

运行容器组的节点，容器组的管理系统和方法

- 5 本申请要求于 2021 年 8 月 9 日提交中国国家知识产权局、申请号为 202110910594.X、
申请名称为“运行容器组的节点，容器组的管理系统和方法”的中国专利申请的优先权，其
全部内容通过引用结合在本申请中。

技术领域

本申请涉及云计算领域，且特别涉及一种运行容器组的节点，容器组的管理系统和方法。

背景技术

- 10 服务网格（service mesh）技术将微服务架构（microservice architecture）的分布式
应用中非功能性的针对流量的服务治理逻辑从业务进程中剥离到边车（sidecar）容器中，以
无侵入的方式提供服务间的连接、安全、流控、灰度发布、观测能力，实现业务轻量化和服
务治理基础设施化。另外，由于服务网格技术是基于传统互联网协议（internet protocol，
IP）网络之上的应用网络技术。因此，在服务网格技术中，服务之间的发现与路由不再直接
15 基于 IP 地址，而是基于服务的元数据信息（包括但不限于服务名称、版本等）。

随着用户需求的发展，微服务的规模和调用复杂度快速增长。如何在持续运行阶段高效的
治理微服务、降低运维成本，是服务网格技术演进的一个重要问题。

发明内容

- 20 本申请实施例提供了一种运行容器组的节点，容器组的管理系统和方法，可以为业务容
器组选择边车，以对业务容器组进行更优的流量管理。

- 第一方面，本申请实施例提供了一种运行容器组的节点，该节点运行有连接控制模块、
边车集群、和第一业务容器组，边车集群包括至少两个边车，其中，连接控制模块，用于接
收与该节点相连的控制台发送的边车分配策略，根据边车分配策略在边车集群中选择第一边
车，并将第一业务容器组发出的数据包转发至第一边车；第一边车，用于对第一业务容器组
25 发出的数据包进行流量管理。

在该方案中，连接控制模块可以按照控制台发送的边车分配策略，从至少两个边车中为
第一业务容器选择边车，并使用选择出的边车对第一业务容器组发出的数据包进行流量管理，
从而可以实现对第一业务容器组的灵活管理，可以为对第一业务容器组进行更优的流量管理，
保障第一业务容器组业务的高可用能力。

- 30 在一种可能的实现方式中，该节点还包括第二业务容器组；连接控制模块，还用于根据
边车分配策略在边车集群中选择第二边车，并将第二业务容器组发出的数据包转发至第二边
车；第二边车，用于对第二业务容器组发出的数据包进行流量管理。其中，第二边车和第一
边车可以为同一边车冗余，也可以为不同的边车。

- 也就是说，连接控制模块可以按照边车分配策略，从至少两个边车中为第二业务容器选
35 择边车，并使用选择出的边车对第二业务容器组发出的数据包进行流量管理，从而可以实现
对第二业务容器组的灵活管理，可以为对第二业务容器组进行更优的流量管理，保障第二业
务容器组的业务的高可用能力。

- 在一种可能的实现方式中，第一边车分配的硬件资源规格高于第二边车分配的硬件资源，
边车分配策略包括第一策略，第一策略用于指示第一业务容器组优先使用第一边车；连接控
40 制模块，具体用于根据第一策略在边车集群中选择第一边车。

也就是说，在该实现方式中，可以设置硬件资源规格高低不同的硬件资源，并在边车分

配策略指示第一业务容器组优先使用高硬件资源规格的边车的情况下，使用高硬件资源规格的边车对第一业务容器组发出的数据包进行流量管理，保障了第一业务容器组业务的服务质量。

5 在一种可能的实现方式中，该节点还包括第二业务容器组，边车分配策略还包括第二策略，第二策略用于指示第一边车的服务对象数量不超过上限值；连接控制模块，还用于确定第一边车的服务对象的数量，在第一边车的服务对象的数量不超过上限值的情况下将第二业务容器组发出的数据包转发至第一边车；第一边车，还用于对第一业务容器组发出的数据包和第二业务容器组发出的数据包同时进行流量管理。

10 也就是说，在该实现方式中，可以设置边车的服务对象数量的上限值，并在该边车当前的服务对象的数量不超过上限值的情况下，继续为该边车分配数据包，使得该边车对该数据包进行流量管理，从而在避免边车过载的同时，提高了边车的资源利用率。

在一种可能的实现方式中，连接控制模块，用于在第一边车失效后，从边车集群中选择第三边车或通知控制台在所述节点创建第三边车，将第一业务容器组发送的另一数据包转发至第三边车；第三边车，用于对第一业务容器组发出的另一数据包进行流量管理。

15 也就是说，在该实现方式中，第一边车失效后，可以重新为第一业务容器组选择第三边车，第三边车可以继续对第一业务容器组发出的数据包进行流量管理，从而保障了第一业务容器组业务的高可用能力。

在一种可能的实现方式中，第三边车是基于第一边车进行功能升级的新版本，或者第三边车是第一边车的复制版本。

20 也就是说，在该实现方式中，在第一边车失效后，可以为第一业务容器组选择与第一边车的功能相同的边车，或者选择比第一边车更高级的边车，以继续对第一业务容器组发出的数据包进行流量管理，从而保障了第一业务容器组业务的高可用能力。

在一种可能的实现方式中，第一边车，用于在对第一业务容器组发出的数据包进行流量管理之后，将数据包发送至后端容器组。

25 也就是说，在该实现方式中，第一边车可以将经过流量管理后的数据包发送至后端容器组，以调用后端容器组的服务或为后端容器组提供服务。

30 在一种可能的实现方式中，第一边车，还用于产生会话标识并发送会话标识至第一业务容器组和连接控制模块；连接控制模块，用于记录会话标识与后端容器组的对应关系；第三边车，用于从第一业务容器组获取会话标识并根据会话标识在连接控制模块记录的对应关系中确定后端容器组，在对第一业务容器组发出的另一数据包进行流量管理之后，将另一数据包发送至后端容器组。

35 也就是说，在该实现方式中，连接控制模块可以记录第一边车产生的会话标识和后端容器组的对应关系，第三边车可以根据该对应关系，将第一容器组发出的其他数据包进行流量管理后发送至该后端容器组，从而避免了同一容器组发出的不同数据包被不同边车发送至不同后端容器组的问题。

在一种可能的实现方式中，边车分配策略包括第三策略，第三策略用于指示边车集群中的边车的服务对象数量为 0 时被优先使用；连接控制模块，还用于确定第一边车的服务对象的数量，在第一边车的服务对象的数量为 0 的情况下将第一业务容器组发出的数据包转发至第一边车。

40 在一种可能的实现方式中，连接控制模块，还用于监控边车集群中每个边车的工作状态，在发现存在下线的边车时，发送下线的边车的信息至控制台。

也就是说,在该实现方式中,可以向控制台反馈下行的边车,以便控制台更新正在运行的边车,并据此制定边车分配策略,实现对容器组的有效管理。

在一种可能的实现方式中,流量管理包括:流量控制、流量安全以及流量观测。

在一种可能的实现方式中,节点为虚拟机、计算机或裸金属服务器。

5 第二方面,本申请实施例提供了一种容器组的管理系统,包括控制台和第一方面所提供的节点。

第三方面,本申请实施例提供了一种节点中的容器组的管理方法,节点运行有连接控制模块、边车集群、和第一业务容器组,边车集群包括至少两个边车,该方法包括:连接控制模块接收与节点相连的控制台发送的边车分配策略,根据边车分配策略在边车集群中选择第一边车,并将第一业务容器组发出的数据包转发至第一边车;第一边车对第一业务容器组发出的数据包进行流量管理。

在一种可能的实现方式中,节点还运行有第二业务容器组,该方法还包括:连接控制模块根据边车分配策略在边车集群中选择第二边车,并将第二业务容器组发出的数据包转发至第二边车;第二边车对所述第二业务容器组发出的数据包进行流量管理。

15 在一种可能的实现方式中,第一边车分配的硬件资源规格高于第二边车分配的硬件资源,边车分配策略包括第一策略,第一策略用于指示第一业务容器组优先使用第一边车;根据边车分配策略在边车集群中选择第一边车包括:根据第一策略在边车集群中选择第一边车。

在一种可能的实现方式中,节点还运行有第二业务容器组,边车分配策略还包括第二策略,第二策略用于指示第一边车的服务对象数量不超过上限值;方法还包括:连接控制模块确定第一边车的服务对象的数量,在数量不超过上限值的情况下将第二业务容器组发出的数据包转发至第一边车;第一边车对第一业务容器组发出的数据包和第二业务容器组发出的数据包同时进行流量管理。

20 在一种可能的实现方式中,该方法还包括:连接控制模块,在第一边车失效后,从边车集群中选择第三边车或通知控制台在节点创建第三边车,将第一业务容器组发送的另一数据包转发至第三边车;第三边车对第一业务容器组发出的另一数据包进行流量管理。

在一种可能的实现方式中,第三边车是基于第一边车进行功能升级的新版本,或者第三边车是第一边车的复制版本。

在一种可能的实现方式中,该方法还包括:第一边车在对第一业务容器组发出的数据包进行流量管理之后,将数据包发送至后端容器组。

30 在一种可能的实现方式中,该方法还包括:第一边车产生会话标识并发送会话标识至第一业务容器组和连接控制模块;连接控制模块记录会话标识与后端容器组的对应关系;第三边车从第一业务容器组获取会话标识并根据会话标识在连接控制模块记录的对应关系中确定后端容器组,在对第一业务容器组发出的另一数据包进行流量管理之后,将另一数据包发送至后端容器组。

35 在一种可能的实现方式中,边车分配策略包括第三策略,第三策略用于指示边车集群中的边车的服务对象数量为0时被优先使用;根据边车分配策略在边车集群中选择第一边车,并将第一业务容器组发出的数据包转发至第一边车包括:连接控制模块确定第一边车的服务对象的数量,在第一边车的服务对象的数量为0的情况下将所述第一业务容器组发出的数据包转发至所述第一边车。

40 在一种可能的实现方式中,该方法还包括:连接控制模块监控边车集群中每个边车的工作状态,在发现存在下线的边车时,发送下线的边车的信息至控制台。

在一种可能的实现方式中，流量管理包括：流量控制、流量安全以及流量观测。

第四方面，本申请实施例提供了一种运行容器组的节点，包括处理器和存储器，处理器用于执行存储器中存储的指令，以执行第二方面所提供的方法。

第五方面，本申请实施例提供了一种计算机可读存储介质，包括计算机程序指令，当计算机程序指令由计算设备集群执行时，计算设备集群执行如第一方面所提供的方法。

第六方面，本申请实施例提供了一种包含指令的计算机程序产品，当指令被计算设备集群运行时，使得计算设备集群执行如第一方面所提供的方法。

本申请实施例提供的运行容器组的节点，容器组的管理系统和方法，可以按照控制台发送的边车分配策略，从至少两个边车中为业务容器选择边车，并使用选择出的边车对该业务容器组发出的数据包进行流量管理，从而可以实现对该业务容器组的灵活管理，可以为对该业务容器组进行更优的流量管理，保障该业务容器组业务的高可用能力。

附图说明

- 图 1 为本申请实施例提供的一种容器组的管理系统的结构示意图；
- 图 2 为本申请实施例提供的一种运行容器组的节点的结构示意图；
- 图 3A 为本申请实施例提供的一种容器组管理框架示意图；
- 图 3B 为本申请实施例提供的一种容器组管理方法流程图；
- 图 4A 为本申请实施例提供的一种容器组管理框架示意图；
- 图 4B 为本申请实施例提供的一种容器组管理方法流程图；
- 图 5A 为本申请实施例提供的一种容器组管理框架示意图；
- 图 5B 为本申请实施例提供的一种容器组管理框架示意图；
- 图 5C 为本申请实施例提供的一种容器组管理方法流程图；
- 图 6A 为本申请实施例提供的一种容器组管理框架示意图；
- 图 6B 为本申请实施例提供的一种容器组管理框架示意图；
- 图 6C 为本申请实施例提供的一种容器组管理方法流程图；
- 图 7 为本申请实施例提供的一种容器组管理方法流程图；
- 图 8 为本申请实施例提供的一种节点的示意性框图。

具体实施方式

下面将结合附图，对本发明实施例中的技术方案进行描述。显然，所描述的实施例仅是本说明书一部分实施例，而不是全部的实施例。

在本说明书的描述中“一个实施例”或“一些实施例”等意味着在本说明书的一个或多个实施例中包括结合该实施例描述的特定特征、结构或特点。由此，在本说明书中的不同之处出现的语句“在一个实施例中”、“在一些实施例中”、“在其他一些实施例中”、“在另外一些实施例中”等不是必然都参考相同的实施例，而是意味着“一个或多个但不是所有的实施例”，除非是以其他方式另外特别强调。

其中，在本说明书的描述中，除非另有说明，“/”表示或的意思，例如，A/B 可以表示 A 或 B；本文中的“和/或”仅仅是一种描述关联对象的关联关系，表示可以存在三种关系，例如，A 和/或 B，可以表示：单独存在 A，同时存在 A 和 B，单独存在 B 这三种情况。另外，在本说明书实施例的描述中，“多个”是指两个或两个以上。

在本说明书的描述中，术语“第一”、“第二”仅用于描述目的，而不能理解为指示或暗示相对重要性或者隐含指明所指示的技术特征的数量。由此，限定有“第一”、“第二”的特征可以明示或者隐含地包括一个或者更多个该特征。术语“包括”、“包含”、“具有”及它们

的变形都意味着“包括但不限于”，除非是以其他方式另外特别强调。

微服务架构为一种面向服务的架构（service oriented architecture, SOA），其将复杂系统切分为多个小服务或者说应用。可以将该小服务或者说应用称为微服务。每个微服务负责实现一个独立的业务逻辑。微服务是围绕业务功能进行构建的，可以独立部署。微服务之间通过相互依赖，从而可以提供一系列的功能。微服务易于被理解和修改，带来了语言和框架选择灵活性。微服务可以运行在容器（container）中。其中，多个相互之间依赖性较高的容器可以构成一个容器组。例如，在 K8S（kubernetes）系统中，多个容器可以封装到一个 pod 中，即容器组为 pod。

在本申请实施例中，运行微服务的容器可称为业务容器。边车可通过容器或进程实现，在边车通过容器实现时，边车又可称为边车容器。

在微服务架构中，服务治理也称 SOA 治理（service oriented architecture governance, SOA governance），是用来管理微服务架构的采用和实现的过程。在基于微服务架构的服务网格中，服务治理可由边车执行。其中，在 K8S（Kubernetes）系统中，边车是设置有边车功能的容器，边车为业务容器提供流量服务治理功能。

在一个方案中，部署 pod 级别的边车。即，每个 pod 对应一个边车，该边车用于对该 pod 进行服务治理。在该方案中，不同的边车对应不同的 pod。一个边车的故障，仅影响该边车对应的 pod，而不影响其他 pod。可以说，该方案有助于保障业务的高可用能力。但在该方案中，若部署的 pod 的数量较多，部署的边车也较多，使得边车所占用的计算资源是不可忽略的，可能会导致链路时延大等问题。

在另一个方案中，部署节点级别的边车。即一个节点部署一个边车，用于对该节点上的多个 pod 进行服务治理。在该方案中，边车的故障，会影响节点上的多个 pod，从而对业务的高可用能力造成较大影响。

针对以上技术问题，本申请实施例提供了一种容器组的管理系统，如图 1 所示，该管理系统可以包括节点 100、节点 200 等多个节点，还可以包括与该多个节点相连的控制台 300。控制台 300 可以管理该多个节点，例如可以向多个节点中的任一节点发送边车分配策略，使得该节点可以根据边车分配策略为该节点中的业务容器组分配边车，以便对该业务容器组发出的数据包进行更优的流量管理，实现业务的高可用能力。

在一些实施例中，分配策略可以是节点所在数据中心的管理人员或租户在控制台 300 配置的，由此，可以实现对容器组的灵活管理。

接下来，以节点 100 为例，对本申请实施例涉及的图 1 所示管理系统中的节点进行示例介绍。

在一些实施例中，节点 100 可以为虚拟机（Virtual Machine, VM）。

在一些实施例中，节点 100 可以具有多种硬件组件，例如一个或多个处理器（例如中央处理单元（central processing unit, CPU）、一个或多个存储器等。节点 100 所具有的硬件组件可以赋予节点 100 数据计算以及数据存储能力。在一个例子中，节点 100 可以为计算机。在另一个例子中，节点 100 可以为裸金属服务器。

参阅图 2，节点 100 可以运行有连接控制模块 110、边车集群 120 以及业务容器组集群 130。其中，边车集群 120 可以包括边车 121、边车 122、边车 123 等至少两个边车。示例性的，边车又称为 sidecar。业务容器组集群 130 可以包括业务容器组 131。业务容器组 131 可以包括一个或多个容器，其中每个容器可以运行有应用或者微服务。连接控制模块 110 可以接收控制台 200 发送的边车分配策略 A。在业务容器组 131 具有流量管理需求时，连接控制

模块 110 可以根据该分配策略 A 在边车集群 120 中为业务容器组 131 选择边车，例如可以选择边车 121。之后，连接控制模块 110 可以将业务容器组 131 发出的数据包转发至边车 121。边车 121 可以对该数据包进行流量管理。

5 在一个例子中，流量管理具体可以为流量控制。其中，流量控制可以包括流量通过、流量拒绝、流量复制以及流量染色中的任一项或任意多项的组合。在一个例子中，流量管理具体可以为流量安全。其中，流量安全可以包括对数据包进行加密或解密等。在一个例子中，流量管理具体可以为流量观测。其中，流量观测可以包括在控制台 300 中绘制调用链等。

10 在一些实施例中，连接控制模块可以配置有边车列表。数据面代理列表可以包括处于运行状态的边车的信息。边车的信息可以包括边车的标识（例如，进程身份标识（identity，ID）、边车的监听端口。其中，边车的监听端口也可以称为边车的入方向端口。通过边车 121 的监听端口，业务容器组 131 发出的数据包可以被发送到边车 121，从而使得边车 121 可以对该数据包进行流量治理。

15 在一些实施例中，边车 121 分配的硬件资源规格高于边车 122 分配的硬件资源，边车分配策略 A 包括策略 A1，策略 A1 用于指示业务容器组 131 优先使用边车 121；由此，连接控制模块 110 可以根据策略 A1 在边车集群 120 中选择边车 121，使得边车 121 对容器组 131 发出的数据包进行服务治理（即流量管理）。该实施例具体实现过程将在下文实施例 2 中进行具体介绍，在此不再赘述。

20 在一些实施例中，继续参阅图 2，业务容器组集群 130 还可以包括业务容器组 132。在业务容器组 132 具有流量管理需求时，连接控制模块 110 还可以根据分配策略 A，在边车集群 120 中选择边车 122，并将业务容器组 132 发出的数据包转发至边车 122。之后，边车 122 可以对该数据包进行流量管理。

25 在一些实施例中，边车集群 120 中的同一个边车可以同时为多个业务容器组发出的数据包进行流量管理。例如，可以设定业务容器组集群 130 还可以包括业务容器组 132。边车分配策略 A 还包括策略 A2，策略 A2 可以用于指示边车的服务对象数量不超过上限值。边车的服务对象是指接受业务容器组的流量管理服务的业务容器组。也就是说，若边车对业务容器组发出的数据包进行流量管理，那么该业务容器组是该边车的服务对象。边车的上限值是指边车在同一时刻，能够进行流量管理的业务容器组的最大数量。其中，连接控制模块可以确定边车 121 的服务对象的数量，并在边车 121 的服务对象的数量不超过边车 121 上限值的情况下，将业务容器组 132 发出的数据包转发至边车 121。之后，边车 121 可以对业务容器组 30 131 发出的数据包和业务容器组 132 发出的数据包同时进行流量管理。

35 在一些实施例中，在边车失效后，例如边车发生崩溃（crash）或重启后，可以为该边车的服务对象重新选择边车，以便新选择的边车继续对失效的边车的服务对象继续进行流量管理。仍以边车 121 为例，如上所述，边车 121 失效前可以对业务容器组 131 发出的数据包进行流量管理。当边车 121 失效后，连接控制模块 110 可以为业务容器组 131 重新确定边车。重新确定的边车可以对业务容器组 131 发出数据包进行流量管理。示例性的，为业务容器组 131 重新确定边车，具体可以为连接控制模块 110 从边车集群 120 中重新选择为业务容器组 131 的数据包进行流量管理的边车。例如，可以设定边车集群 120 包括边车 123，连接控制模块 110 从边车集群 120 中选择边车 123，使边车 123 对业务容器组 131 发出的数据包进行流量管理。具体而言，可以将业务容器组 131 发出的数据包发送至边车 123，使得边车 123 对 40 该数据包进行流量管理。示例性的，为业务容器组 131 重新确定边车，具体可以为连接控制模块通过控制台 300 在节点 100 中创建边车 123，然后，将业务容器组 1131 发出的数据包发

送至边车 123, 使得边车 123 可以对该数据包进行流量管理。其中, 在节点 100 中创建边车 123 的具体过程将在下文的实施例 1 中进行具体介绍, 在此不再赘述。

在一些实施例中, 边车 123 可以在边车 121 的基础上进行功能升级的新版本边车。其中, 边车的版本升级过程将在下文进行具体介绍, 在此不再赘述。

5 在一些实施例中, 边车 123 是边车 121 的复制版本。也就是说, 控制台 300 可以再次在节点 100 中创建功能与边车 121 相同的边车。

10 在一些实施例中, 边车 121 在对业务容器组 131 发出的数据包进行流量管理之后, 将数据包发送至后端容器组。在一个例子中, 后端容器组可以是其他节点, 例如节点 200, 上的业务容器组。在一个例子中, 后端容器组可以是节点 100 上的除业务容器组 131 之外的其他业务容器组。

15 在这些实施例的一个示例中, 边车 121 还可以产生会话标识, 并将该会话标识发送至业务容器组 131 和连接控制模块 110。该会话标识可以是指业务容器组 131 和后端容器组之间的会话标识。连接控制模块 110 可以记录会话标识与后端容器组的对应关系。边车 123 在对业务容器组 131 发出的数据包进行流量管理时, 可以从业务容器组 131 获取该会话标识, 然后可以根据会话标识与后端容器组的对应关系以及从业务容器组 131 获取的会话标识, 确定后端容器组, 从而在边车 123 对业务容器组 131 发送的新的数据包进行流量管理之后, 将新的数据包发送至后端容器组。其中, 新的数据包可以是指边车 123 根据会话标识与后端容器组的对应关系以及从业务容器组 131 获取的会话标识, 确定后端容器组之后, 业务容器组 131 发送的数据包。具体将在下文实施例 4 中进行具体介绍, 在此不再赘述。

20 在一些实施例中, 边车分配策略 A 可以包括策略 A3, 策略 A3 用于指示边车集群 120 中的边车的服务对象数量为 0 时被优先使用。连接控制模块 110 可以确定边车 121 的服务对象的数量, 并且在边车 121 的服务对象的数量为 0 的情况下将业务容器组 131 发出的数据包转发至边车 121。其中, 服务对象是指边车所服务的业务容器组, 具体可以参考下文对实施例 2 介绍的内容实现, 在此不再赘述。

25 在一些实施例中, 连接控制模块 110 还用于监控边车集群 120 中每个边车的工作状态, 在发现存在下线的边车时, 发送下线的边车的信息至控制台 300。具体将在下文实施例 1 中进行具体介绍, 在此不再赘述。

30 本申请实施例提供的节点可以运行多个边车, 并根据分配策略, 在多个边车中为业务容器组选择边车, 然后, 利用选择出的边车对业务容器组发出的数据包进行流量管理, 从而可以对业务容器组的数据包进行更优的流量管理, 可实现业务的高可用能力。

接下来, 在具体实施例中, 对本申请实施例提供的节点以及容器组管理方案进行具体说明。

实施例 1

35 参阅图 3A, 本实施例提供了一种节点 100, 其可以运行有业务容器组集群 130, 包括业务容器组 131、业务容器组 132 等多个业务容器组。节点 100 还运行有 sidecar 集群 120, 包括 sidecar121、sidecar122、sidecar123 等多个 sidecar。示例性的, 该多个 sidecar 中每个 sidecar 可以绑定不同的硬件资源, 例如可以绑定一个或多个中央处理器 (central processing unit, CPU), 还可以绑定其他计算模块 (例如专门用于加密解密的硬件、图形处理器 (graphics processing unit, GPU) 等)。节点 100 还运行有连接管理模块 110。示例性的, 连接管理模块 110 可以包括协议栈 111 和 sidecar 启动管理器 112。

40 微服务节点 100 可以接入网络, 该网络可以包括后端容器组 210、后端容器组 220 等多

个后端容器组。在一个示例中，后端容器组可以运行于指该网络中除节点 100 之外的其他节点。在另一个示例中，后端容器组可以运行于节点 100。节点 100 中的业务容器组所发出的数据包在经过 sidecar 的流量管理后，可以发送到后端容器组，以调用后端容器组的服务或者为后端容器组提供服务。

- 5 其中，该网络还可以包括控制台 300。其中，控制台 300 可以接收管理人员或者说运维人员的操作，并响应该操作对网络中的节点 100 以及其他节点进行控制。例如，控制台 300 可以向 sidecar 发送服务列表，以进行服务列表同步，使得 sidecar 可以根据服务列表进行流量管理。再例如，控制台 300 可以向连接管理模块 110 发送边车分配策略 A，以便连接管理模块 110 根据边车分配策略 A，在 sidecar 集群中为业务容器组选择 sidecar。选择出的
10 sidecar 对业务容器组发出的数据包进行流量管理。

连接管理模块 110 可以采用多活方式管理节点 100 中的多个 sidecar。即该多个 sidecar 可以同时处于运行状态或者说工作状态。

- 示例性的，同一 sidecar 可以为同时为多个业务容器组的链路提供流量管理。在一个例子中，如图 3A 所示，sidecar121 可以同时为链路 1311、链路 1312 以及链路 1321 提供流量
15 管理。其中，链路 1311 和链路 1312 是源自业务容器组 131 的链路，链路 1321 是源自业务容器组 132 的链路。

示例性的，不同 sidecar 可以同时为同一业务容器组的不同链路提供流量管理。在一个例子中，如图 3A 所示，sidecar121 可以为源自业务容器组 132 的链路 1321 提供流量管理。sidecar123 可以为源自业务容器组 132 的链路 1322 提供流量管理。

- 20 其中，在本实施例中，为业务容器组的链路提供流量管理具体可以是指为业务容器组通过该链路发送的数据包提供流量管理。例如，对该数据包进行流量控制、流量安全、流量观测等。其中，流量控制可以包括流量通过、流量拒绝、流量复制以及流量染色中的任一项或任意多项的组合。

- 继续参阅图 3A，节点 100 还可以包括协议栈 111，其可以用于在业务容器组产生连接请求时，从 sidecar 集群 120 中为该连接请求分配 sidecar，从而建立该业务容器组和分配的
25 sidecar 之间的链路，使得分配的 sidecar 可以为该业务容器组通过该链路发送的数据包提供流量管理。

接下来，介绍为连接请求分配 sidecar 的方案。

- 协议栈 111 可以调用或者读取 sidecar 列表。示例性的，如图 3A 所示，sidecar 列表可以设置在协议栈 111 中。sidecar 列表可以包括处于运行状态的 sidecar 的信息。sidecar 的信息可以包括 sidecar 的标识（例如，进程身份标识（identity, ID）、sidecar 的版本信息、sidecar 的监听端口。其中，sidecar 的版本信息用于表征 sidecar 版本。在一个例子中，
30 sidecar 的版本信息可以为版本号。其中，版本号的大小代表了版本的高低。sidecar 的监听端口也可以成为 sidecar 的入方向端口。通过 sidecar 的监听端口，连接请求可以被发送到
35 该 sidecar，从而可以建立该连接请求的源业务容器组和该 sidecar 之间的链路。

示例性的，sidecar 的信息还可以包括 sidecar 的当前连接数。当前连接数用于表征 sidecar 当前所连接的链路的数量。以图 3A 所示的 sidecar121 为例，其连接链路 1311、链路 1312、链路 1321，因此，当前连接数为 3。

- 示例性的，sidecar 的信息还可以包括 sidecar 当前连接业务容器组的标识等。业务容器组的标识用于唯一标识业务容器组，由此，协议栈 111 可以通过 sidecar 当前连接业务容器组的标识，确定 sidecar 当前所连接业务容器组。其中，sidecar 连接业务容器组是指与
40

sidecar 之间具有链路业务容器组。例如，如图 3A 所示，业务容器组 131 属于 sidecar121 的当前连接业务容器组。

示例性的，sidecar 的信息还可以包括 sidecar 绑定的硬件资源信息（例如 CPU 信息、是否开启硬件加速等信息）等。在其他实施例中，sidecar 列表还可以包括其他信息，在此不再一一列举。

当业务容器组 P 产生连接请求 P1 时，该连接请求 P1 可以被发送至协议栈 111。协议栈 111 可以读取 sidecar 列表中记录的 sidecar 信息，并根据 sidecar 信息，为该连接请求 P1 分配 sidecar。接下来，进行具体说明。

在一个说明性示例中，协议栈 111 可以判断业务容器组 P 是否为 sidecar 当前连接业务容器组，即判断业务容器组 P 和 sidecar 列表中的 sidecar 之间已经存在了一条或多条链路。

若业务容器组 P 和 sidecar 列表中的 sidecar 之间已经存在了一条或多条链路。可以设定业务容器组 P 和 sidecarB1 之间已经存在了一条或多条链路。那么可以将 sidecarB1 分配给连接请求 P1。进而，协议栈 111 可以将连接请求 P1 发送至 sidecarB1。例如，协议栈 111 可以通过 sidecarB1 的监听端口发送连接请求 P1，使得连接请求 P1 可以被发送至 sidecarB1。sidecarB1 在接收到连接请求 P1 时，可以根据连接请求 P1，在 sidecarB1 和业务容器组 P 之间再建立一条链路。业务容器组 P 可以通过该链路向 sidecarB1 发送数据包。sidecarB1 接收到该数据包后，可以对该数据包进行流量管理。

若业务容器组 P 和 sidecar 列表中的 sidecar 之间不存在链路，即业务容器组 P 不是 sidecar 列表中任一 sidecar 的当前连接业务容器组。可以根据 sidecar 列表中各 sidecar 的当前连接数，确定当前连接数最少的 sidecar。可以设定 sidecar 列表中的 sidecarB2 的当前连接数最少。可以将 sidecarB2 分配给连接请求 P1。sidecarB2 在接收到连接请求 P1 时，根据基于连接请求 P1，在 sidecarB2 和业务容器组 P 之间再建立一条链路。业务容器组 P 可以通过该链路向 sidecarB2 发送数据包。sidecarB2 接收到该数据包后，可以对该数据包进行流量管理。

在一个说明性示例中，协议栈 111 在接收到连接请求 P1 时，可以判断 sidecar 列表中是否存在新版本的 sidecar。具体而言，可以判断 sidecar 列表中的所有 sidecar 的版本是否相同。若 sidecar 列表中一个或多个 sidecar 的版本高于其他 sidecar 的版本，那么可以确定该一个或多个 sidecar 为新版本的 sidecar。若 sidecar 列表中所有 sidecar 的版本均相同，那么可以确定 sidecar 列表中不存在新版本的 sidecar。

当 sidecar 列表中存在新版本的 sidecar 时，可以将新版本的 sidecar 分配给连接请求 P1。新版本的 sidecar 在接收到连接请求 P1 时，根据基于连接请求 P1，在新版本的 sidecar 和业务容器组 P 之间再建立一条链路。业务容器组 P 可以通过该链路向新版本的 sidecar 发送数据包。新版本的 sidecar 接收到该数据包后，可以对该数据包进行流量管理。

在一个说明性示例中，可以对节点 100 中的 sidecar 进行热升级。即可以在用户不感知的情况下，完成 sidecar 的升级。

接下来，结合图 3B，对实施例 1 提供的 sidecar 分配方案以及 sidecar 升级方案，进行更具体地说明。

运维人员可以在控制台 300 配置 sidecar 创建信息，用于在节点 100 中创建新的 sidecar。其中，sidecar 创建信息可以包括版本信息、硬件资源信息。

sidecar 创建信息还可以包括被替换 sidecar 的信息，用于指示新创建的 sidecar 用于替换节点 100 中的哪些 sidecar。其中，被替换 sidecar 可以是节点 100 中已处于运行状态

的 sidecar。被替换 sidecar 的信息可以包括被替换 sidecar 的标识以及被替换 sidecar 的剩余运行时长等。示例性的，当被替换 sidecar 有多个时，可以为不同的被替换 sidecar 设置不同的剩余运行时长，以防止或缓解多个被替换 sidecar 同时下线而导致的连接请求突增的问题。另外，当 sidecar 创建信息包括被替换 sidecar 的信息时，该 sidecar 创建信息也可以称为 sidecar 升级信息，即该 sidecar 创建信息可用于升级 sidecar。

经过上述配置后，控制台 300 可以执行步骤 401，向 sidecar 启动管理器 111321 发送 sidecar 创建信息。

sidecar 启动管理器 112 可以基于 sidecar 创建信息，创建新的 sidecar。当 sidecar 创建信息包括版本信息时，sidecar 启动管理器 112 创建符合版本信息的 sidecar。即创建的新 sidecar 的版本符合该版本信息。在创建 sidecar 的过程中，sidecar 启动管理器 112 还可以执行步骤 402，为新的 sidecar 分配监听端口。示例性的，sidecar 启动管理器 112 可以根据节点 100 中监听端口的占用情况，为新的 sidecar 分配监听端口。在分配监听端口时，可以将监听端口号分配给新的 sidecar，以完成监听端口的分配。sidecar 启动管理器 112 可以执行步骤 403，启动新的 sidecar。如此，在 sidecar 侧创建了新的 sidecar。

新的 sidecar 可以执行步骤 404，新的 sidecar 可以进行初始化，进入运行状态。

在新的 sidecar 执行了步骤 404 之后，sidecar 启动管理器 112 可以执行步骤 407，更新 sidecar 列表，并将更新后的 sidecar 列表发送至协议栈 111。其中，更新 sidecar 列表包括：将新的 sidecar 的信息（例如监听端口、版本信息、当前连接数、当前连接业务容器组的标识、硬件资源信息等信息）添加到 sidecar 列表中。示例性的，sidecar 启动管理器 112 可以更新后的 sidecar 列表可以携带在 Ebpf (extended BPF) map 中，然后，向协议栈 111 发送 Ebpf map，从而将更新后的 sidecar 列表发送至协议栈 111。

在一些实施例中，在新的 sidecar 执行了步骤 404 之后，sidecar 启动管理器 112 可以执行步骤 406，监听 sidecar 的运行状态。在一些实施例中，sidecar 启动管理器 112 可以建立其和 sidecar 之间的域套接字(domain socket)长连接，并通过域套接字长连接监听 sidecar 的运行状态。并根据监听到的结果，执行步骤 407 以及执行步骤 408。其中，在步骤 408，sidecar 启动管理器 112 可以向控制台 300 发送 sidecar 更新信息，其中包括新的 sidecar 的运行状态以及标识信息等。

另外，当 sidecar 创建信息包括被替换 sidecar 的信息时，在步骤 404，新的 sidecar 在步骤 404 中，还可以启动域套接字 (domain socket) 监听。并且，新的 sidecar 还可以执行步骤 405，新的 sidecar 通过域套接字连接被替换 sidecar。新的 sidecar 可以通过其和被替换 sidecar 之间的域套接字，监听被替换 sidecar 的运行状态，并向被替换 sidecar 发送下线指令，以指示被替换 sidecar 下线。其中，当被替换 sidecar 的信息包括被替换 sidecar 的剩余运行时长时，新的 sidecar 在完成初始化后开始计时，当计时达到剩余运行时长时，新的 sidecar 可以执行步骤 409，将被替换 sidecar 下线。具体而言，新的 sidecar 可以通过其和被替换 sidecar 之间的域套接字，向被替换 sidecar 发送下线指令。被替换 sidecar 可以响应该指令而下线。

另外，可以理解，节点 100 中的一个或多个 sidecar 可能会执行步骤 410，发生崩溃(crash)或重启。其中，为方便描述，可以将 sidecar 的崩溃或重启，统称下线。

sidecar 的下线可以导致 sidecar 和业务容器组之间的链路断开，以及导致 sidecar 和 sidecar 启动管理器 112 之间的域套接字断开。即在步骤 411 中，下线的 sidecar 和业务容器组之间的链路断开。在步骤 412 中，下线的 sidecar 和启动管理器 112 之间的域套接字断

开。当下线的 sidecar 和数据面启动管理器 112 之间的域套接字断开时, sidecar 启动管理器 112 可以确认 sidecar 下线, 进而执行步骤 413, 向控制台 300 发送 sidecar 更新信息, 其中包括下线的 sidecar 的标识信息等。

sidecar 启动管理器 112 在确认 sidecar 下线时, 还可以执行步骤 414, 更新 sidecar 列表, 并将更新后的 sidecar 列表发送至协议栈 111。其中, 更新 sidecar 列表可以包括将下线的 sidecar 从 sidecar 列表中删除或者列为不运行状态。在一些实施例中, 协议栈 111 可以执行步骤 415, 清理下线 sidecar 的连接信息。该连接信息可以包括下线 sidecar 历史连接的业务容器组等信息。

当业务容器组和 sidecar 之间的链路断开 (例如, 因 sidecar 的下线而断开) 时, 业务容器组可以产生连接请求, 以再次请求连接 sidecar。具体而言, 业务容器组可以在步骤 416 中, 向协议栈 111 发送连接请求。为方便描述, 可以将发出连接请求的业务容器组称为待连接业务容器组。协议栈 111 在接收到连接请求之后, 可以执行步骤 417, 为待连接业务容器组选择 sidecar。

在一些实施例中, 可以通过如下规则, 选择 sidecar。

首先, 判断 sidecar 列表中是否有存在新版本的 sidecar。具体而言, 可以判断 sidecar 列表中的所有 sidecar 的版本是否相同。若 sidecar 列表中一个或多个 sidecar 的版本高于其他 sidecar 的版本, 那么可以确定该一个或多个 sidecar 为新版本的 sidecar。当 sidecar 列表中存在新版本的 sidecar 时, 可以将新版本的 sidecar 作为选择的 sidecar。

当 sidecar 列表中不存在新版本的 sidecar 时, 可以判断待连接业务容器组和 sidecar 列表中的 sidecar 之间是否已经存在了一条或多条链路。若待连接业务容器组和 sidecar 列表中的 sidecarB1 之间已经存在了一条或多条链路。那么可以将 sidecarB1 作为选择的 sidecar。

若待连接业务容器组和 sidecar 列表中的 sidecar 之间不存在链路, 那么可以从 sidecar 列表中确定当前连接数最少的 sidecar。并将当前连接数最少的 sidecar 作为选择的 sidecar。

通过上述规则, 在选择出 sidecar 之后, 协议栈 111 可以执行步骤 418, 将连接请求发送给选择的 sidecar, 以便在该 sidecar 和待连接业务容器组之间创建链路, 使得该 sidecar 可以对业务容器组通过该链路发送的数据包进行流量管理。

在另一些实施例中, 可以通过图 2 所示实施例所提供的规则, 选择 sidecar。

实施例 1 提供的节点可以支持 sidecar 的多实例运行, 并可通过连接数对每个可用 sidecar 实例进行负载均衡。该节点还可以支持动态扩充 sidecar 实例及对已有实例热升级, 新连接优先使用热升级后的 sidecar, 并控制被替换 sidecar 的下线时间窗口, 降低瞬间连接切换导致的连接队列堆积问题。并且, 在该节点中, 协议栈与 sidecar 启动管理器可以自动管理每个 sidecar 的监听端口, 当 sidecar 因本身问题导致的宕机时, 该 sidecar 连接的业务容器组将被重新定位到其他可用 sidecar 上, 无需人工干预, 提升系统整体可用性。

35 实施例 2

本申请实施例提供了一种节点 100, 其运行有 sidecar 集群 120, 可以包括多种类型的 sidecar, 其中, 不同类型的 sidecar 的性能高低不同。可以设定 sidecar 集群 120 包括 sidecar121 和 sidecar122, 其中, sidecar121 为高性能 sidecar, sidecar122 为低性能 sidecar。

高性能 sidecar 的性能高于普通 sidecar 的性能。具体而言, 相对于普通 sidecar, 高性能 sidecar 配置了更多的硬件资源, 例如可以配置有更多数量的 CPU, 也可以配置有加速

硬件，等等。也就是说，高性能 sidecar 分配的硬件资源高于普通性能 sidecar。由此，高性能 sidecar 比普通 sidecar 具有更强大的数据处理能力。当高性能 sidecar 为业务容器组的链路提供流量管理时，可以保障该业务容器组的服务质量 (Quality of Service, QoS)。因此，对于具有服务质量要求的业务容器组，可以在其和高性能 sidecar 之间创建链路，使得高性能 sidecar 可以为该链路提供流量管理，以保障该业务容器组的服务质量。

运维人员可以通过控制台 300 配置优先级列表，优先级列表可以包括多种类型的连接请求。其中，不同类型的连接请求对应不同类型的 sidecar。例如，节点 100 设置有高性能 sidecar 和普通 sidecar。优先级列表可以包括第一类型的连接请求和第二类型的连接请求。其中，第一类型的连接请求对应高性能 sidecar，第二类型的连接请求对应普通性能 sidecar。

在一个例子中，每一类型的连接请求可以对应至少一种业务容器组标签 (label)。不同类型的连接请求对应不同的业务容器组标签。也就是说，每一类型的连接请求可以通过业务容器组标签来表征。业务容器组标签用于表示一种类型的业务容器组。其中，当业务容器组为 pod 时，业务容器组标签为 pod label。优先级列表记录了连接请求对应的业务容器组标签和 sidecar 类型的对应关系。例如，一类型的连接请求对应业务容器组标签 Q1 和业务容器组标签 Q2，并且该类型的连接请求对应高性能 sidecar。那么，优先级记录列表可以记录业务容器组标签 Q1 和高性能 sidecar 的对应关系，也记录了业务容器组标签 Q2 和高性能 sidecar 的对应关系。

在一个例子中，每一类型的连接请求可以对应至少一种服务类型。不同类型的连接请求对应不同的服务类型。也就是说，每一类型的连接请求可以通过服务类型来表征。优先级列表记录了连接请求对应的服务类型和 sidecar 类型的对应关系。例如，一类型的连接请求对应服务类型 S，并且该类型的连接请求对应高性能 sidecar。那么，优先级记录列表可以记录服务类型 S 和高性能 sidecar 的对应关系。

参阅图 4A，节点 100 还运行有连接控制模块 110。连接控制模块 110 可以包括协议栈 111 和控制面代理 113。

参阅图 4A，控制台 300 可以将配置的优先级列表发送至控制面代理 113。控制面代理 113 可以保存该优先级列表，以及将优先级列表发送至协议栈 111。示例性的，控制面代理可以通过 Ebpf map 将优先级列表发送至协议栈 111。

协议栈 111 还配置有 sidecar 列表，该 sidecar 列表记录有节点 100 的 sidecar 的类型、sidecar 监听端口等信息。其中，sidecar 列表的获取以及更新方式可以参考上文对实施例 1 的介绍。其中，与实施例 1 不同的是，在实施例 2 中可以预先配置 sidecar 标识和 sidecar 类型的对应关系。由此，协议栈 111 可以根据 sidecar 标识和 sidecar 类型的对应关系，将 sidecar 的类型记录到 sidecar 列表中。

协议栈 111 还可以配置有业务容器组标签和业务容器组标识的对应关系。在一个例子中，业务容器组标识可以为 cgroup.ns (control group namespace)。其中，业务容器组标识用于表示一个业务容器组，业务容器组标签用于表示一种类型的业务容器组。也就是说，多个业务容器组具有多个业务容器组标识，其中，多个业务容器组和多个业务容器组标识一一对应。当该多个业务容器组为同一类型的业务容器组时，该多个业务容器组的业务容器组标签相同。其中，业务容器组标签与业务容器组标识的对应关系的确定方式将在下文对图 4B 所示实施例的介绍，在此不再赘述。

在一个说明性示例中，优先级列表记录了业务容器组标签和 sidecar 类型的对应关系。当协议栈 111 接收到业务容器组，例如业务容器组 131，发送的连接请求时，协议栈 111 可

可以根据该连接请求的源地址（例如源 IP 地址），确定业务容器组 131 的业务容器组标识。然后，可以根据业务容器组 131 的业务容器组标识，以及业务容器组标签和业务容器组标识的对应关系，确定业务容器组 131 的业务容器组标签。在确定业务容器组 131 的业务容器组标签后，可以根据优先级列表，确定业务容器组 131 的业务容器组标签对应的 sidecar 类型。

- 5 然后，协议栈 111 将该连接请求发送至确定出的 sidecar 类型的数据代理。例如，可以设定业务容器组 131 的业务容器组标签对应的 sidecar 类型为高性能 sidecar，则协议栈 111 可以将来自业务容器组 131 的连接请求发送至高性能 sidecar。使得高性能 sidecar 为来自业务容器组 131 的数据包提供流量管理。

- 10 在一个说明性示例中，优先级列表记录了服务类型和 sidecar 类型的对应关系。当协议栈 111 接收到业务容器组，例如业务容器组 132，发送的连接请求时，协议栈 111 可以确定该连接请求的目标服务。然后，可以根据该目标服务的服务类型以及优先级列表，确定该目标服务对应的 sidecar 类型。然后，协议栈 111 可以将该连接请求发送至确定出的 sidecar 类型的数据代理。例如，可以设定业务容器组 132 发送的连接请求的目标服务对应的 sidecar 类型为普通能 sidecar，则协议栈 111 可以将来自业务容器组 132 的连接请求发送至普通性能 sidecar。使得普通性能 sidecar 为来自业务容器组 132 的数据包提供流量管理。

接下来，结合图 4B，对实施例 2 提供的方案进行更具体地说明。

- 参阅图 4B，节点 100 中的业务容器组监听模块（例如，pod 监听模块（monitor））可以通过步骤 501 对业务容器组创建模块进行监听，以监听业务容器组创建。示例性的，业务容器组创建模块具体可以为 kubelet。业务容器组创建模块在步骤 502 中，可以创建业务容器组。如此，业务容器组监听模块通过步骤 503，可以检测到业务容器组创建事件。通过业务容器组创建事件，业务容器组监听模块可以获取创建的业务容器组的业务容器组标识和业务容器组标签。之后，业务容器组监听模块可以执行步骤 504，建立业务容器组标识和业务容器组标识的对应关系。业务容器组监听模块可以通过步骤 505，将业务容器组标签和业务容器组标识的对应关系发送至协议栈 111。示例性的，业务容器组监控模块可以通过 Ebpf map 将业务容器组标签和业务容器组标识的对应关系发送至协议栈 111。

运维人员可以在控制台 300 配置优先级列表。优先级列表具体可以参考上文所述，在此不再赘述。控制台 300 可以通过步骤 507，向控制面代理 113 发送优先级列表。控制面代理 113 可以通过步骤 508，向协议栈 111 发送优先级列表。协议栈 111 可以通过步骤 509，保存优先级列表。

- 30 协议栈 111 可以通过步骤 510，接收业务容器组发送的连接请求。为方便表述，可以将发送连接请求的业务容器组称为待连接业务容器组。协议栈 111 可以解析连接请求的源地址，并在步骤 511 中根据连接请求的源地址，确定业务容器组标识。接着，协议栈 111 可以在步骤 512 中，根据确定的业务容器组标识，确定业务容器组标签。进而，协议栈 111 可以在步骤 513 中，根据优先级列表，确定业务容器组标签对应的 sidecar 类型，从而可以确定待连接业务容器组对应的 sidecar 类型。

协议栈 111 可以在步骤 514 中，为待连接业务容器组选择 sidecar。

若优先级列表为空，即没有配置连接请求和 sidecar 类型的对应关系。则从 sidecar 列表中，选择当前连接数最少的 sidecar。

- 40 若高性能 sidecar 的当前连接数为零，且待连接业务容器组不对应高性能 sidecar，则从 sidecar 列表中，选择当前连接数最少的 sidecar。也就是说，在高性能 sidecar 没有连接业务容器组的情况系，即使待连接业务容器组对不对应高性能 sidecar，也可以允许高性

能 sidecar 为待连接业务容器组提供流量管理,从而可以在高性能 sidecar 空闲时,提升 sidecar 整体资源利用率。

若待业务容器组对应高性能 sidecar,则选择高性能 sidecar。

- 5 之后,协议栈111可以执行步骤515,将连接请求发送至选择的sidecar。以便在该sidecar和待连接业务容器组之间创建链路,使得该 sidecar 可以对业务容器组通过该链路发送的数据包进行流量管理。

实施例2提供的方案在实施例1支持多活高可用的情况下,支持使用不同性能的 sidecar 对不同类型的业务容器组或目标服务,进行流量管理,从而保证某些对服务质量要求较高的连接请求使用独立或较多硬件资源,并可以在高性能 sidecar 空闲时,提升 sidecar 的整体资源利用率。

10

实施例3

在服务网格系统中,sidecar 需要感知系统中的服务变更,并获取因服务变更而发生更新的服务列表。其中,服务变更可以包括服务上线、服务下线等。

- 15 在一种方案中,参阅图5A,服务网格系统中的节点M、节点N等各节点的 sidecar 和服务网格系统中的控制面之间建立套接字(socket)长连接。在后端容器组发生服务变更时,控制面可以产生服务变更消息。该服务变更消息可用于描述后端容器组发生的服务变更,或者服务变更消息包括因后端容器组发生的服务变更而发生更新的服务列表。控制面通过其和各个 sidecar 之间的长连接,将服务变更消息发送至各 sidecar。其中,后端容器组可以是指能够响应节点M或节点N的服务调用而提供相应服务的容器组。

- 20 在该方案中,若服务网格系统中节点数量的增加,控制面的连接数将大大增加。服务网格系统中的每个服务的下线或下线均可引起服务变更,进而使得控制面产生服务变更消息。其中,控制面需要轮询其连接的所有 sidecar,并进行服务变更消息的发送。当控制面的连接数非常大时,轮询 sidecar 以及发送服务变更消息将消耗控制面大量的计算资源并对网络形成较大压力。另外,控制面是按照 sidecar 的启动顺序或某个随机顺序,进行轮询和服务变更消息发送的。这可能导致同一节点内不同 sidecar 接收服务变更消息的时间相差较大。进而导致同一节点内 sidecar 感知到的服务实例不一致的时间窗口变大,如此,可能引发业务异常。

并且控制面的套接字文件描述符数量也有限。

因此,在该方案中,上述限制导致了服务网格系统难以扩大规模。

- 30 本实施例提供了一种方案,参阅图5B,控制台300可与各节点的控制面代理建立连接(例如套接字长连接)。如此,控制台300可以将服务变更消息发送至各节点的控制面代理。例如,控制面将服务变更消息发送至节点100的控制面代理113和节点200的控制面代理213。然后,控制面代理113可以将服务变更消息发送至节点100中的 sidecar121、sidecar122等各个 sidecar。控制面代理213可以将服务变更消息发送至节点200中的 sidecar221、sidecar222等各个 sidecar。
- 35

- 如此,控制面的连接数将大大减少。并且当服务网格中发生服务变更时,控制面在节点层级上进行轮询和服务变更消息的发送,降低了控制面计算资源的消耗以及对网络的压力。并且,节点的控制面代理接收到服务变更消息后,通过节点内部通信机制,将服务变更消息发送至该节点中各 sidecar,如此,大大降低了同一节点内 sidecar 所感知到的服务实例不一致的时间窗口。
- 40

另外,在本实施例中,当控制面和控制代理之间的网络发生故障时,控制面代理可以作

为离线服务中心,为所在节点内的 sidecar 提供的读取功能,实现 sidecar 和控制侧的通信。其中,控制侧包括控制面代理和控制面。

接下来,结合图 5C,对实施例 3 提供的方案进行更具体地说明。

参阅图 5C,控制台 300 可以通过步骤 601,对后端容器组进行服务变更监听。并且在步骤 602 中,建立控制台 300 和节点 100 内的控制面代理 113 之间的连接。该连接可以为套接字长连接。在步骤 603 中,控制面代理 113 可以和其所在节点内的 sidecar 建立连接。

当控制台 300 在感知到后端容器组发生服务变更时,可以在步骤 604 中产生服务变更消息,并通过步骤 605,将服务变更消息发送至控制面代理。

控制面代理在接收到服务变更消息后,可以通过步骤 606,向 sidecar 发送服务变更消息。sidecar 在步骤 607 中,可以根据服务变更消息更新路由信息。

实施例 4

客户端第一次向服务端请求会话(session)时,服务端会为客户端创建一个会话,并通过特殊算法算出的一个会话标识(session identity, sessionID),用于标识该会话。服务端可以向客户端返回该会话标识。客户端可以将会话标识保存在本地 cookie 中。当客户端再次访问对服务端时,可将会话标识发送至服务端。服务端会再次使用该会话标识对应的会话为客户端提供相应服务。

其中,在服务网格系统中,客户端为可调用服务的业务容器组。服务端也可以称为后端容器组,其可以为业务容器组提供的服务。服务端可以包括多个服务实例,并使用其中的一个或多个服务实例为客户端提供服务。

参考图 6A,基于实施例 1 提供的方案,业务容器组 C 可以通过链路 C1 和 sidecar M1 连接,通过链路 C2 和 sidecar M2 连接。业务容器组 C 可以通过链路 C1 发送对后端容器组的连接请求,该连接请求可以携带会话标识 E。sidecar M1 接收到该连接请求后,可以提取其中的会话标识 E,然后根据会话标识 E,使用哈希(hash)算法,确定路由策略,以便按照该路由策略,将连接请求发送至服务实例。例如,可以设定该路由策略将该连接请求发送至后端容器组中的服务实例 D1。

同理,业务容器组 C 也可以通过链路 C2 发送对后端容器组的连接请求,该连接请求可以携带会话标识 E。sidecar M2 接收到该连接请求后,可以提取其中的会话标识 E,然后根据会话标识 E,使用哈希(hash)算法,确定路由策略,以便按照该路由策略,将连接请求发送至服务实例。例如,可以设定该路由策略将该连接请求发送至后端容器组中的服务实例 D2。

可以理解,在后端容器组发送服务变更时,不同 sidecar 接收服务变更消息的时间不一致,可能导致不同 sidecar 在进行哈希计算时,采用的哈希环大小不一致。例如,sidecar M1 在接收到服务变更消息之前,进行哈希计算;sidecar M2 在接收到服务变更消息之后,进行哈希计算;那么 sidecar M1 和 sidecar M2 分别进行哈希计算时,所采用的哈希环大小不一致,从而产生不同的路由策略。也就是说,业务容器组 C1 发起的不同连接请求,可能会被发送至不同的服务实例,进而可能引发业务异常。

实施例 4 提供了一种方案,在不同 sidecar 为同一业务容器组的不同链路提供流量管理的情况下,可以将该业务容器组通过不同链路发生的连接请求,发送至同一服务实例。接下来,结合图 4B 和图 4C,对该方案进行示例说明。

该方案可以应用于图 6B 所示的容器组管理系统,包括节点 100、后端容器组 210。其中,节点 100 可以包括 sidecar121 和 sidecar122。示例性的,节点 100 还可以包括连接控制模块 110。业务容器组 131 可以通过链路 1311 和 sidecar121 连接,通过链路 1313 和 sidecar122

连接。后端容器组可以配置有服务实例 G1、服务实例 G2 等多个服务实例。

参阅图 6B, 节点 100 可以配置有会话标识 F 和服务实例的映射关系。其中, 当一 sidecar, 例如 sidecar121, 根据会话标识 F, 进行哈希计算, 得到的路由策略所对应的服务实例时, sidecar121 可以建立会话标识 F 和该服务实例的映射关系。sidecar121 可以向 sidecar122 分享该映射关系。示例性的, sidecar121 可向连接控制模块 110 发送该映射关系。sidecar122 可以读取连接控制模块 110 中的会话标识 F 和服务实例的映射关系, 也可以将该映射关系保存在本地缓存中。

当 sidecar121 接收到业务容器组 131 通过链路 1311 发送的包含会话标识 F 的连接请求时, 可以按照会话标识 F 和服务实例的对应关系, 确定服务实例。进而将该连接请求路由到该服务实例。当 sidecar122 接收到业务容器组 131 通过链路 1313 发送的包含会话标识 F 的连接请求时, 可以按照会话标识 F 和服务实例的对应关系, 确定服务实例。进而将该连接请求路由到该服务实例。如此, 在不同 sidecar 为同一业务容器组的不同链路提供流量管理的情况下, 可以将该业务容器组通过不同链路发生的连接请求, 发送至同一服务实例。

接下来, 结合图 6C, 对实施例 4 提供的方案进行更具体地说明。

如图 6C 所示, 业务容器组 131 可以通过步骤 701 向后端容器组 210 发送会话请求。后端容器组 210 可以在步骤 702 中, 响应该会话请求为业务容器组 131 分配会话标识 F, 并通过步骤 703, 将会话标识 F 发送至业务容器组 131。业务容器组 131 在步骤 704 中, 可以保存会话标识 F。

业务容器组 131 可以通过步骤 705, 向 sidecar121 发送包含会话标识 F 的连接请求。

在一个说明性示例中, sidecar121 在接收到包含会话标识 F 的连接请求时, 可以通过步骤 706a, 在连接控制模块 110 中查询会话标识 F 对应的服务实例。

在一个说明性示例中, sidecar121 在接收到包含会话标识 F 的连接请求时, 可以通过步骤 706b, 在 sidecar121 的本地缓存中查询会话标识 F 对应的服务实例。也就是说, 在该示例中, 会话标识 F 和服务实例的对应关系可以被保存在 sidecar121 的本地缓存中。

业务容器组 131 可以通过步骤 707, 向 sidecar122 发送包含会话标识 F 的连接请求。

在一个说明性示例中, sidecar122 在接收到包含会话标识 F 的连接请求时, 可以通过步骤 708a, 在连接控制模块 110 中查询会话标识 F 对应的服务实例。

在一个说明性示例中, sidecar122 在接收到包含会话标识 F 的连接请求时, 可以通过步骤 708b, 在 sidecar122 的本地缓存中查询会话标识 F 对应的服务实例。也就是说, 在该示例中, 会话标识 F 和服务实例的对应关系可以被保存在 sidecar122 的本地缓存中。

继续参阅图 6C, sidecar121 可以执行步骤 709, 若未查询到服务实例, 则根据会话标识 F 进行哈希运算, 得到对应于会话标识 F 的服务实例。也就是说, 通过上述步骤 706a 和步骤 706b, 均未查询到对应于会话标识 F 的服务实例。这说明在 sidecar122 执行步骤 706a 时, 连接控制模块 110 中没有保存会话标识 F 和服务实例的对应关系; 以及 sidecar121 在执行步骤 706b 时, sidecar121 本地缓存中没有保存会话标识 F 和服务实例的对应关系。如此, sidecar121 根据会话标识进行哈希运算, 以得到对应于会话标识 F 的服务实例。

接着, sidecar121 可以执行步骤 710, 创建会话标识 F 和服务实例的映射关系, 并向连接控制模块 110 发送会话标识 F 和服务实例的映射关系。连接控制模块 110 在步骤 711 中, 可以保存会话标识 F 和服务实例的映射关系。示例性的, sidecar121 可以执行步骤 712, 更新本地缓存, 以将会话标识 F 和服务实例的映射关系保存在本地缓存中。

并且, sidecar121 可以执行步骤 713, 根据服务实例, 创建连接。即将连接请求发送至

对应于会话标识 F 的服务实例，以建立业务容器组 131 和该服务实例之间的连接。

继续参阅图 6C，sidecar122 可以执行步骤 714，若未查询到服务实例，则根据会话标识 F 进行哈希运算，得到对应于会话标识 F 的服务实例。也就是说，通过上述步骤 708a 和步骤 708b，均未查询到对应于会话标识 F 的服务实例。这说明在 sidecar122 执行步骤 708a 时，
5 连接控制模块 110 中没有保存会话标识 E 和服务实例的对应关系；以及 sidecar122 在执行步骤 708b 时，sidecar122 本地缓存中没有保存会话标识 F 和服务实例的对应关系。如此，sidecar122 根据会话标识进行哈希运算，以得到对应于会话标识 F 的服务实例。

接着，sidecar122 可以执行步骤 715，创建会话标识 F 和服务实例的映射关系，并向连接控制模块 110 发送会话标识 F 和服务实例的映射关系。

10 连接控制模块 110 可以执行步骤 716，确定连接控制模块 110 中已存在会话标识 F 和服务实例的映射关系（在上述步骤 711 中保存了会话标识 F 和服务实例的映射关系），不再根据 sidecar122 发送的会话标识 E 和服务实例的映射关系进行更新。连接控制模块 110 还可以执行步骤 717，将在步骤 711 中保存的会话标识 F 和服务实例的映射关系发送至 sidecar122。sidecar122 可以执行步骤 718，更新本地缓存，以将从连接控制模块 110 接收的会话标识 F 和服务实例的映射关系保存在本地缓存中。
15

sidecar122 可以根据从连接控制模块 110 接收的会话标识 F 和服务实例的映射关系，确定对应于会话标识 F 的服务实例。然后，sidecar122 可以执行步骤 719，根据服务实例，创建连接。即将连接请求发送至对应于会话标识 F 的服务实例，以建立业务容器组 131 和该服务实例之间的连接。

20 另外，在一个说明性示例中，可以配置会话标识的生命周期。连接控制模块 110 可以在会话标识的生命周期结束时，确定该会话标识过期，并且执行步骤 720，清理过期的会话标识。其中，在步骤 720 中，具体清理过期的会话标识和服务实例的映射关系。连接控制模块 110 可以执行步骤 721a，向 sidecar121 发送会话标识清理命令。sidecar121 可以响应于会话标识清理命令，执行步骤 722a，清理缓存，以将过期的会话标识和服务实例的映射关系从本地缓存中清理掉。连接控制模块 110 可以执行步骤 721b，向 sidecar122 发送会话标识清理命令。sidecar122 可以响应于会话标识清理命令，执行步骤 722b，清理缓存，以将过期的会话标识和服务实例的映射关系从本地缓存中清理掉。
25

由此，通过实施例 4 提供的方案，在不同 sidecar 为同一业务容器组的不同链路提供流量管理的情况下，可以将该业务容器组通过不同链路发生的连接请求，发送至同一服务实例，
30 进而使得该服务实例可以通过不同的链路为该业务容器组提供服务。

接下来，基于上文所描述的容器组管理方案，对本申请实施例提供的一种节点中的容器组的管理方法进行介绍。可以理解的是，该方法是上文所描述的容器组管理方案的另一种表达方式，两者是相结合的。该方法是基于上文所描述的容器组管理方案提出，该方法中的部分或全部内容可以参见上文对容器组管理的描述。

35 该节点运行有连接控制模块、边车集群、和第一业务容器组，边车集群包括至少两个边车。参阅图 7，该方法包括如下步骤。

步骤 7001，连接控制模块接收与节点相连的控制台发送的边车分配策略，根据边车分配策略在边车集群中选择第一边车，并将第一业务容器组发出的数据包转发至第一边车。

步骤 7002，第一边车对第一业务容器组发出的数据包进行流量管理。

40 在一些实施例中，该节点还运行有第二业务容器组，该方法还包括：连接控制模块根据所述边车分配策略在所述边车集群中选择第二边车，并将所述第二业务容器组发出的数据包

转发至所述第二边车；所述第二边车对所述第二业务容器组发出的数据包进行流量管理。

在这些实施例的一个示例中，所述第一边车分配的硬件资源规格高于所述第二边车分配的硬件资源，所述边车分配策略包括第一策略，所述第一策略用于指示所述第一业务容器组优先使用所述第一边车；所述根据所述边车分配策略在边车集群中选择所述第一边车包括：

5 根据所述第一策略在边车集群中选择所述第一边车。

在一些实施例中，所述节点还运行有第二业务容器组，所述边车分配策略还包括第二策略，所述第二策略用于指示所述第一边车的服务对象数量不超过上限值；所述方法还包括：所述连接控制模块确定所述第一边车的服务对象的数量，在所述数量不超过所述上限值的情况下将所述第二业务容器组发出的数据包转发至所述第一边车；所述第一边车对所述第一业务容器组发出的数据包和所述第二业务容器组发出的数据包同时进行流量管理。

10

在一些实施例中，所述方法还包括：所述连接控制模块，在所述第一边车失效后，从所述边车集群中选择第三边车或通知所述控制台在所述节点创建所述第三边车，将所述第一业务容器组发送的另一数据包转发至所述第三边车；所述第三边车对所述第一业务容器组发出的另一数据包进行流量管理。

15

在这些实施例的一个示例中，所述第三边车是基于所述第一边车进行功能升级的新版本，或者所述第三边车是所述第一边车的复制版本。

在这些实施例的另一个示例中，所述方法还包括：所述第一边车在对所述第一业务容器组发出的数据包进行流量管理之后，将所述数据包发送至后端容器组。

20

在该示例的一个例子中，所述方法还包括：所述第一边车产生会话标识并发送所述会话标识至所述第一业务容器组和所述连接控制模块；所述连接控制模块记录所述会话标识与所述后端容器组的对应关系；所述第三边车从所述第一业务容器组获取所述会话标识并根据所述会话标识在所述连接控制模块记录的所述对应关系中确定所述后端容器组，在对所述第一业务容器组发出的另一数据包进行流量管理之后，将所述另一数据包发送至后端容器组。

25

在一些实施例中，所述边车分配策略包括第三策略，所述第三策略用于指示所述边车集群中的边车的服务对象数量为 0 时被优先使用；所述根据所述边车分配策略在边车集群中选择所述第一边车，并将所述第一业务容器组发出的数据包转发至所述第一边车包括：所述连接控制模块确定所述第一边车的服务对象的数量，在所述第一边车的服务对象的数量为 0 的情况下将所述第一业务容器组发出的数据包转发至所述第一边车。

30

在一些实施例中，所述方法还包括：所述连接控制模块监控所述边车集群中每个边车的工作状态，在发现存在下线的边车时，发送下线的所述边车的信息至所述控制台。

35

在一些实施例中，所述流量管理包括：流量控制、流量安全以及流量观测。

本申请实施例提供的容器组管理方法，可以按照控制台发送的边车分配策略，从至少两个边车中为业务容器选择边车，并使用选择出的边车对该业务容器组发出的数据包进行流量管理，从而可以实现对该业务容器组的灵活管理，可以为对该业务容器组进行更优的流量管理，保障该业务容器组业务的高可用能力。

40

值得注意的是，在本申请实施例中，边车也可以对从节点外或节点内其他业务容器组发送至与边车绑定的业务容器组的数据包进行流量处理，流量处理的策略也可以参照以上方式进行设置，本申请实施例对此不作限定。

参阅图 8，本申请实施例还提供了一种节点 800，包括处理器 810 和存储器 820，处理器 810 用于执行存储器 820 中存储的指令，使得节点 800 可以执行如图 7 所示的方法。

45

本申请实施例还提供了一种计算机可读存储介质，包括计算机程序指令，当所述计算机

程序指令由计算设备集群执行时，所述计算设备集群执行如图 7 所示的方法。

本申请实施例还提供了一种包含指令的计算机程序产品，当所述指令被计算机设备集群运行时，使得所述计算机设备集群执行如图 7 所示的方法。

可以理解的是，本申请的实施例中的处理器可以是中央处理单元 (central processing unit, CPU)，还可以是其他通用处理器、数字信号处理器 (digital signal processor, DSP)、
5 专用集成电路 (application specific integrated circuit, ASIC)、现场可编程门阵列 (field programmable gate array, FPGA) 或者其他可编程逻辑器件、晶体管逻辑器件，硬件部件或者其任意组合。通用处理器可以是微处理器，也可以是任何常规的处理器。可以理解的是，
10 在本申请的实施例中涉及的各种数字编号仅为描述方便进行的区分，并不用来限制本申请的实施例的范围。

权利要求

1. 一种运行容器组的节点，其特征在于，所述节点运行有连接控制模块、边车集群、和第一业务容器组，所述边车集群包括至少两个边车，其中，

5 所述连接控制模块，用于接收与所述节点相连的控制台发送的边车分配策略，根据所述边车分配策略在边车集群中选择第一边车，并将所述第一业务容器组发出的数据包转发至所述第一边车；

所述第一边车，用于对所述第一业务容器组发出的数据包进行流量管理。

2. 根据权利要求1所述的节点，其特征在于，还包括第二业务容器组；

10 所述连接控制模块，还用于根据所述边车分配策略在所述边车集群中选择第二边车，并将所述第二业务容器组发出的数据包转发至所述第二边车；

所述第二边车，用于对所述第二业务容器组发出的数据包进行流量管理。

3. 根据权利要求2所述的节点，其特征在于，所述第一边车分配的硬件资源规格高于所述第二边车分配的硬件资源，所述边车分配策略包括第一策略，所述第一策略用于指示所述第一业务容器组优先使用所述第一边车；

15 所述连接控制模块，具体用于根据所述第一策略在所述边车集群中选择所述第一边车。

4. 根据权利要求1至3任一项所述的节点，其特征在于，还包括第二业务容器组，所述边车分配策略还包括第二策略，所述第二策略用于指示所述第一边车的服务对象数量不超过上限值；

20 所述连接控制模块，还用于确定所述第一边车的服务对象的数量，在所述第一边车的服务对象的数量不超过所述上限值的情况下将所述第二业务容器组发出的数据包转发至所述第一边车；

所述第一边车，还用于对所述第一业务容器组发出的数据包和所述第二业务容器组发出的数据包同时进行流量管理。

5. 根据权利要求1至4任一项所述的节点，其特征在于，

25 所述连接控制模块，用于在所述第一边车失效后，从所述边车集群中选择第三边车或通知所述控制台在所述节点创建所述第三边车，将所述第一业务容器组发送的另一数据包转发至所述第三边车；

所述第三边车，用于对所述第一业务容器组发出的另一数据包进行流量管理。

30 6. 根据权利要求5所述的节点，其特征在于，所述第三边车是基于所述第一边车进行功能升级的新版本，或者所述第三边车是所述第一边车的复制版本。

7. 根据权利要求5或6所述的节点，其特征在于，

所述第一边车，用于在对所述第一业务容器组发出的数据包进行流量管理之后，将所述数据包发送至后端容器组。

8. 根据权利要求7所述的节点，其特征在于，

所述第一边车，还用于产生会话标识并发送所述会话标识至所述第一业务容器组和所述连接控制模块；

所述连接控制模块，用于记录所述会话标识与所述后端容器组的对应关系；

5 所述第三边车，用于从所述第一业务容器组获取所述会话标识并根据所述会话标识在所述连接控制模块记录的所述对应关系中确定所述后端容器组，在对所述第一业务容器组发出的另一数据包进行流量管理之后，将所述另一数据包发送至后端容器组。

9. 根据权利要求 1 或 2 所述的节点，其特征在于，所述边车分配策略包括第三策略，所述第三策略用于指示所述边车集群中的边车的服务对象数量为 0 时被优先使用；

10 所述连接控制模块，还用于确定所述第一边车的服务对象的数量，在所述第一边车的服务对象的数量为 0 的情况下将所述第一业务容器组发出的数据包转发至所述第一边车。

10. 根据权利要求 1 至 9 任一项所述的节点，其特征在于，

所述连接控制模块，还用于监控所述边车集群中每个边车的工作状态，在发现存在下线的边车时，发送下线的所述边车的信息至所述控制台。

15 11. 根据权利要求 1 至 10 任一项所述的节点，其特征在于，所述流量管理包括：流量控制、流量安全以及流量观测。

12. 根据权利要求 1 至 11 任一项所述的节点，其特征在于，所述节点为虚拟机、计算机或裸金属服务器。

13. 一种容器组的管理系统，其特征在于，包括控制台和权利要求 1 至 11 所述的节点。

20 14. 一种节点中的容器组的管理方法，其特征在于，所述节点运行有连接控制模块、边车集群、和第一业务容器组，所述边车集群包括至少两个边车，所述方法包括：

所述连接控制模块接收与所述节点相连的控制台发送的边车分配策略，根据所述边车分配策略在边车集群中选择第一边车，并将所述第一业务容器组发出的数据包转发至所述第一边车；

所述第一边车对所述第一业务容器组发出的数据包进行流量管理。

25 15. 根据权利要求 14 所述的方法，其特征在于，所述节点还运行有第二业务容器组，所述方法还包括：

所述连接控制模块根据所述边车分配策略在所述边车集群中选择第二边车，并将所述第二业务容器组发出的数据包转发至所述第二边车；

所述第二边车对所述第二业务容器组发出的数据包进行流量管理。

30 16. 根据权利要求 15 所述的方法，其特征在于，所述第一边车分配的硬件资源规格高于所述第二边车分配的硬件资源，所述边车分配策略包括第一策略，所述第一策略用于指示所述第一业务容器组优先使用所述第一边车；

所述根据所述边车分配策略在边车集群中选择所述第一边车包括：根据所述第一策略在边车集群中选择所述第一边车。

35 17. 根据权利要求 14 至 16 任一项所述的方法，其特征在于，所述节点还运行有第二业务

容器组，所述边车分配策略还包括第二策略，所述第二策略用于指示所述第一边车的服务对象数量不超过上限值；所述方法还包括：

所述连接控制模块确定所述第一边车的服务对象的数量，在所述数量不超过所述上限值的情况下将所述第二业务容器组发出的数据包转发至所述第一边车；

- 5 所述第一边车对所述第一业务容器组发出的数据包和所述第二业务容器组发出的数据包同时进行流量管理。

18. 根据权利要求 14 至 17 任一项所述的方法，其特征在于，所述方法还包括：

- 10 所述连接控制模块，在所述第一边车失效后，从所述边车集群中选择第三边车或通知所述控制台在所述节点创建所述第三边车，将所述第一业务容器组发送的另一数据包转发至所述第三边车；

所述第三边车对所述第一业务容器组发出的另一数据包进行流量管理。

19. 根据权利要求 18 所述的方法，其特征在于，所述第三边车是基于所述第一边车进行功能升级的新版本，或者所述第三边车是所述第一边车的复制版本。

20. 根据权利要求 18 或 19 所述的方法，其特征在于，所述方法还包括：

- 15 所述第一边车在对所述第一业务容器组发出的数据包进行流量管理之后，将所述数据包发送至后端容器组。

21. 根据权利要求 20 所述的方法，其特征在于，所述方法还包括：

所述第一边车产生会话标识并发送所述会话标识至所述第一业务容器组和所述连接控制模块；

- 20 所述连接控制模块记录所述会话标识与所述后端容器组的对应关系；

所述第三边车从所述第一业务容器组获取所述会话标识并根据所述会话标识在所述连接控制模块记录的所述对应关系中确定所述后端容器组，在对所述第一业务容器组发出的另一数据包进行流量管理之后，将所述另一数据包发送至后端容器组。

- 25 22. 根据权利要求 14 或 15 所述的方法，其特征在于，所述边车分配策略包括第三策略，所述第三策略用于指示所述边车集群中的边车的服务对象数量为 0 时被优先使用；

所述根据所述边车分配策略在边车集群中选择所述第一边车，并将所述第一业务容器组发出的数据包转发至所述第一边车包括：

所述连接控制模块确定所述第一边车的服务对象的数量，在所述第一边车的服务对象的数量为 0 的情况下将所述第一业务容器组发出的数据包转发至所述第一边车。

- 30 23. 根据权利要求 14 至 22 任一项所述的方法，其特征在于，所述方法还包括：

所述连接控制模块监控所述边车集群中每个边车的工作状态，在发现存在下线的边车时，发送下线的所述边车的信息至所述控制台。

24. 根据权利要求 14 至 23 任一项所述的方法，其特征在于，所述流量管理包括：流量控制、流量安全以及流量观测。

25. 一种运行容器组的节点，其特征在于，包括处理器和存储器，所述处理器用于执行所述存储器中存储的指令，以执行权利要求 14 至 24 任一项所述的方法。

26. 一种计算机可读存储介质，其特征在于，包括计算机程序指令，当所述计算机程序指令由计算设备集群执行时，所述计算设备集群执行如权利要求 14 至 24 中任一项所述的方法。
5 法。

27. 一种包含指令的计算机程序产品，其特征在于，当所述指令被计算机设备集群运行时，使得所述计算机设备集群执行如权利要求的 14 至 24 中任一项所述的方法。