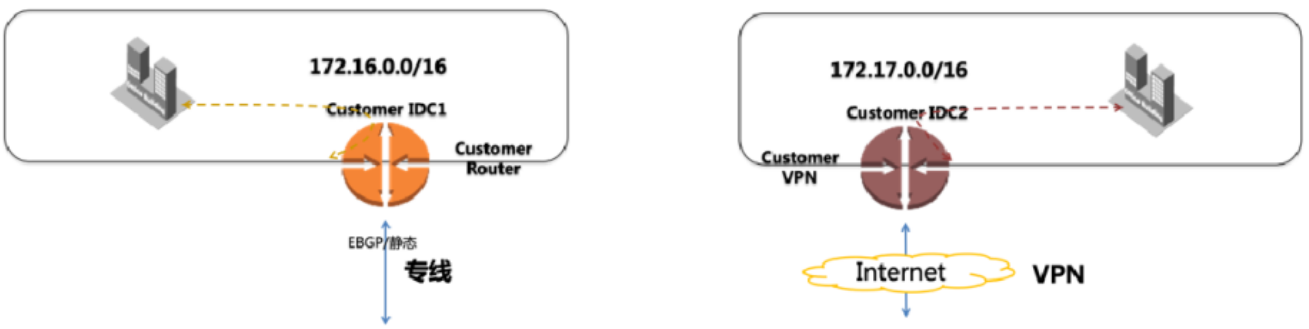
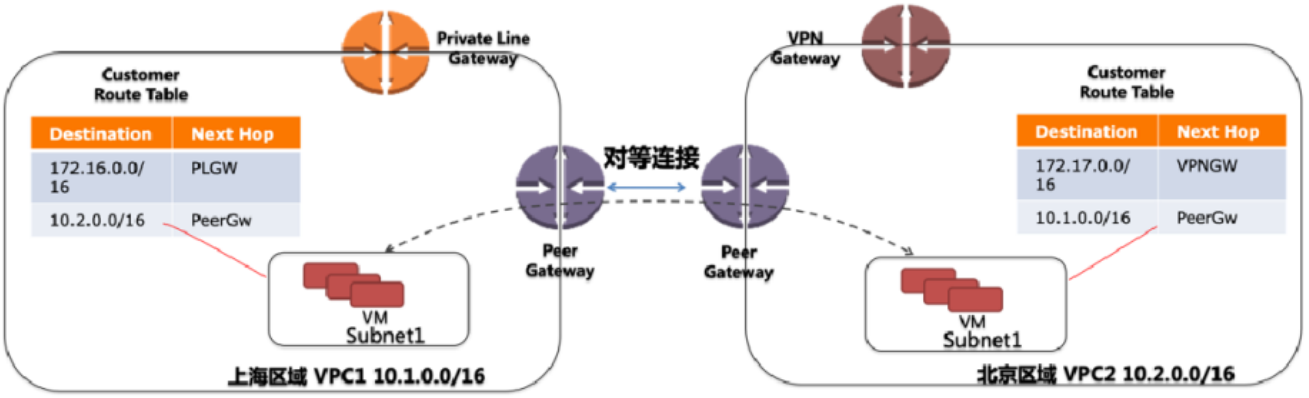
海纳百川，有容乃大 ——云网络SDN控制系统演进之路

唐昌令 鹅厂网事 2020-06-29

VPC提供给客户在云端创建自定义的网络服务，用户可以自定义在云端VPC的子网、IP规划等网络参数，将VPC抽象成用户在云端的数据中心。VPC对等连接方案解决了跨域云端数据中心之间的网络互通，VPC专线方案使用运营商专线将云端VPC和用户自有IDC连接起来，部分用户出于成本或者容灾考虑，采用IPsec/SSL VPN通过internet实现IDC与VPC的互通。





腾讯云提供在全球范围的云网络接入服务，且在计算和网络领域已经迈入“双百时代”（全网服务器总数量突破100万台，带宽峰值突破100T），如何高效、弹性、低成本地实现网络资源的互联互通，是一个充满挑战的课题。随着云网络SDN/NFV技术发展不断演进，控制器在其中扮演了越来越重要的角色。

云网络控制器SDN理念演进

腾讯云网络SDN实践最早在2015年开始，第一个项目Hyena是通过Openflow + 白盒交换机实现防火墙的功能，之后在黑石网络和腾讯云网络提供各类云接入网关服务。随着业务场景的丰富以及业务规模的扩大，为了适应云网络业务的快速变化，云网络系统团队对于SDN的理解也在逐渐的加深，从最初Openflow的尝试，到后面确定以SDN/NFV技术为主导的云网络系统框架，我们在践行SDN的道路上经历了三个重要的阶段。

在践行SDN的道路上经历了二个重要的阶段。

1

SDN 1.0——可编程性、自动化编排

SDN混沌初开，工业界对于SDN的商用案例寥寥无几，各大云网络提供商底层主要还是依赖传统网络设备实现互联互通， 腾讯DCI是构建在运营商物理网络以及传统网络设备上，使用MPLS L3VPN组网技术架构。为了快速上线云接入网关服务，满足客户各类互通需求，SDN 1.0选择了混合SDN模式，云接入网络融入到DCI网络的L3VPN区域，云网络控制器通过北向API支持管理云网络流量，而传统网络协议管理网络上的其他流量。

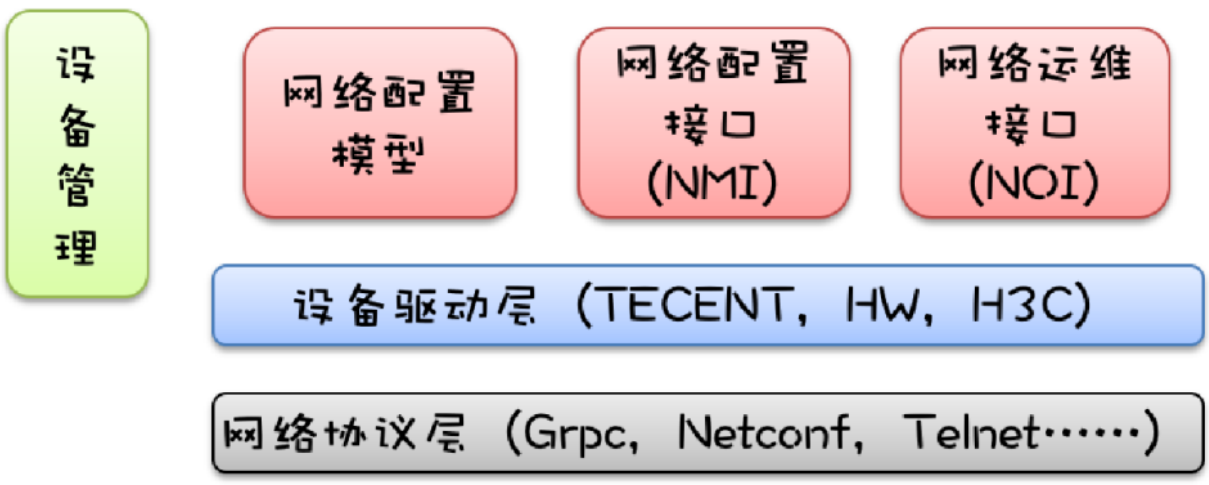
SDN 1.0时代，网络的转发与路由控制更多地还是借助于网络设备自身的能力，对于控制器而言，最重要的是解决如下两点：

- **可编程性与自动化**

通过北向API，对云业务提供多种云网络接入服务，支持灵活调整云端VPC与用户IDC网络连接关系，流量统计与限速，路由策略控制等等，同时通过控制器实现自动化网络编排，加速用户开通网络服务的时效性以及减少网络工程师繁重的网络变更任务和误操作的可能性。

- **多厂商和开放式架构**

在南向适配层，通过提炼通用的网络配置模型，对不同的云网关编排服务提供统一的网络配置接口和网络运维接口，对下屏蔽不同厂商设备配置差异以及网络通信协议差异，新增设备驱动plugin和网络协议plugin支持灵活热加载



2

SDN 2.0——转控分离、集中控制

混合SDN模式在云网络业务发展初期提供了稳定可靠的云接入网络能力，SDN技术在云网络业务场景快速落地铺开，但是随着云接入业务规模的不断扩大，混合SDN模式的弊端也越来

越明显，DCI网络的稳定性和可扩展性受到了极大的冲击。Overlay模式成为取而代之的一种更好的SDN架构，Underlay仍然采用原来的MPLS L3VPN技术，同时在现有基础设施之上通过VXLAN网络隧道运行多个虚拟网络拓扑。

随着SDN/NFV技术的逐步成熟，云网络系统自研高性能转发平面NGW、分布式BGP路由协议平面FCR，同时控制器也在转控分离体系中扮演着不可或缺的角色。控制器一方面负责Overlay虚拟网关间BGP Peer互联互通关系管理以及转发面VXLAN隧道信息同步。另一方面，在L2 VXLAN场景实现了ARP在NGW转发集群内的广播，以支持转发集群ECMP横向扩容，在L3 VXLAN场景通过FCR BGP协议学习动态路由并进行实时路由计算，最终迭代成L3 VXLAN路由指导NGW转发。

SDN 2.0时代，转发、协议和控制解耦，控制器作为SDN体系的拧合器和协调器，除了具备SDN 1.0所需要的功能之外，也扩展了新的内涵：

- **集中控制**

通过全局视角，创建Overlay虚拟网络内部的互联关系、分配BGP Peer互联网段以及同步VXLAN隧道（VTEP，VNI等等）信息，并且在控制器上面实现集中式路由计算，指导Overlay网络点到点的转发。

- **高可靠、高性能、可扩展**

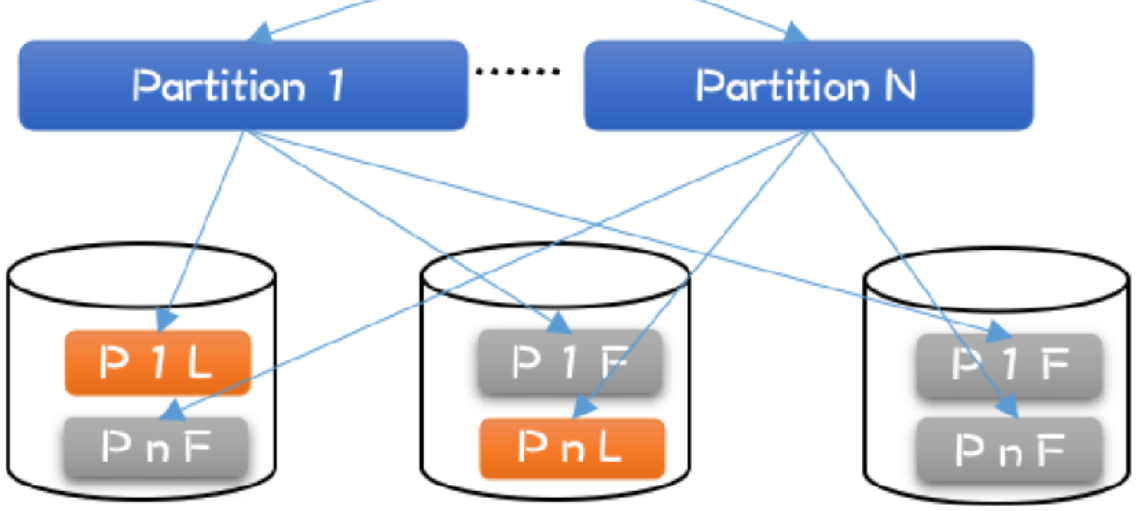
由于实时路由计算以及ARP集群广播等功能的加入，控制器的高可靠、实时性以及横向扩展的需求日益强烈，控制器在分布式集群管理、组件微服务化、高性能后端存储等技术上面做了全方位的技术优化，具体细节在下一章节会重点阐述。

3

SDN 3.0——弹性伸缩、故障切换

实现了转控分离，是否意味着达到了SDN的终极形态？不是的，这仅仅是SDN的开始。SDN从原意来理解，Software Defined Network，既然是软件定义，这其中的想象空间还非常庞大。就像提到SDS（软件定义存储）时，绝不会简单的认为SDS是存储和控制分离。

那么，SDN到底还意味着什么？我们还是以SDS为例，什么是SDS？简单的说，SDS存储系统给用户提供一个逻辑卷（Volume），用户根本不需要关心这个逻辑卷到底映射在哪个服务器或者硬盘上面，用户也不需要担心这个Volume的容量和性能规格问题，因为它是可以支持弹性伸缩的；同样的，如果这个Volume对应的磁盘故障了，用户也完全不感知，因为存储底层有多副本高可靠并且系统会自动将故障盘的逻辑分片迁移到其他正常的磁盘上去。



对于SDN控制系统，对外提供各类Overlay虚拟网关接入服务，迫切需要如下软件定义的能力：

- **支持指定AZ就近接入**

部分客户对接入时延非常敏感，如果VPC内部跨多个AZ，要求虚拟网关（vRouter）对不同的AZ能够自动关联多个网关分片（partition），负责对应AZ下的流量分发，以减少VPC内多个AZ的就近接入网络时延；

- **支持接入带宽弹性伸缩**

直播、游戏的客户在业务高峰时期对于带宽要求非常大，低谷期又需要将带宽快速降低以节省成本；

- **支持网关跨AZ容灾**

当虚拟网关所在的AZ发生灾难性故障时，可以在业务无感知的情况下快速切换到其他AZ的容灾资源上；

- **支持故障迁移**

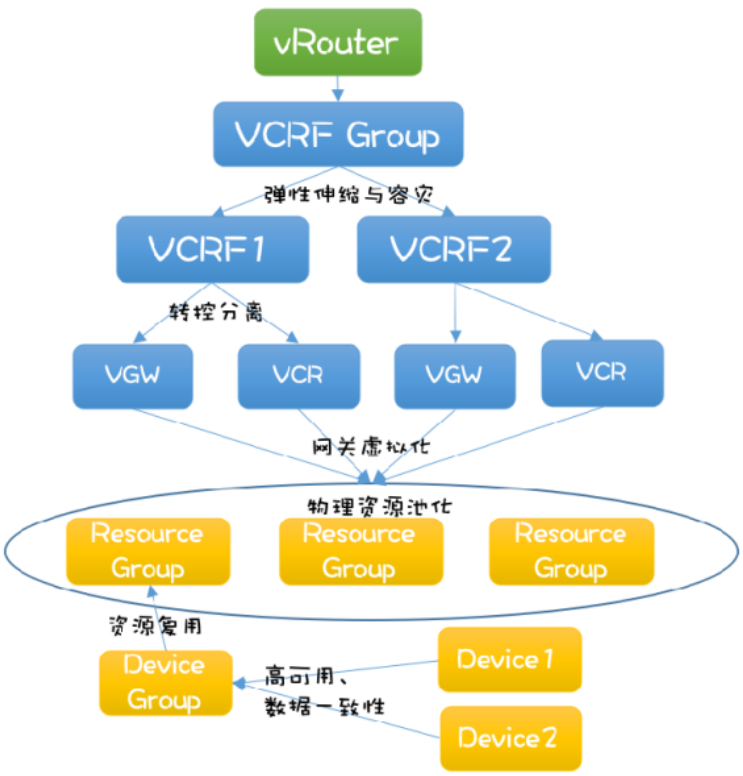
当虚拟网关所在的底层资源集群负载不均或者部分资源集群处于亚健康状态时，系统能自动将优先级高的虚拟网关迁移到正常的资源集群；

通过对上述业务诉求的持续迭代，我们抽象了一套通用的虚拟网络资源管理（VCRF GROUP）模型，借助这套模型框架可以天然支持各类软件定义的需求。VCRF是VRF的SDN化定义，我们知道VRF全称是Virtual Routing Forwarding，表示一个逻辑隔离的路由转发空间。VCRF全称是Virtual Cloud Routing Forwarding，它与VRF最大的不同在于VCRF内部进一步转控分离，包含了一个vGW和vCR。vGW（Virtual Gateway）表示转发平面对象，vCR（Virtual Cloud Routing）表示路由控制对象，转发控制通过逻辑对象关联而不是物理设备关联，实现了转发和控制平面支持独立横向扩展的能力。

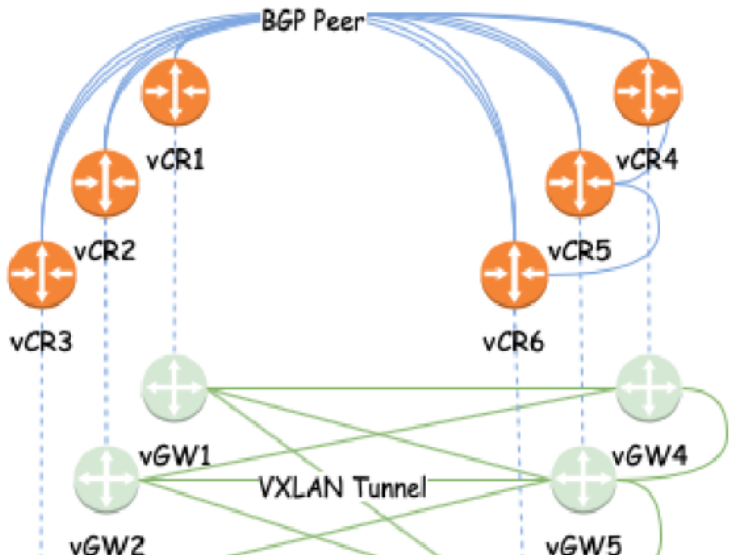
VCRF GROUP就是把多个VCRF进一步组合起来，VCRF在GROUP中支持不同的角色。如果

全部是active则对外体现一个更大的ECMP逻辑网关集群，当业务网关vRouter需要带宽资源弹性伸缩时，对于控制系统仅仅只是在VCRF GROUP模型里面增删VCRF而已，进一步的，VCRF可以携带AZ信息，这样就能更好地实现就近接入特性。如果GROUP中同时包含active和standby的VCRF，vRouter对外就具备了跨AZ的容灾的能力。

在底层网络建设上，按照资源组（ResourceGroup）的粒度进行转发和协议网关集群池化，vGW和vCR分别是转发和路由协议资源组的虚拟化网关。每个ResourceGroup底层都是由设备组（DeviceGroup）构成，DeviceGroup内部由多个设备通过ECMP或者HA组网构建，从而保证底层资源组的高可用和性能扩展，这点类似SDS系统的多副本概念，DeviceGroup业务配置和转发路由的一致性由控制器来保证。



同时，控制器通过全局视角建立Overlay网络控制面和转发面的统一链路管理模型，底下的VXLAN隧道就像一条条奔涌的江河，上面的BGP Peer链路就像一道道大坝的闸门，配合Overlay网络的链路状态探测和流量统计，能快速关联到每条转发和控制链路的负载情况以及健康状态，通过控制器智能运营分析实现自动化流量迁移和故障切换。底层虚拟网关分片的流量迁移和故障切换对于用户侧完全透明，并且时刻保持Overlay接入网关的稳定性能和高可靠。



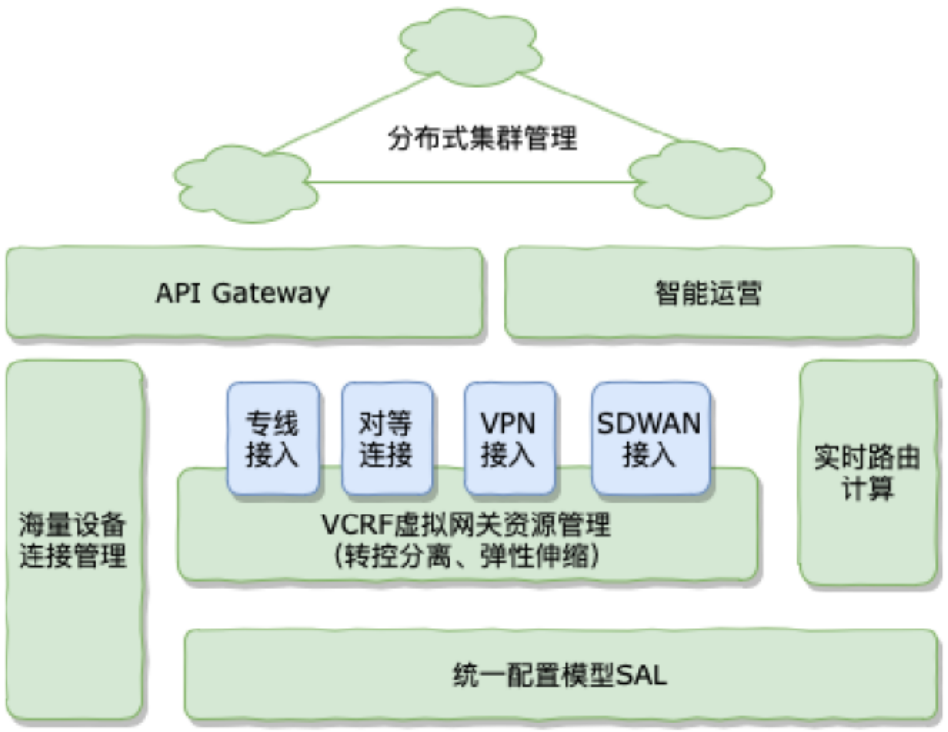
云网络控制器软件架构演进

云网络控制器软件平台经历了从开源ODL（OpenDaylight）控制器平台往基于微服务构建的自研控制器平台迁移的过程。SDN发展之初，ODL平台的出现给我们提供了一个非常不错的基础框架：南向支持丰富的网络协议插件以及厂商设备驱动plugin，强一致的内存数据库Datastore带来不错的性能体验，内嵌式存储模块简化控制器集群的建设和运维，基于模型驱动的服务抽象层（MD-SAL）精简了运维相关的业务接口开发，等等一系列功能为控制器商业化提供了保障。这一阶段的云接入网关服务均是基于ODL平台独立开发，业务得以快速上线并且保持稳定运营。

随着业务场景的不断增加，各个接入服务垂直开发，代码复用性不足，并且大部分基础功能都是相似的。另一方面，随着业务规模的指数级膨胀，ODL原有的框架逐渐无法支撑并且由于内嵌存储很难实现横向扩展，业务对账等场景下大数据量的读写性能变差，甚至进一步影响业务系统本身的稳定性。

统一平台、组件微服务化

通过对云网络接入服务的功能分析，我们抽象出一套统一的控制器平台框架，其中包含几个核心的基础组件，比如海量设备管理，南向配置服务抽象层，实时路由计算，VCRF GROUP虚拟网络资源管理框架等等。而各类云接入网关服务只需要在这个统一平台框架上面各自实现一套服务编排的逻辑处理，从而简化了业务需求变化对软件迭代开发的影响范围。



为了减少业务频繁变更对基础核心组件的影响，我们对这些关键组件服务正在做微服务化改造，业务网关的升级不再影响基础组件的运行，从而确保南向设备连接以及实时路由计算不会发生中断或重试。

在微服务化改造过程中，ODL框架已经不再适合轻量级的服务拆分，我们选择Springboot来构建基础服务组件，并且将这些微服务组件的管理接入到腾讯统一的微服务管理框架。

2

性能、扩展性、可靠性优化

这几年控制器团队持续投入对ODL平台代码的性能优化和bug fix，包括数据面读写通道分离，指定Shard Leader选举实现local read，RPC消息异步处理框架等等。但是ODL平台无法支持横向扩展的致命缺陷（ODL集群设计之初主要是解决可靠性的问题）让我们下定决心做更为彻底的架构改造。

- **业务与存储分离**

存储组件不再内嵌到业务进程中，一方面避免业务进程升级对存储稳定性的影响，另一方面业务和存储可以独立的横向扩容。因此，我们舍弃了原来的Datastore内存数据库，采用后置第三方分布式存储软件。

- **存储读写分离**

为了避免业务对账、查询等等运维功能对控制系统的IO资源抢占，我们选择了高速缓存 + 持久化数据库的架构。在存储中间件选型上面，重点考虑极致性能和支持分布式横向扩展，我们最终选择了Redis集群 + MongoDB数据库集群的方案。

- **组件间消息队列通信**

控制器组件间数据通信，为了保证消息可靠性以及平滑各组件处理性能不均等的问题，我们采用Kafka消息队列对组件间的消息通信进行性能的削峰填谷，并且当其他组件故障时，数据消息也不会被丢弃，从而保证业务处理的最终一致性。

展望未来

回望前时路，云网络SDN控制器的发展方向也越来越清晰，纵然未来业务还会不断地变化调整，但是我们心有丘壑，坚定向前。

- **纯软化**

提供统一软件定义网络框架，实现业务需求的变化不需要修改设备基础配置或者变更底层网

络架构。

- **海量化**

转发平面是浩浩荡荡的百川，控制平面就是飘在天空的云海，吸纳了全网的路由，通过实时计算最终又滋润到每条江河，指导江河的流量转发。海量的路由规格，实时的计算性能和传递，决定了控制器未来的高度。

- **智能化**

控制器是全网流量的控制大脑，拥有了Overlay网络的全局链路视图，通过后续更全面更实时的网络指标监控和智能分析，并且将这一切可视化地呈现给用户。

鹅厂网事

分享鹅厂网络的那些事

扫码关注！解锁更多~