateway api来驱动标准化。这一举措在本文中有很好的解释，并在KubeCon Europe 2021上进行了演示。

Gateway API开源项目引入了更多的资源类型，比如GatewayClass、Gateway、HTTPRoute和TCPRoute。目标是为网关功能定义有表现力的、可扩展的和面向角色的接口，但仍然让不同的开发人员将他们的技术插入Kubernetes。Gateway API在2021年2月之前曾被命名为"服务API"，它允许供应商公开高级功能，如流量分割、RBAC和TCP路由。对用户来说，Gateway API抽象了实现细节，从而更容易在技术之间切换，从而减少了厂商锁定。2022年7月，Gateway API项目发布了v0.5.0，并进入beta版。

从标准Ingress到Gateway API的过程表明了一个方向上的转变:标准Ingress最初是一个草率的标准化，导致了各种Ingress crd不可避免的碎片化。即将到来的Gateway API标志着重新标准化的谨慎努力，在一致性和可扩展性之间取得平衡。

这个领域的竞争已经开始。最值得注意的是，Envoy 团队在2022年5月引入了Envoy Gateway项目，合并了contour和Emissary项目，作为Gateway API的认可。然而，从生产准备的角度来看，大多数Gateway API实现仍处于开发的早期阶段。尽管有雄心勃勃的愿景，Gateway api还没有达到生产使用的成熟级别。

建议

Service 对象缺乏流量控制，应避免单独使用。未来的Gateway API尚未成熟，使得标准Ingress 和Ingress CRD成为控制入口流量的唯一两个可行选项。在撰写本文时，建议如下:

- 如果需求未定义，则使用标准的入口控制器启动POC。
- 当您发现特定的需求，如路由配置委派、TCP路由和最小化注释时，请探索Ingress CRD市场但要警惕技术锁定。
- 如果您对服务网格有需求，那么选择时应该考虑与服务网格兼容的上下文。

最后但并非最不重要的是，关注Gateway API项目的最新进展，并为供应商中立制定策略。

欢迎关注我的公众号“ **云原生拓展** ”，原创技术文章第一时间推送。