基于表项按需下发的VPC快速上线技术设计方案

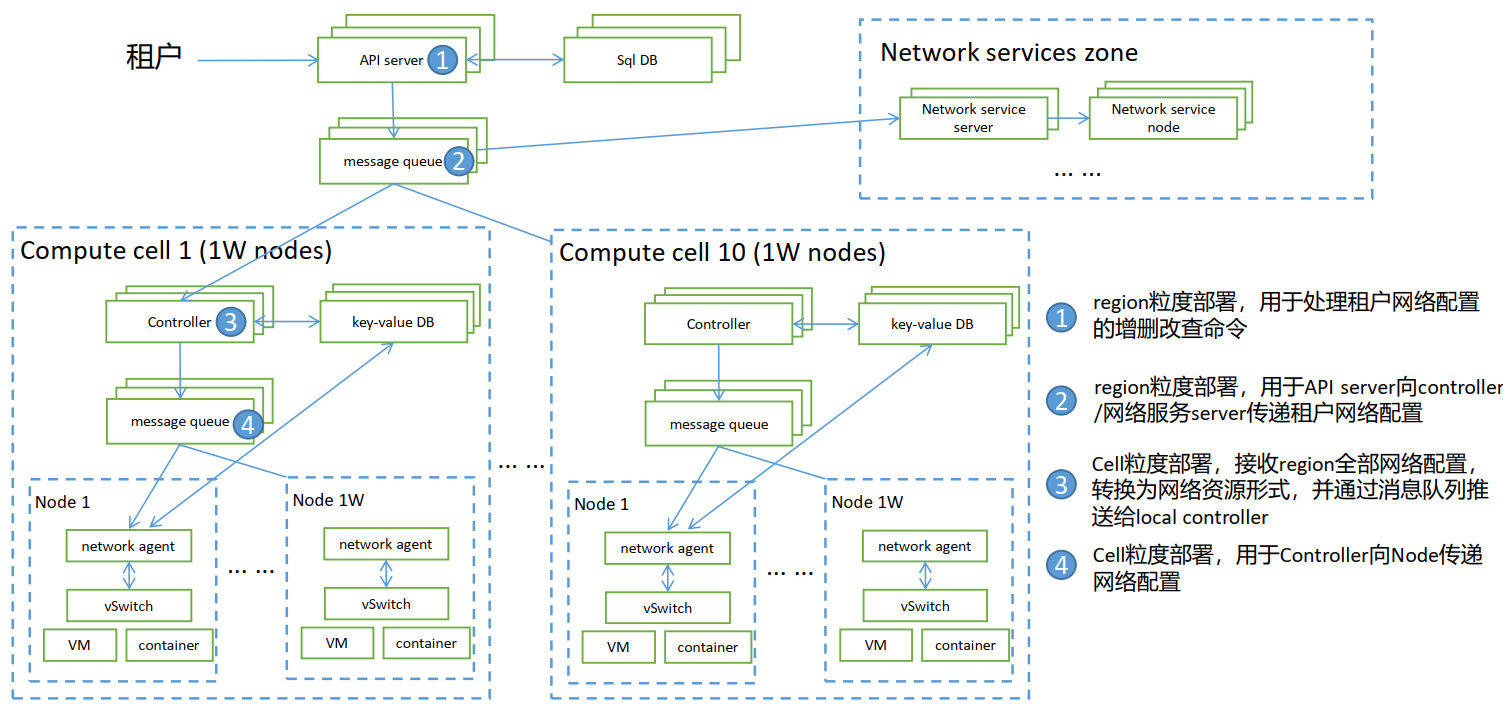
撰写人：赵功名、张敏利、凃化清、张钱宇

修改记录：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 版本 | 主要的修改 | 日期 | 完成人 |
| 1 | V4 | 完善项目需求、设计目标、总体设计 | 2020.3.14 | 张钱宇、赵功名 |
| 2 | V4 | 虚机上线概要设计 | 2020.3.15 | 张敏利、赵功名 |
| 3 | V4 | 虚机迁移概要设计 | 2020.3.15 | 凃化清、赵功名 |
| 4 | V5 | 新增模块划分章节 | 2020.3.28 | 张敏利、凃化清、张钱宇、赵功名 |
| 5 | V6 | 新增工作流程章节 | 2020.4.03 | 凃化清、张钱宇、张敏利、赵功名 |
| 6 | V7 | 新增多网关节点匹配及大流检测部分 | 2020.04.28 | 张敏利、凃化清、张钱宇、赵功名 |
| 7 | V8 | 新增direct path设计 | 2020.05.23 | 张敏利、凃化清、赵功名 |

基于表项按需下发的VPC快速上线技术设计方案

1. **项目需求**



图一：华为云虚拟网络平台现状架构示意图

目前，华为云虚拟网络平台在上线虚拟机时采用转发流表项全部预下发的方式，并且为每个东西向的计算节点实现互访一跳即达。当大客户有超大规模VPC的需求时，上线时需要在宿主机上下发大量流表，进而导致VPC的虚拟机上线速度慢。所以对于超大规模VPC网络上线而言，华为云虚拟网络平台面临的挑战有以下几点：（1）下发的流表项数量没有和VPC规模大小解耦，随着VPC规模增大，上线速度显著变慢，无法收敛上线时间；（2）若多个新上线虚机处在同一宿主主机上，则该高密服务器上流表数巨大，引入服务器性能过载或故障的风险。（3）虚拟机迁移时，其他虚机上到此虚机的表项均需要进行修改，因此需要同步全部计算节点，增加了控制器负载和迁移延迟。

华为云虚拟网络平台希望采用对每个计算节点表项按需下发的方式来解决上面的挑战。计算节点表项按需下发需要满足：（1）在虚拟机上线时只部署少量必要流表，其余表项在虚拟机上线后逐步按需下发，以提高虚拟机上线速度。（2）上线时下发的流表项大小与VPC规模解耦；（3）部署在宿主机上的表项数量大大减少，从而降低高密度服务器的过载和故障风险；（4）在虚拟机迁移时不需要同步全部计算节点。

1. **设计目标**

**1、总体目标**

为了提高超大规模VPC虚拟机上线/迁移速度，华为云网络平台希望按需下发流表项，并达成以下几个目标：（1）虚拟机上线速度得到提升，与VPC大小解耦合，并趋于固定值。（2）计算节点表项大大减少，降低高密度服务器的过载和故障风险。（3）虚拟机迁移时不需要同步全部计算节点，从而降低迁移延迟。

**2、技术难点**

（1）多个网关节点与计算节点的动态匹配：

① 节点与网关的比例不固定，主要根据网络的负载情况决定，但流量动态变化很大。

（2）网关节点扩缩容/单点故障时，计算节点和网关节点的映射关系如何修改？

（3）流量统计和分析

① 如何兼顾统计的准确性和效率。

② 如何选择部分大流进行一跳即达转发

1. **总体设计**

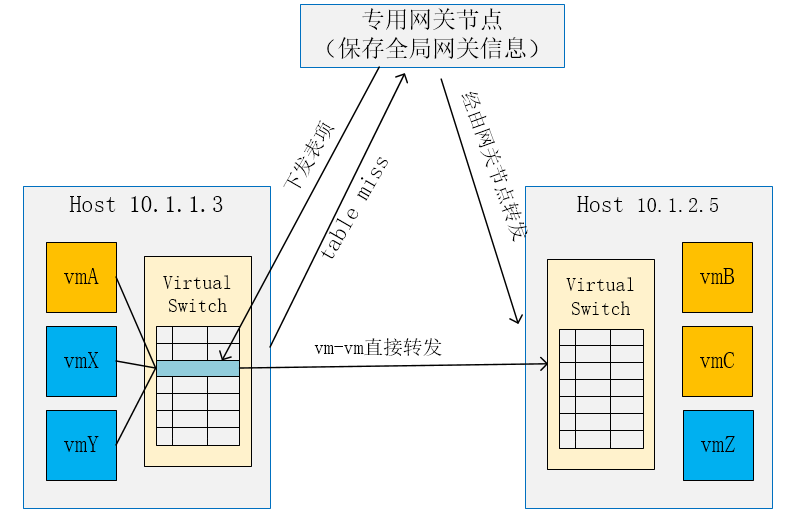
本方案主要通过三个设计思路提高虚机上线/迁移速度。(1)通过分级流表安装/转发模型，在网关节点保存全局表项信息，虚机上线时只在节点侧部署一条默认表项将流转发至网关节点，由网关节点进行查表转发；后续根据大小流检测结果，将大流的表项信息更新至宿主主机侧，实现大流的一跳即达。利用分级流表转发模型，减少减少宿主机上流表安装数量，降低计算节点过载风险和虚机上线时间。（2）通过基于中继转发的虚机迁移方案，在虚机迁移时，前期只更新网关节点，保持对其他虚机的透明。其他虚机发往该虚机的小流会转发至网关节点实现正确转发，其他虚机发往该虚机的大流依然发往原主机，通过在原主机ovs上安装相应流表，将这些流转发至虚机新的位置，后续逐步更新其他主机侧的流表信息，删除中继表项，通过分级流表和延迟更新方案，避免虚机迁移时同步全部计算节点，提高虚机迁移速度。（3）多计算节点与多网关节点的动态匹配。在大规模VPC网络中，单个网关节点无力响应所有计算节点，因此需要考虑多网关节点模型。利用基于bucket的ECMP模型，实现多计算节点与多网关节点的动态匹配，实现更为便捷的网关节点扩缩容。

1. **概要设计**

本文档针对虚机上线、虚机迁移等场景给出初步的设计思路。

**4.1 分级表项转发模型**

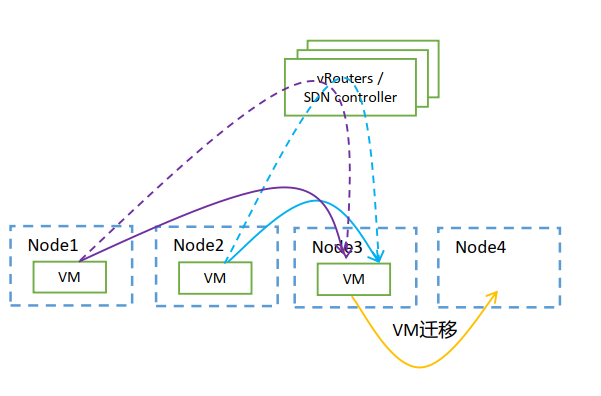
目前的虚机上线方案需要在计算节点侧转发流表全部预下发，导致在超大规模VPC的虚机上线时因需要在宿主主机安装大量流表，导致上线速度慢。而据调研，在云计算网络中超过98%的虚机对之间的通信量小于20kbps，超过83%的虚机对之间从不通信。因此，虚机侧只需下发少量必要的表项，我们将全局的路由表项保存在外置的专用网关节点上。在虚机上线时，其流表管控分为两个阶段。第一个阶段是，首先下发虚机相关配置，使之接入网络，然后配置一条默认路由以将新上线的虚机的流量转发至网关节点，由网关节点查询全局路由信息，实现转发操作。此时虚机已经上线成功，并可以和其他虚机进行通信。由于只为虚机配置了默认路由（转发至网关节点的表项），无需在虚机侧ovs上部署庞大的流表项，虚机的上线时间将大大缩短。但由于流量需要经过一个网关节点进行转发，降低了转发效率。针对这一点，在虚机成功上线后，第二个阶段是，我们将定期（比如每隔1分钟）在虚机侧查询其流量接收/发送信息（包含虚机对间发送的数据量），并筛选出通信量较大的虚机对（比如通信量最大的1万个虚机对），然后将这些虚机对的流表项更新至虚机侧的OVS，使虚机间频繁通信的流可以直接经过虚机侧的OVS进行转发，无需经过网关节点，从而提高了转发效率。并且因为只在宿主主机上下发小部分经常通信的虚机间的表项，从而减少了虚机侧ovs表项的数量，降低高密度服务器的故障率。如在图二中，专用网关节点保存了全局的流表信息，若主机10.1.1.3需要向主机10.1.2.5发送数据包时，首先在10.1.1.3主机侧流表查找是否存储目的主机的IP地址和MAC地址，如果发生table-miss就将数据包转发给专用网关节点，由网关节点进行转发；若存在匹配表项，则直接一跳即达转发。此外，网关节点定期进行大流检测并将大流信息更新至主机侧，保持大流的一跳即达转发。



图二、网关节点模型

**4.2 基于中继转发的虚机迁移方案**

针对虚机迁移场景，除了上述降低虚机上线时间的方案外，我们拟提出基于旧节点的中继转发方案来缩短迁移虚机的上线时间，其流表管控分为两个阶段。第一个阶段是，当发生虚机迁移事件后，我们并不需要在第一时间同步全部计算节点，而是在该虚机迁移之前所处主机侧OVS上部署一条表项将转发至该虚机的流全部转发至虚机迁移之后的位置，从而借助原位置实现数据转发，避免了对网络其他计算节点的表项修改操作，大大缩短了迁移虚机的上线时间。此外，网关节点上存储的全局表项信息针对该虚机进行更新，以实现将流导向新的虚机。第二个阶段是，在迁移虚机上线之后，逐步地实现其他虚机与新节点vm间的一跳互达。首先是根据新节点vm上收集的流统计信息，优先更新通信量大的vm间的表项，实现一跳即达，最终在给定时间内（比如1分钟内），完成所有关于迁移虚机的表项更新。由于在转发流的时候采用网关节点/分级流表安装转发的方式，只有少部分的vm所在的节点上的OVS有关于与迁移虚机通信的表项，由于网关节点更新在第一阶段已经完成，我们只要再完成这些少部分的vm的表项更新就完成了虚机迁移的全部表项更新。在这些表项更新完成之后，控制器将原虚机所在节点上的中继表项删除。



图三：基于中继转发的vm迁移方案

例如，在图三所示的虚机迁移场景中，若vm从node3迁移至node4，第一步是仅在node3上的ovs上部署一条表项，将目的地为该迁移vm的流转发至连接node4的端口，而不修改全局其他节点上的流表信息，并对与这四个节点相连接的网关节点针对node3上的虚机进行跟新。假设node1上的vm需要与该vm通信，并且其表项直接存储在node1的OVS上，则首先根据node1上的表项信息将该流转发至node3，然后根据node3上的表项信息将该流转发至node4，从而找到迁移后的新地址，完成通信。通过中继转发方式，避免了迁移初期对全部计算节点的同步，提高了迁移速度。若node2上的vm需要与迁移虚机进行通信，且其表项信息没有存储在node2的ovs上，而是存储在与node2相连的网关节点上，此时按照网关节点/分级流表转发的方式，将流导向node4上面的迁移虚机上。第二步是对vm侧的ovs上有关于迁移虚机的表项进行跟新，在图三所示的场景中只需要对node1上的表项信息进行更新，因为node2上的vm采用的是网关节点/分级流表转发的方式，所以node2上的表项信息不需要更新。在更新完成之后，将node3上的中继表项删除，从而完成整个表项更新过程。需要指出的是，我们会在给定的时间内，完成所有关于迁移虚机的表项更新，从而避免长期对旧节点的中继转发，影响网络稳定性。

**4.3 多网关节点的动态匹配**

为实现多个网关节点的动态匹配，我们在源节点层和网关节点层之间增加一层桶。桶是一个逻辑概念，增加一层桶的目的是为了避免网关节点新增/故障带来的影响。桶以规则的形式在Node节点中的流表上实现，

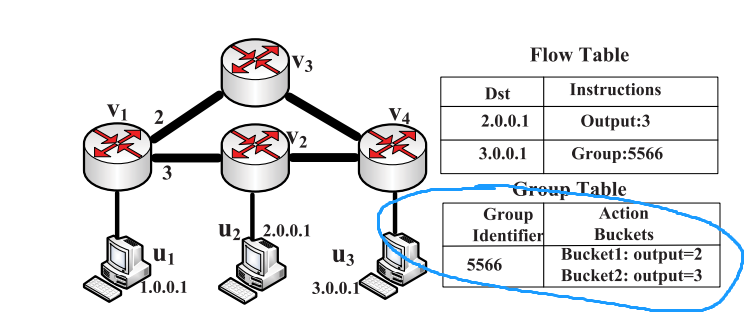


图2、使用规则实现桶

每个桶可分配给单个网关节点，每个网关节点可能被分配给多个桶。每个Node上的桶的数量是固定的。假设当前VPC网络中有N个网关节点，我们对每个计算节点分配的桶数量要远大于N，目的是桶数量能满足网络增加网关节点。我们为每个计算节点分配M个桶（M>>N,比如M=100N），在同一个计算节点上的租户共用这M个桶。所有Node的虚拟交换机都知道Node中租户的桶集合和桶到网关节点的映射，映射是通过控制器根据算法机制（如负载均衡）下发。当一个Node上的一个数据包需要转发且在Node节点上的流表没有匹配的规则时，这个数据包需要被发给网关节点。通过租户id计算a = hash（租户id）％M，虚拟交换机将其哈希到计算节点的某个桶中，然后将数据包转发到当前桶a对应的网关节点。由于M在构造上是恒定的，因此网关节点增加或故障不会影响哈希结果：无论活动的业务网关的数量如何，连接始终hash到同一桶。此外，通过租户id进行hash，可以保证同一计算节点上同一租户的vm被hash到一个桶，从而由一个网关节点管理，实现网关节点与vpc的关联。

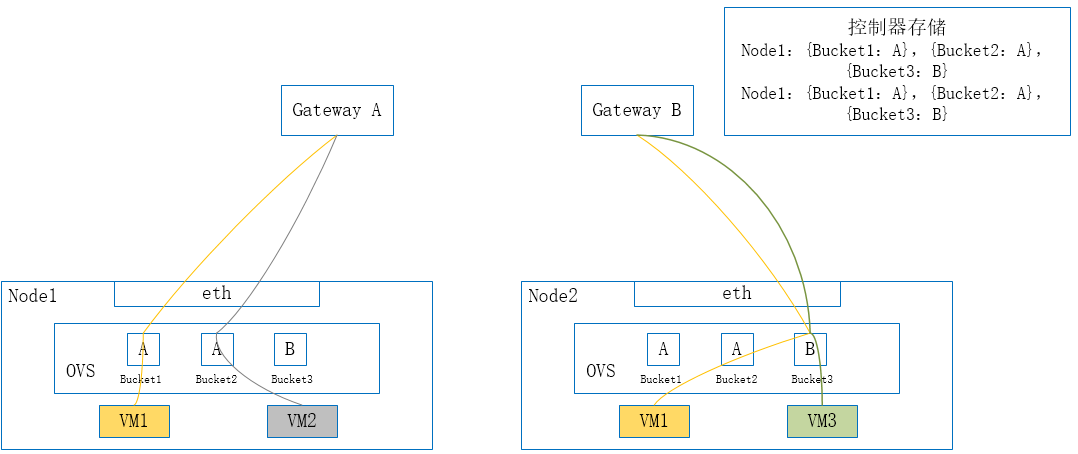


图3、桶的示意图

如图3所示，每个Node里面都有3个桶，这3个桶等价于ovs里的一个组表项，每个Node里的租户共用这些桶。网关节点每隔一段时间上传流量数据（比如，到达本业务网关的流量来自哪个桶），控制器将该数据存储，为以后网关节点扩缩容/单点故障时做准备。控制器收到网关节点上传数据得知网关节点A所有流量来自Node1的Bucket 1/2，控制器将该数据进行存储。

当新增业务网关时，控制器根据收集地信息通过算法选择部分桶的流量（比如，负载过载的业务网关中部分桶的流量）分配给新的业务网关。控制器修改选择的桶的映射关系。当新增业务网关时，只需要修改的控制器选择的部分计算节点即可。

我们通过例子进行详细地说明新增网关节点时的具体操作。假设初始状态如图4所示，虚拟网络平台中有两个网关节点A和B，有三个计算节点Node 1，2和3，还有一个控制器控制Node 1，2和3里的vSwitch。租户1在Node 1和2分别申请了一个虚拟机（VM1），租户2在Node 3申请了一个虚拟机（VM1）。Node 1中桶Bucket1、2和3分别映射Network Service Node A、A和B。Node 2中桶Bucket1、2和3分别映射Network Service Node A、A和B。Node 3中桶Bucket1、2和3分别映射Network Service Node A、A和A。控制器中存储着每个计算节点中桶和网关节点之间的映射表。

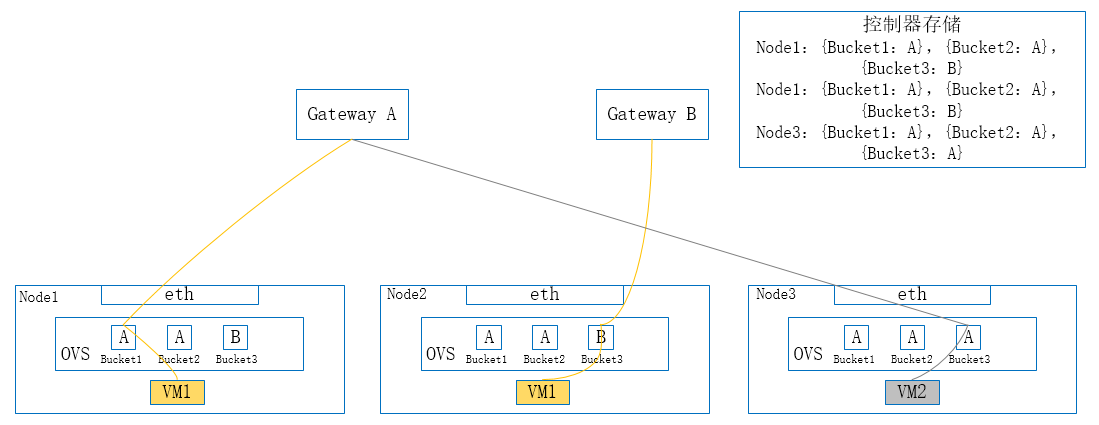


图4、初始状态

如图5所示，当新增一个网关节点C时，首先控制器利用算法选出负载较大的网关节点A，选择调整Node 1上Bucket 1中的流到业务网关C。控制器下发规则，修改Node 1中Bucket 1映射为网关节点C，并更新控制器存储信息。新增网关节点C，只需同步Node 1即可，Node 2，3不受影响。

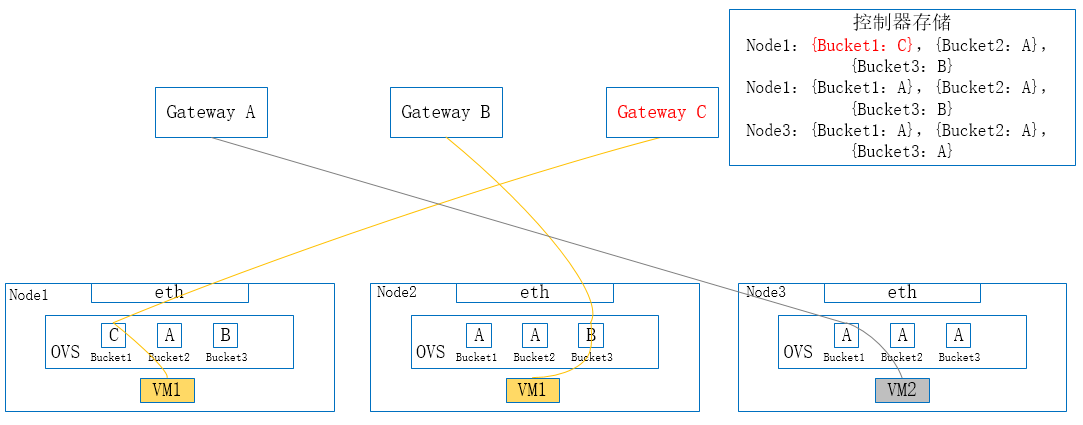


图5、新增网关节点C

当业务网关故障或缩容时，控制器根据收集地信息通过算法选择一些网关节点（比如，负载最低的网关节点），把故障或缩容的网关节点的桶重新映射到选择的网关节点，控制器修改故障或缩容的业务网关的桶的映射关系。所以当业务网关故障或缩容时，需要修改的节点只和故障或缩容的网关节点相对应的桶所属的计算节点有关。因为每个网关节点最多处理p个计算节点的流，控制器需要修改最多p个计算节点，无需修改全部计算节点。

我们通过例子进行详细地说明删除一个业务网关时的具体操作。初始状态如图6所示，虚拟网络平台中有三个网关节点A、B和C，有三个计算节点Node 1，2和3，还有一个控制器控制Node 1，2和3里的vSwitch。租户1在Node 1和2分别申请了一个虚拟机（VM1），租户2在Node 3申请了一个虚拟机（VM1）。Node 1中桶Bucket1、2和3分别映射到网关节点A、A和B。Node 2中桶Bucket1、2和3分别映射网关节点A、A和B。Node 3中桶Bucket1、2和3分别映射网关节点A、A和A。控制器中存储着每个计算节点中桶和网关节点之间的映射表。

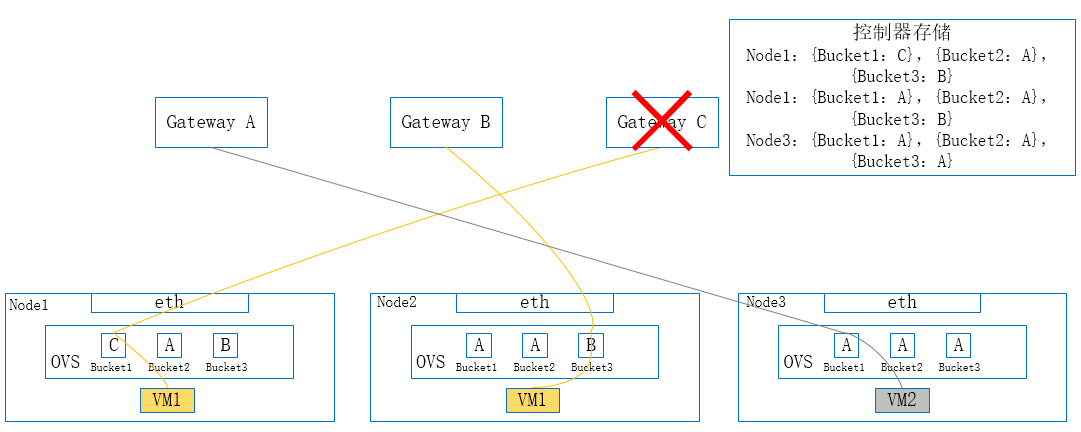


图6、初始状态

如图7所示，当网关节点C失效时，控制器对与网关节点C存在映射的节点Node 1下发规则，修改Node 1中Bucket 1的映射为A。控制器更新存储信息。因为网关节点C一共绑定了1个计算节点，所以删除业务网关C时，只需同步Node 1即可，Node 2和3不受影响。

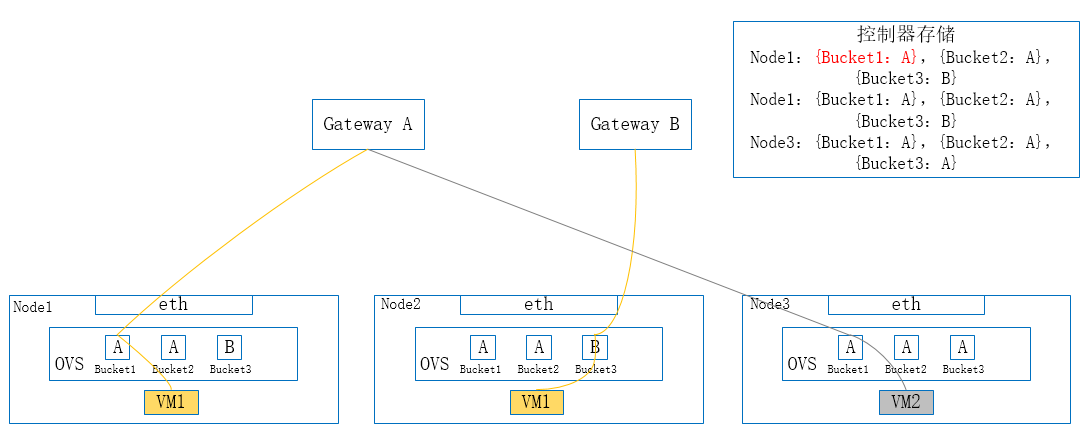


图7、删除业务节点C

注意无论是增加或者删除业务网关，需要同步计算节点是只和故障或缩容的网关节点相对应的桶所属的计算节点有关。

**4.4 流量统计与大流检测**

在分级表项转发模型中，需要增加流量统计和分析的模块来甄别大流。

Netflow技术是一种基于网络流信息的统计技术，可以对网络中的业务流量情况进行统计。一个Netflow系统包括三个主要部分：探测器，采集器，报告系统。探测器是用来监听网络数据的。采集器用来收集探测器传来的数据的。报告系统负责利用从采集器收集到的数据产生易读的报告的。我们拟将Netflow技术适配我们的网关节点模型，使之可以根据流量统计情况自行决定是否下发表项。该方案属于成熟技术。

**4.5 direct path设计**

这里direct path指的是VM之间可以直接通信，而不需要经过网关节点。网关节点上有大小流检测功能，它可以实现定期将大流的表项下发到对应的VM上。若有一个VM需要将一个数据包转发到目的VM，但流表上没有对应的表项，就将数据包转发给网关节点，由网关节点代为转发。简单来说，就是大流可以不经过网关节点转发，而小流由网关节点代为转发。

由于网关节点是否拥有一个VPC的所有表项，即网关节点是否是identical的对direct path设计没有影响，在下文中我们假设一个网关节点拥有一个VPC的全局表项。在SDN网络中，控制器拥有网络的全局信息，可以对交换机下发流表，也能为主机之间建立VXLAN隧道。我们把网关节点做成一个类似于SDN控制器的角色。实现direct path需要经过一下几步：

1. 初始化配置。在一个主机上线时，需要为其下发一条优先级最低的默认表项，其主要作用是发生table miss时将数据包转发给网关节点，由网关节点代为转发。
2. 网关节点挑选出需要下发表项的大流，为对应的endpoint之间建立VXLAN隧道，建立隧道所需要的IP地址等信息都可以在网关节点存储的表项中获取。
3. 配置好VXLAN隧道之后，就可以下发大流的表项。

在上述过程中，网关节点使用OVSDB/RESTCONF/NETCONF协议对ovs进行VXLAN隧道的建立。网关节点使用OpenFlow协议对主机侧的ovs进行流表下发操作。

图一、直接路径模型

例如在上图中，Host1上的VMA需要和Host2上的VMZ进行通信，其通信步骤如下：

1. 初始化配置，在Host1上新建一个默认表项，表示发生table miss时可以将数据包转发给网关节点。

vx0, remote\_ip = 192.168.3.30, type = VXLAN, port=10 //配置该主机到网关结点的隧道

table=1, priority=100,actions=output:10 //下发一条默认流表项，优先级为最低，将不匹配其他表现的流全部通过与网关的隧道端口导向网关结点。

1. 当VMA和VMZ开始通信时，VMA侧的OVS没有到VMZ的表项，此时会发生table miss，VMA通过默认表项将数据包转发给网关节点，由网关节点代为转发。
2. 当VMA与VMZ之间的通信量达到大流的标准的时，网关节点会为VMA与VMZ所在的主机间建立一个VXLN隧道，

Host1的隧道配置:

vx1, remote\_ip = 192.168.2.20, type = VXLAN, port=20

host2的隧道配置:

vx1, remote\_ip = 192.168.1.10, type = VXLAN, port=20

在隧道建立好之后，网关节点将对应的表项洗发至VMX和VMZ所在的主机的OVS上。

Host1的流表：

table=1, tun\_id=200, nw\_dst= 10.0.1.3, action=output:20//直接路径表项，基于目的地址进行匹配

table=1, priority=100, actions=output:10

host2的流表：

table=1, tun\_id=200, nw\_dst= 10.0.1.3, action=output:20//直接路径表项，基于目的地址进行匹配

table=1, priority=100,actions=output:10//默认表项

1. 后续的VMA与VMZ之间通信可以不在经过网关节点，而是直接通信。
2. **模块划分**

如下图所示，我们将表项按需下发方案拆分为两个上层模块：虚机上线模块和虚机迁移模块；三个下层模块：流量统计、流表更新管理和多网关节点匹配。两个上层模块通过调用相应的下层模块实现相应功能。模块的设计思路见下表。

|  |  |
| --- | --- |
| **功能模块** | **功能** |
| 流量统计 | 实时统计流量的交互信息并定期将其发送至管控平面。 |
| 流表更新管理 | 定期根据流量统计模块收集到的流量交互信息，选择通信量较大的流更新至主机侧。 |
| 多网关节点匹配 | 网关节点集群拥有网络全局的流表项，多个网关节点与多个计算节点进行动态匹配，利用基于bucket的ecmp实现该匹配 |
| 虚机上线 | 当新虚机被创建，根据由多网关节点匹配模块在主机侧创建的ecmp表项，可以将新上线的虚机的流量统一转发至某一网关节点，由网关节点通过查询全局表项进行转发操作。此时虚机已经上线成功，可以和其他虚机进行通信。之后流表更新管理模块会根据从交换机收集到的流量将该虚机频繁通信的流的表项下发至虚机侧的交换机。 |
| 虚机迁移 | 当虚机迁移的目标位置的资源已经准备好之后，第一时间并不对全局的路由表项进行更新，而是在原虚机侧OVS上临时部署一条中继表项将转发至该虚机的流全部转发至虚机迁移之后的位置，从而借助原位置实现数据转发。之后进行网关节点和其他计算节点侧交换机与该虚机有关的表项的更新。更新完成后，删除中继流表项。由流表管理模块定期自动将迁移后的虚机频繁通信的流的表项下发至虚机侧的交换机。 |

1. **工作流程**

**5.1 虚机上线工作流程**

虚机上线工作的上层模块是图四中的“虚机上线”，它需要调用底层模块“多网关节点匹配”、“流量统计”和“流表更新管理”来完成虚机上线的工作。下面给出虚机上线的工作流程及每一步涉及到的底层模块。

5.1.1网关节点更新

当有新虚机创建时，在网关节点更新该虚机的表项信息，使网关节点保持正确的全局表项信息。

6.1.2虚机上线配置

同时根据多网关节点匹配模块在主机侧创建的ecmp表项，可以将新上线的虚机的流量统一转发至某一网关节点，，由网关节点通过查询全局表项进行转发操作。此时，虚机上线成功，可以和其他虚机进行通信。

6.1.3虚机流量定期统计

定期（比如每隔1分钟）调用“流量统计”模块，在新上线虚机侧查询其流量接收/发送情况（包含虚机对间发送的数据量），并筛选出通信量较大的虚机对（比如通信量最大的1万个虚机对）。

6.1.4 OVS表项定期更新

定期调用“流表更新管理”模块，将上述通信量大的虚机对的流表项更新至虚机侧的OVS，使频繁通信的流可以直接通过虚机侧的OVS进行转发（一跳即达）。

**6.2 虚机迁移工作流程**

虚机迁移工作的上层模块是图四中的“虚机迁移”，它需要调用底层模块“流表更新管理”、“多网关节点匹配”和“流量统计”这几个底层模块来完成虚机迁移的工作。下面给出虚机迁移的工作流程及每一步涉及到的底层模块。

6.2.1 虚机资源迁移

虚机迁移时需要把原虚机的内存数据，硬盘数据等资源迁移到目的位置，此过程可以使用华为原来成熟的解决方案。

6.2.2中继表项部署

这个过程涉及到“流表更新管理”模块。管控层知道原虚机位置以及目的虚机位置，它在原虚机侧流表部署一条中继规则，该规则将目的地为原虚机的数据包导向新虚机的位置。

6.2.3网关表项更新

将网关节点上关于迁移虚机的规则进行更新，使得目的地为迁移虚机且经过网关节点匹配的数据包能导向新的虚机的位置。

6.2.4 全局表项依次更新

这个过程涉及“流表更新管理”和“流量统计”模块，该过程根据流量统计的结果，依次将网络中的所有关于迁移虚机的表项进行更新，使得不需要再经过原虚机侧的中继规则就可以到达目的虚机。

6.2.5中继表项删除

这个过程涉及“流表更新管理”模块，在全局表项依次更新完成之后，就不在需要中继规则进行中继转发，可以将中继表删除。