# B4: Experience with a Globally-Deployed Software Defined WAN

## 背景介绍

随着广域网上链路的日益复杂，协议、数据等规模的不断扩大，想要管理和提高网络利用率变得越发的困难，现在网络运营商在没有找到有效的解决方法之前，只能通过不断扩大链路带宽，增加铺设的链路数量，才能屏蔽掉网络故障所带来的影响。然而Google拥有两种广域网类型，一种是传统的面向用户的，给用户提供基础应用的网络，另一种就是Google的数据中心。Google计划从改造数据中心入手，因为其网络设施相对简单，设备类型、数量相对较少，而链路的带宽需求却是巨大的，所以对该网络的改进，首先改造方便，另外，带来的经济效益巨大。

为此，Google计划采用基于SDN的架构，来重构数据中心网络，以往的网络采用静态hash实现负载均衡，但难以做到真正的负载均衡，且无法针对不同的应用区别对待，因此急于增加网络利用率、简化管理，所采用的手段大致是流量工程和软件管理。

## 改造过程

第一阶段在2010年春天完成，把OpenFlow交换机引入到网络里面，但这时OpenFlow交换机对同网络中的其他非OpenFlow设备表现得就像是传统交换机一样，只是网络协议都是在Controller上完成的，外部行为来看表现得仍然像传统网络。

第二阶段是到 2011年中完成，这个阶段引入更多流量到OpenFlow网络中，并且开始引入SDN管理，让网络开始向SDN网络演变。

第三个阶段在2012年初完 成，整个B4网络完全切换到了OpenFlow网络，引入了流量工程，完全靠OpenFlow来规划流量路径，对网络流量进行极大的优化。

## 设计架构

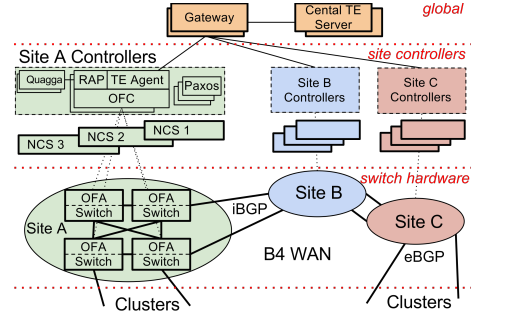


图1 SDN网络架构

第一层为物理交换机，这与通常的网络设备没有本质区别，只是其中转发表的规则制定不是协议间自行计算了，而是由上层控制器提供，但不能无限制的接受所有下发的字段，需要基于一定规则，这称之为NDM（可协商的数据转发面模型），即上层所下发的流表仍然要基于现有的芯片架构模型。

第二层为controller，这里Google并没有采用单独的controller作为本site的单独控制器，而是设置了多个控制器，由Paxos程序选举主控制器，这种方式极大得提升了SDN网络的容错性。

第三层是全局控制器，GateWay收集下方传来的链路信息，传达到TE server中，TE根据这些信息来规划路径信息，当有一个新的业务要在数据中心网络上传输时，TE也会综合考虑该业务的优先级、带宽需求等信息，为其分配一条最佳路由。

## 按需定制的网络路由

由于传统的网络都是无状态，尽力交付的模型，因此无法适应日益复杂的应用程序，有的应用带宽需求量小，但延迟要求高，而有的网络带宽需求大，且容忍部分丢包与重传。而SDN网络恰好可以根据应用程序自身的需求来设置相应的转发规则，进一步，如果要针对每种单独的应用都制定转发规则的话，那么转发表的大小将会很大，这在进行匹配查找的过程中会花费大量时间。

而Google认为，不一定要将每一种应用单独进行处理，可以将它们按延迟要求、带宽大小、是否可靠等进行归类，具体的，Google数据中心拥有三种主要应用服务，分别是：

1. 面向用户的远程存储、访问。
2. 计算资源、数据资源的调度，以进行云计算。
3. 数据中心之间的数据状态同步。

Google将它们按照传输量大小、延迟敏感度、优先级进行排序。形成类似于{Source site, Dest site, QoS}这样的三元组。之后，TE server就会根据传来的元组信息，结合链路状态为其分配当下的路由。

我认为这是一种值得推崇的做法，事实上也应当这么做，以前说软件定义网络，现在还应该再加上网络适应应用。我认为这三者是密不可分的。

## 逐步改进的SDN网络架构

Google并不是直接将数据中心改造为SDN架构的，而是在其之上不断扩充SDN的关键模块来逐步实现软件定义网络这一规划的。具体的，Google先是将控制器融入到传统网络中，但并不完全取代接管控制平面，而是采取控制和原有路由相结合的方式，交换机既可以从控制器中获得转发规则，也能由原来的路由协议计算转发规则。从它的改进过程就可以得知一二。另外从控制器和交换机的连接关系中也能得出这个结论。

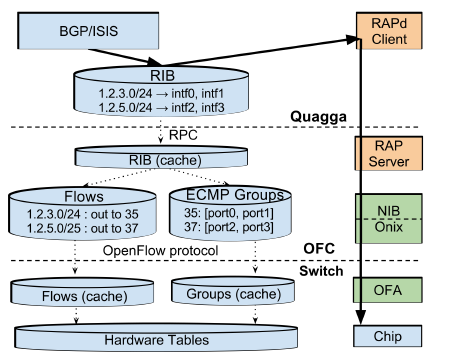


图2 控制器和交换机的连接关系

上图是控制器和交换机的连接关系，OFA收集链路信息，将信息整合通过RAP整合至控制器中，计算出来的网络拓扑和路由规则都保存在RIB中，而与它平级的是Quagga协议栈，这是Google采用的开源BGP/IS-IS协议，用于路由计算。因此可以看出，交换机中转发表的来源实际上有两个，一个是控制器传来的，另一个是传统路由协议算法计算出来的。这样做的好处也提升了B4网络的容错性。

## 流量工程

架构设计好了，总得实现吧。那么这一部分其实就是如何根据链路信息，并结合应用特点来计算路由的算法实现了。

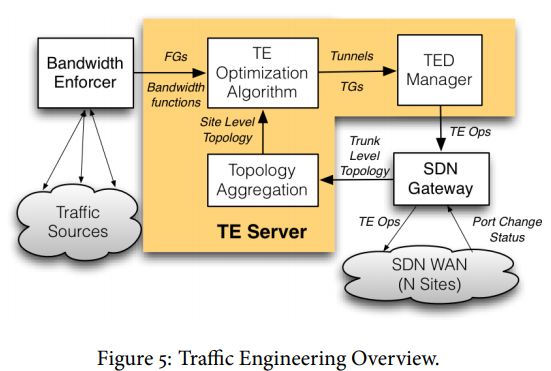


图3 流量工程总览

如图3，TE server的输入，一个是FGs，即由应用的{Source site, Dest site, QoS，bandwidth}组成的四元组，另一个是网络拓扑信息，是由控制器传来的。TE算法的目标就是根据现有网路拓扑信息，结合应用的优先级为其分配具体的连路隧道，并为之分配相适应的带宽。具体的分配步骤如下：

1、选定初始的隧道。

2、为每个隧道按照优先级分配带宽，要么平等对待，要么完全满足。

3、如果每一个隧道都被分配完，则结束，否则进入4。

4、剥离超过平均满足水平的链路带宽，加入待剩余分配带宽，进入2。

从逻辑上为之分配路由之后，还要从具体的链路上也要做好分配。这一部分无法达到最优化求解，因为硬件级的流量控制无法做到绝对精细划分，因此Google通过贪心算法来为不同路由分配具体链路。具体做法如下：

1、将隧道进行分裂，并对每个分得的流量向下取整。

2、对每一个单独的分裂的小的隧道，作为因子，分配给相应流。

3、如果分配完毕则结束，否则进入2。

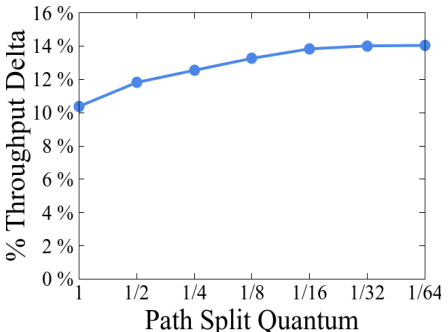


图4 隧道离散化程度及其对应吞吐量大小

离散化会降低性能，由于不能无限分割，总存在浪费带宽资源。分配精细程度越高，利用率越高，但相应的控制成本增加，根据图4所示的二者关系图，google最终选定分配粒度大小为1/4。

## 部署效果和展望

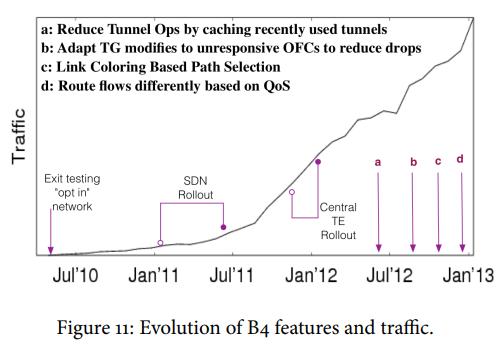


图5 B4网络的流量增长示意图

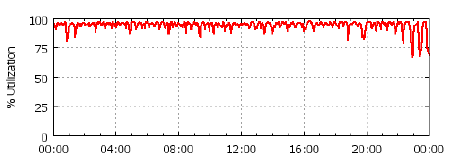


图6 B4网络一天的链路利用率

从Google开始逐步加入控制器，到后来完全部署SDN网络，可以看到随着时间的推移，该网络所能够支撑的流量越发的庞大，这至少证明了B4架构的成功性。另外从图6也可以看出B4架构的优势，从原来链路利用率只有30%到至今的100%。

当然，B4并不完全就是最优的，这篇论文作者本人也提到，由于采用传统的交换机，导致上层应用制定的转发规则到下层交换机实际运用仍存在一定的转换开销。另外，TE计算结果的变更也可能会导致链路拓扑的变化，尽管已经通过链路聚合来尽可能减小这种震荡，但一旦发生路由变动，开销将是巨大的，尤其是当链路上数据正在传输时，需要通过顺序依赖来确保数据不会丢失。

## B4的启示

1、第一个公开的成功的SDN案例，让人了解到分布式的controller应当如何工作及工作效果。

2、QoS四元组的制定推进了流量工程，将应用特性作为制定转发路径时考虑的因素是一个不错的尝试。

3、演示了如何将SDN和传统路由算法相结合，为从传统网络过渡到SDN提供了经验。