中国科学: 信息科学 2016 年 第 46 卷 第 5 期: 665–676
SCIENTIA SINICA Informationis

《中国科学》杂志社
SCIENCE CHINA PRESS

论 文

利用基于 SDN 的 IP 融合切换技术实现面向多业务的动态流量调控

马守江, 胡道允, 李晟如, 薛娜娜, 邵妍, 李索恒, 朱祖勍*

中国科学技术大学信息科学技术学院, 合肥 230027
* 通信作者. E-mail: zqzhu@ieee.org

收稿日期: 2015–07–24; 接受日期: 2015–09–06
中国科学院 “感知中国” 先导专项子课题 (批准号: XDA06010303)、国家自然科学基金 (批准号: 61371117) 和电子科技大学光纤传感与通信教育部重点实验室开放课题资助项目

摘要 本文基于软件定义网络 (software defined betworking, SDN) 概念, 利用 OpenFlow 技术设计并实现了一种新颖的 IP 融合切换技术. 该技术基于当前网络状况对 IP 数据流的下一跳转发方式进行灵活选择, 在 IPv4 与 IPv6 域之间完成高效的动态切换. 因此, 它可以在 IPv4 与 IPv6 共存的异构网络中实现服务质量 (QoS) 感知的动态流量调控. 本文对 OpenFlow 进行了适当的拓展, 并完成了 OpenFlow 系统设计以实现 IP 融合切换技术, 同时提出了面向多业务的动态流量调控算法, 并利用真实的网络交换机和服务器搭建了实验平台, 验证了系统和算法的性能. 视频和文件同步传输实验的结果证明了所提出的 IP 融合切换技术在实现动态流量调控方面的优越性.

关键词 SDN OpenFlow 流量工程 IP 融合切换 QoS

1 引言

在过去的几十年里, 传统的 IP 网络改变了人们的生活方式, 极大地促进了经济和社会的发展, 人们几乎可以在任何地方方便地接入到互联网. 然而, 因为传统 IP 网络接入因特网都需要 IP 地址, 随着互联网的快速发展以及网络设备和新型网络应用的增加, IPv4 地址资源已经即将耗尽. 互联网号码分配局 (IANA) [1)] 在 2011 年 2 月宣布全球最后一段 IPv4 地址段分配完成, 这也意味着在不久的将来, 网络设备通过 IPv4 地址接入到互联网将变得越来越难. 因此, 为了解决这一问题, IPv6 作为下一代互联网的希望被提出和逐渐部署开来 [1].

但是由于经济和技术上的困难, 想要在现有网络中同步部署并更新 IPv6 地址几乎是不可能的. 因此, IPv4 与 IPv6 将长期共存于互联网中 [2], IPv4 到 IPv6 的转换需要循序渐进的进行. 为了实现 IP 平滑演进, 需要一种能够支持 IPv4 和 IPv6 不同网络或设备间相互通信的技术, 其中互联网工程任务组 (IETF) [2)] 制定和规范了许多机制以实现 IPv4 和 IPv6 的相互通信 [3]. 值得注意的是, 在 IPv4 到

1) Internet Assigned Numbers Authority (IANA). https://www.iana.org/.
2) Internet Engineering Task Force (IETF). http://www.ietf.org/.

引用格式: 马守江, 胡道允, 李晟如, 等. 利用基于 SDN 的 IP 融合切换技术实现面向多业务的动态流量调控. 中国科学: 信息科学, 2016, 46: 665–676, doi: 10.1360/N112014-00315

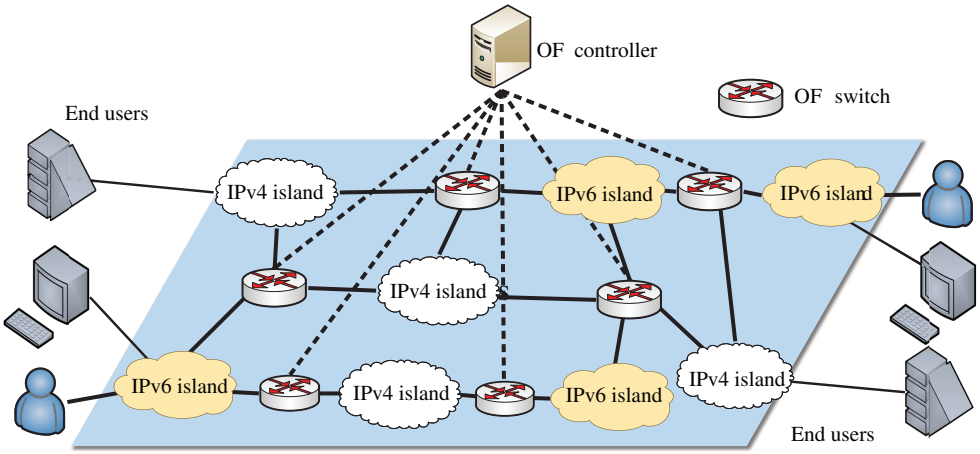


图 1 (网络版彩图) 具有 QoS 感知的动态流量调控网络系统架构

Figure 1 (Color online) Network architecture for QoS-aware flexible traffic engineering

IPv6 平滑演进过程中, IPv4 "孤岛" 和 IPv6 "孤岛" 问题逐渐暴露出来, 这为实现网络流量调控带来了新的挑战. 例如, 传统的 IPv4 业务只能在 IPv4 域内或相邻 IPv4 域间进行转发, 然而如果能够在相邻空闲的 IPv6 域中转发的话, 传统的 IPv4 业务可以获得更好的服务质量 (QoS) 保证 [4].

最近, 软件定义网络 (SDN) 的概念被提出, 它通过实现网络中控制平面和数据平面的解耦 [5], 用软件定义的方式对网络进行监控和管理, 这使得网络具有灵活的可编程性. OpenFlow 的提出 [6] 使 SDN 的概念得以实现, OpenFlow 系统中使用 OpenFlow 协议作为控制平面和数据平面的通信协议, 采用基于流的流表匹配转发策略实现对不同数据流的转发. 此外, OpenFlow 系统的经济性、灵活性和可编程性也使得部署新的路由算法和流量调控更加快捷. 比如, 文献 [7] 演示了一种利用 SDN 的经济有效地部署 IPv6 的方案; 文献 [8] 讨论了利用 SDN 实现 IPv4 和 IPv6 之间数据流的互通问题. 由于以上 SDN 的特性, 我们可以通过集中的控制平面获取到全局网络状况并实现 QoS 感知, 所以利用 SDN/OpenFlow 来实现面向多业务的动态流量调控可以提高网络链路利用率, 为多业务提供更高效的 QoS 保证.

本文基于软件定义网络 (software defined networking, SDN) 概念, 在 IPv4 和 IPv6 域共存的异构网络中利用 OpenFlow 系统设计并实现了具有 QoS 感知的动态流量调控策略. 首先, 利用 OpenFlow 系统可获取当前网络状况, 基于网络状况对 IP 数据流的下一跳转发方式进行灵活地选择. 为了实现这种方案, 我们对 OpenFlow 进行了适当地拓展, 以支持 IP 融合切换技术. 最后, 我们通过真实的网络交换机和服务器搭建了实验平台, 提出面向多业务的动态流量调控算法, 利用视频和文件同步传输实验验证了系统和算法性能. 论文的其他部分安排如下, 第 2 节详细描述了动态流量调控网络系统架构和操作过程; 第 3 节具体介绍了实验过程, 并对实验结果进行了详细地分析; 第 4 节是对研究内容的总结.

2 系统架构与操作原则

2.1 网络系统架构

图 1 展示了具有 QoS 感知的动态流量调控网络系统架构. 网络由多个 IPv4 域和 IPv6 域组成,

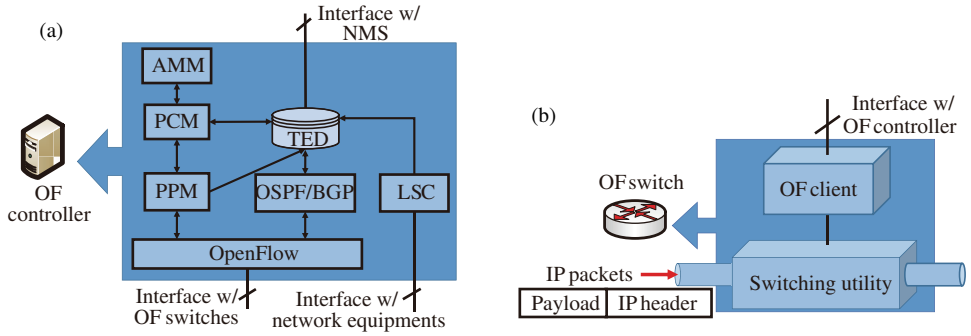


图 2 (网络版彩图) OpenFlow 控制器和交换机功能结构

Figure 2 (Color online) OpenFlow controller and switch function structure. (a) OpenFlow controller; (b) OpenFlow switch

在各个域中, 由 IPv4 或 IPv6 路由器负责转发业务. 在传统网络中, 不同域间利用转发网关, 通过如简单因特网转换机制 (SIT) [9] 实现通信. 而在 OpenFlow 网络中, 不同域间通过 OpenFlow 交换机进行连接, OpenFlow 交换机再由 OpenFlow 控制器进行集中管理和控制. OpenFlow 控制器可以通过多种方式获取网络全局信息, 例如通过与域间路由器建立简单网络管理协议 (SNMP) 的连接, 或者通过数据流传输过程中的丢包率或延时来获取, 之后根据获取的网络全局信息对数据流进行流量调控.

在通过 OpenFlow 系统实现 IP 融合切换过程中, OpenFlow 控制器决定下一跳是 IPv4 或 IPv6 域并下发 IP 切换指令 (通过流表信息), OpenFlow 交换机负责接收并执行相应的 IP 切换指令. IP 切换过程是通过建立 IPv4/IPv6 隧道完成的, OpenFlow 交换机在 IPv4 数据包中插入 IPv6 报文头, 封装成一个 IPv6 数据包; 同样的, OpenFlow 交换机在 IPv6 数据包中插入 IPv4 报文头并封装成一个 IPv4 数据包. OpenFlow 交换机和 OpenFlow 控制器通过拓展的 OpenFlow 协议完成上述操作, 针对不同业务 QoS 需求, 在 OpenFlow 系统中实现动态流量调控算法, 提高链路利用率.

2.2 OpenFlow 控制器

图 2(a) 是 OpenFlow 控制器功能结构框图, 各个功能模块的具体描述如下:

路径提供模块 (link provision module, LPM). 负责与 OpenFlow 交换机和路径计算模块 (LCM) 通信, 用于发送和接收 OpenFlow 消息, 完成建立或切换数据流的转发路径. 例如, 在一条新路径建立过程中, 首先 LPM 会接收到从交换机发来的 Packet-In 消息, 然后转发给 LCM 模块计算路径. 之后将从 LCM 得到路径计算结果并封装成 Flow-Mod 消息, 发送到路径上的所有 OpenFlow 交换机, 完成路径建立. 此外, LPM 还与流量调控数据库 (TED) 进行通信, 在 TED 中存储数据流转发信息.

路径计算模块 (link computation module, LCM). 从 LPM 模块获取路径计算任务, 利用 QoS 感知的动态流量调控算法实现数据流路径的计算和优化. 当 LCM 接收到路径计算任务之后, 首先从 TED 中获取当前网络状态信息, LCM 会针对不同优先级的业务计算不同的路径转发策略, 之后调用 LPM 把路径转发信息封装成流表信息. 此外, 当一条数据流因为网络状态的变化需要切换路径时, LCM 会根据 TED 中的网络状态信息对路径重新优化, 计算出新的转发路径之后调用 LPM 完成新路径的建立.

流量调控数据库 (traffic engineering database, TED). 存储当前网络状态信息, 包括动态接入的 OpenFlow 交换机、IPv4 和 IPv6 域边缘节点信息、每条链路的带宽利用率信息以及不同业务 QoS 需求信息. OSPF/BGP 模块、链路状态收集模块 (LSC) 和 LPM 会实时更新 TED 数据库, 以保证数

据库中信息的实时性和有效性. 此外, 管理人员也可以通过外部网络管控系统 (NMS) 读取 TED 中的信息.

OSPF/BGP 模块. 在域内使用 OSPF 协议时, 每个 OpenFlow 交换机可以像传统的路由器一样接收 OSPF 消息 [10], 学习并广播路由信息; 在域间使用 BGP 协议时, OpenFlow 控制器作为一个单独的 BGP 发言人, 与每个域内 BGP 发言人交换路由信息. 由于所有的 OpenFlow 交换机由一个统一的控制器管理, 这种集中控制的方式使得控制器可以根据收集到的路径信息实现集中式的路由算法.

链路状态收集模块 (link state collection, LSC). 利用简单网络管理协议 (SNMP) 收集实时的链路状态信息 [11, 12]3), 包括链路利用率等.

地址池模块 (address provision module, APM). 管理 IPv4/IPv6 地址, 用于实现 IPv4/IPv6 隧道建立.

2.3 OpenFlow 交换机

OpenFlow 交换机的内部结构如图 2(b) 所示. OpenFlow Client 模块与 OpenFlow 控制器通过拓展的 OpenFlow 协议进行通信, 解析和配置从控制器收到的流表信息. 对数据包的操作在 Switching Utility 模块完成, 包括下面将要提到的本工作中拓展的 OpenFlow 行为.

我们使用 OpenvSwitch 4) 作为 OpenFlow 交换机, 在此基础上拓展了 OpenFlow 协议并设计了 5 个新的流表匹配行为, 可实现动态流量调控的 IP 融合切换功能. 5 个新的流表匹配行为如下:

PUSH_IPv6: 在 IPv4 数据包之上添加 IPv6 报文头, 然后封装成 IPv6 数据包, 实现 IPv6 隧道;

POP_IPv6: 将 IPv6 报文头从数据包中移除, 实现 IPv6 数据包的解封装过程;

MOD_IPv6: 修改 IPv6 数据包的源和目的地址;

PUSH_IPv4: 在 IPv6 数据包之上添加 IPv4 报文头, 然后封装成 IPv4 数据包, 实现 IPv4 隧道;

POP_IPv4: 将 IPv4 报文头从数据包中移除, 实现 IPv4 数据包的解封装过程.

例如, 当控制器通过利用动态流量调控算法对一个 IPv4 的业务流进行路径计算时, 计算路径中包含 IPv4 域到 IPv6 域的切换路径, 那么控制器会将计算结果以流表的形式发送到各个节点上. 在进入 IPv6 域的转换节点上对 IPv4 数据包进行封装操作, 利用拓展的 PUSH_IPv6 和 MOD_IPv6 操作将 IPv4 数据包封装成 IPv6 数据包, 之后交给 IPv6 网络进行传输. 在此封装的 IPv6 数据包经过 IPv6 边缘节点进入 IPv4 域时, 利用拓展的 POP_IPv6 操作将 IPv6 报文头从数据包中移除, 实现解封装过程. 之后交给 IPv4 域进行传输. 同样的, 对于一个 IPv6 业务流来说, 如果计算路径包含 IPv6 域到 IPv4 域的切换路径, 可以利用 PUSH_IPv4, POP_IPv4 和 MOD_NW_SRC/MOD_NW_DST 几种行为进行封装和解封装过程, 其中 MOD_NW_SRC 和 MOD_NW_DST 是现有 OpenFlow 协议已经支持的流表行为, 分别表示修改 IPv4 数据包的源和目的地址.

2.4 基于 OpenFlow 的 IP 融合切换流程

下面通过一个实例具体说明借助于 OpenFlow 系统实现 IP 融合切换的流程, 实例为 IPv4 数据流通过 IPv6 域传输的情形, 具体步骤如流程图 3 所示.

另外, 如果 LCM 判断到流的下一跳是 IPv4 域, 那么入口交换机将直接转发数据包到相应的输出端口上, 而不会做 IP 切换. 注意基于 OpenFlow 的 IP 融合切换技术与传统的 IPv4/IPv6 转发网关的

3) How to calculate bandwidth utilization using SNMP. http://www.cisco.com/c/en/us/support/docs/ip/simple-network-management-protocol-snmp/8141-calculate-bandwidth-snmp.html.

4) OpenvSwitch. http://openvswitch.org/.

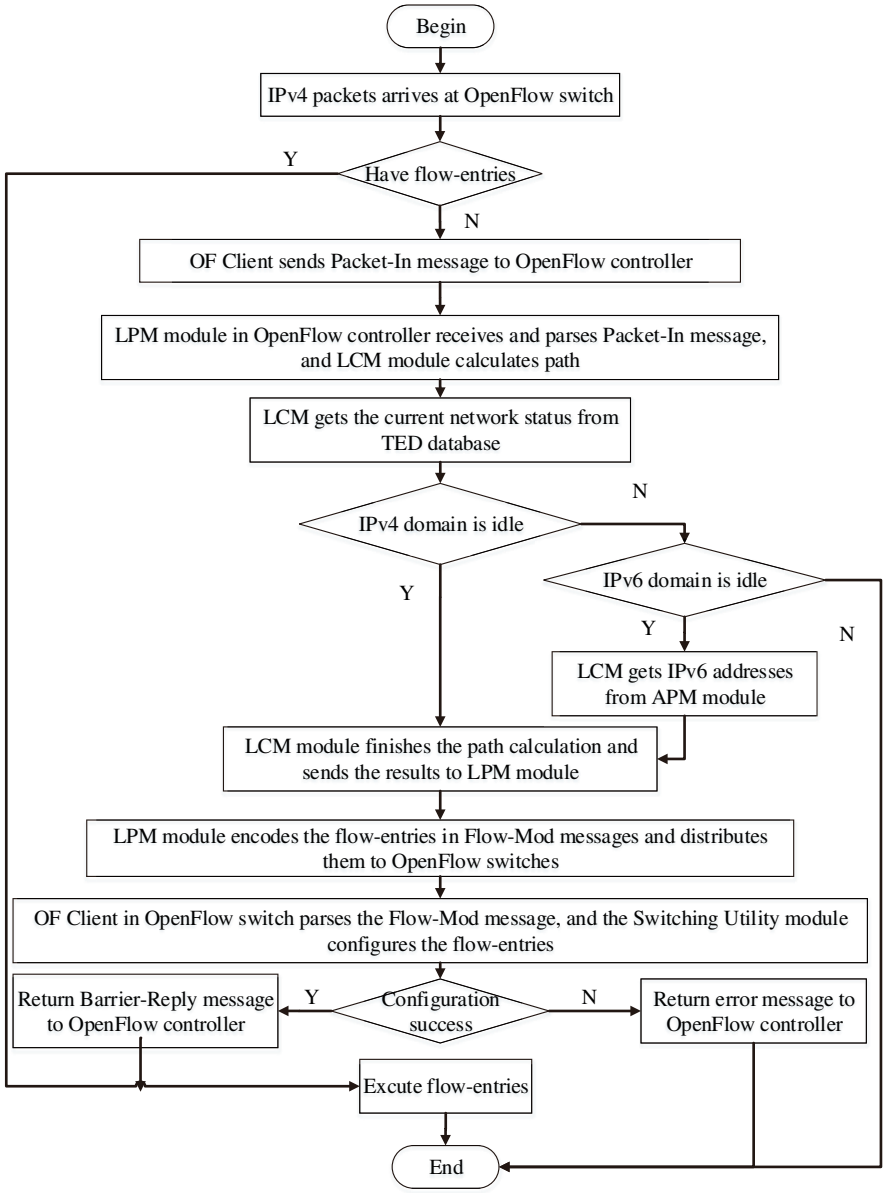


图 3 基于 OpenFlow 的 IP 融合切换流程

Figure 3 IP-forwarding interchanging process based on OpenFlow

区别是基于 OpenFlow 的 IP 融合切换技术可以针对每条流实现动态的 IP 切换, 而这种切换对于其他数据流的转发规则是不会有任何影响的, 其他数据流仍能按照自己的转发规则在转发节点上传输.

2.5 动态流量调控算法

为了实现具有 QoS 感知的 IP 融合切换策略, 我们设计了动态流量调控算法, 如算法 1 所示. 算法 1 详细说明了实现动态流量调控的流程, 算法由 OpenFlow 控制器的 LCM 模块完成. 首先, 我们通过 OSPF/BGP 模块以及 OpenFlow 协议获取网络信息 $G(V, E)$, 这里的 V 和 E 分别表示节点和链路

信息; 链路利用率 u_e 由控制器的 LSC 模块收集; s-d 表示拓扑 $G(V,E)$ 中的节点对; $f(s,d)$ 表示 s-d 的数据流; 数据流 $f(s,d)$ 的优先级 $\mathrm{pr}_{f(s,d)}$ 可以有多种获取方式, 例如根据业务流本身端口号等信息或者由网络管理者人为决定. 算法 1 的第 3∼9 行是获取当前链路利用率最小的路径, 第 10∼27 行判断当前是否所有路径都已拥塞, 如果拥塞, 那么将判断与传输优先级最小的业务交换路径之后是否仍会拥塞, 若仍然拥塞则放弃服务, 否则调用 LPM 建立相关路径, 更新 TED 数据库. 如果当前计算路径不拥塞, 第 28∼31 行直接调用 LPM 建立新路径并更新 TED 数据库. 注意到由于算法是基于业务优先级及网络链路状态的, 所以使得网络流量调控具有 QoS 感知能力.

算法 1 Flexible traffic engineering (F-TE) algorithm with IP-forwarding interchanging

Input: Network Topology $G(V,E)$, the current link utilization $\{u_e,\ \forall e\in E\}$, the K-th shortest path $\{p_{s,d}^{(k)}, k\in[1,K]\}$ got from KSP algorithm for the pair of source and destination node; When one new flow comes, the link utilization of the i-th path is recalculated to be u_i'; The minimum priority for all flows in link k is $\mathrm{pr}_{\min}(k), k\in[1,K]$, the priority of flow $f(s,d)$ is $\mathrm{pr}_{f(s,d)}$.

1: // New flow arrives:
2: **if** new flow $f(s,d)$ arrives **then**
3: $i=0$, $m_{\min}=+\infty$;
4: **for** $k\in[1,K]$ **do**
5: $m=\max\limits_{e\in p_{s,d}^{(k)}}(u_e)$;
6: **if** $m<m_{\min}$ **then**
7: $i=k$, $m_{\min}=m$;
8: **end if**
9: **end for**
10: calculate u_i';
11: **if** $u_i'>100\%$ **then**
12: $\mathrm{pr}_{\min}=+\infty$, $\mathrm{id}=i$;
13: // find the path for the flow with the minimum priority:
14: **for** $k\in[1,K]$ **do**
15: **if** $\mathrm{pr}_{f(s,d)}>\mathrm{pr}_{\min}(k)$ and $\mathrm{pr}_{\min}>\mathrm{pr}_{\min}(k)$ **then**
16: $\mathrm{pr}_{\min}=\mathrm{pr}_{\min}(k)$;
17: $\mathrm{id}=k$;
18: **end if**
19: **end for**
20: recalculate the link utilization when the i-th path $p_{s,d}^{(i)}$ is exchanged with the path $p_{s,d}^{(\mathrm{id})}$ with minimum priority;
21: **if** $u_i''>100\%$ **then**
22: block the flow;
23: **else**
24: the i-th path $p_{s,d}^{(i)}$ is exchanged with the path $p_{s,d}^{(\mathrm{id})}$ with the minimum priority;
25: LPM sets up the paths for the two flows;
26: update TED database;
27: **end if**
28: **else**
29: LPM sets up the path $p_{s,d}^{(i)}$ for the flow $f(s,d)$;
30: add the information on the flow $f(s,d)$ to the TED database;
31: **end if**
32: **end if**

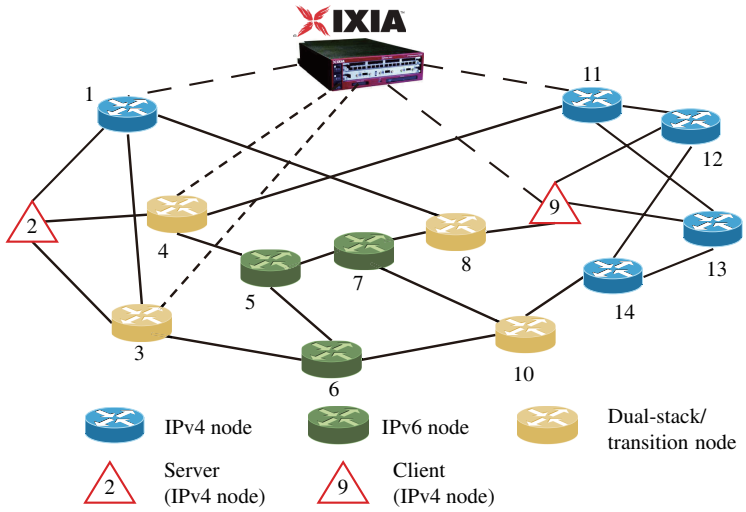


图 4 网络实验拓扑

Figure 4 Experimental topology

3 实验结果及分析

3.1 实验环境

我们采用高性能服务器联想 ThinkServer RD540 搭建了如图 4 实验拓扑, 拓扑模型参考的是 NSFNET 拓扑结构, 图中每个节点表示一个单独的服务器, 在每台服务器上装有 OpenvSwitch 软件交换机, 用于实现传统 IP 路由器或 OpenFlow 交换机功能. 另外, OpenFlow 控制器我们选用的是 POX 控制器[5)], 部署在另外一台单独的服务器上, POX 控制器通过拓展的 OpenFlow 协议与每台 OpenFlow 交换机通信, 可实现路由和相关算法. 同时, 用一台 IXIA 流量生成器为网络中注入背景流量, 使网络中产生拥塞. 为了演示多业务下实现 IP 融合切换对动态流量调控带来的优越性, 我们考虑了网络中同时存在视频串流和文件传输的情形, 并设计了 3 种不同的实验场景下视频串流和文件同步传输实验:

- 场景 1 (无流量调控 without TE): 没有流量调控, 视频串流和文件传输都选择图 4 中节点 2 到节点 9 的最短路径.
- 场景 2 (静态流量调控 static TE): 在节点 3, 4, 8 和 10 上使用传统的双栈节点 IP 路由器. 可以在同构 IP 网络内部 (IPv4 或 IPv6) 通过切换数据流的转发路径来实现流量调控, 但是并不支持 IP 融合切换的功能, 即 IPv4 数据流无法在 IPv6 域中传输, 反之亦然.
- 场景 3 (动态流量调控 F-TE): 在节点 3, 4, 8 和 10 上部署支持 IP 融合切换的 OpenFlow 交换机, 可实现动态流量调控算法, 即可根据网络状态动态地改变 IP 数据包的下一跳转发方式 (IPv4 转发切换成 IPv6 转发, 或 IPv6 转发切换成 IPv4 转发), 利用 IP 融合切换动态地调整网络中的流量分布.

对于视频串流, 我们采用的是一个经过编码的 1080P 的 H.264 视频序列, 视频序列时长为 60 s. 在图 4 的节点 2 上, 视频内容被封装在 IP/UDP/RTP 包头后的数据段内, 并被发送到目的节点 9 上. 节点 9 上的视频客户端接收播放视频, 并记录下视频帧序列号、时间等信息用于数据分析. 同时, 我

5) POX Wiki. http://openflow.stanford.edu/display/ONL/POX+Wiki/.

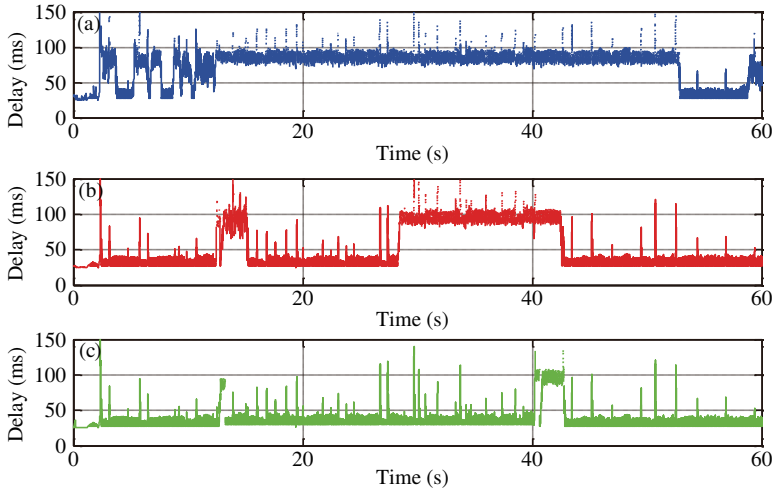


图 5 (网络版彩图) 端到端视频串流传输延时

Figure 5 (Color online) Results on packet-level end-to-end delay of video streaming. (a) Without TE; (b) static TE; (c) F-TE

们采用 Iperf [6] 软件生成 TCP 数据包作为文件传输, 文件传输由节点 2 发往节点 9. 需要注意的是, 因为视频串流对传输时延和抖动有很高的 QoS 需求, 所以在实验中假设视频串流具有更高的传输优先级.

3.2 实验结果和分析

图 5 是 3 个实验场景下视频传输延时的统计结果. 从图中可以明显地看出场景 3 (动态流量调控) 下具有最小的视频传输延时, 因为在这种场景下, 视频串流可以利用 IP 融合切换技术避开拥塞的链路, 例如当 IPv4 域网络比较拥塞时, 视频串流可以通过 IP 融合切换技术切换到 IPv6 域传输, 同样的, 当 IPv6 域拥塞时, 视频串流又可以切换回 IPv4 域. 当然, 我们看到当 IPv4 域和 IPv6 域都发生拥塞时, 比如在 [40, 43] s, 视频串流传输延时增大. 此外, 场景 1 (无流量调控) 下的视频延时结果是最糟糕的, 因为在这种场景下一旦最短路径 (2→1→8→9) 发生拥塞, 由于网络缺少流量调控功能, 视频串流无法避免严重的队列延时. 对于场景 2 (静态流量调控), 当视频串流的最短路径发生拥塞时, 可以通过流量调节使视频串流切换到次短路径 2→4→11→13→9 上, 从而避免拥塞. 然而, 当整个 IPv4 域都变得拥塞时, 视频串流无法避免拥塞路径, 例如链路 1→8 和 4→11 在时间 [28, 42] s 都发生拥塞, 视频串流在这一段时间内延时增大. 我们注意到在所有场景下的延时结果都有相似趋势的噪声峰值出现, 这是由于节点 2 和节点 9 上的操作系统环境所造成的.

图 6 为 3 种场景下接收端视频译码情况. 每条红色竖条表示由于丢包这一视频帧没有被成功译码. 同样的, 场景 3 (动态流量调控) 视频帧丢帧率最小. 图 7 是针对场景 2 和场景 3 统计的接收视频亮度分量峰值信噪比 (Y-PSNR) 统计结果, 注意到在 [10, 20] s, 由于路径发生了切换导致部分视频数据包丢失, 所以 Y-PSNR 曲线出现一小段下降. 与此同时, 也注意到场景 2 (静态流量调控) 的 Y-PSNR 下降的时间比场景 3 (动态流量调控) 下降的时间长, 这是因为当视频串流路径发生拥塞时, 对于场景 2 只会转发到 IPv4 域内的另一条路径, 这导致视频和文件共享一段路径, 从而发生传输竞争, 导致更长时间的丢帧. 而对于场景 3 (动态流量调控), 视频串流很容易切换到 IPv6 域内的一条非

6) Iperf — the TCP/UDP bandwidth measurement tool. https://iperf.fr/.

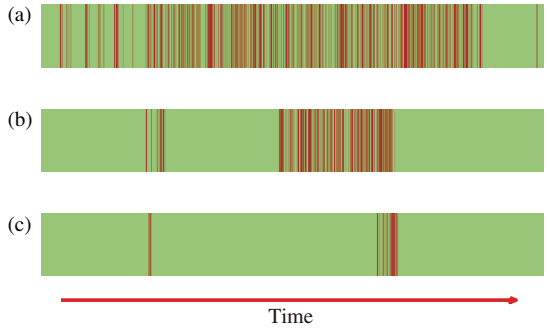


图 6 视频帧可解码性

Figure 6 Results on video frames' decodability. (a) without TE; (b) static TE; (c) F-TE

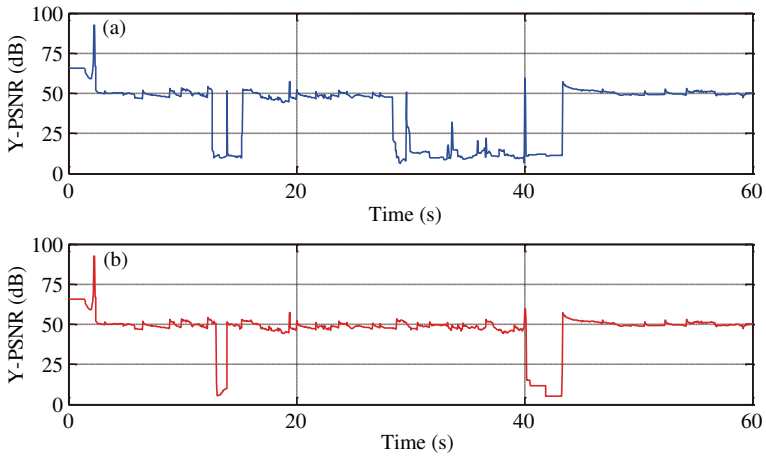


图 7 接收视频 Y-PSNRs

Figure 7 Results on received videos' Y-PSNRs. (a) Static TE; (b) F-TE

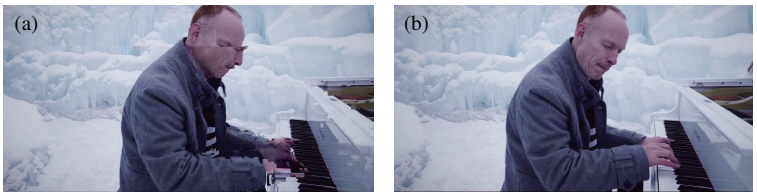


图 8 视频播放截图

Figure 8 Screen-shots of video play-backs. (a) Static TE; (b) Flexible TE

拥塞的路径上, 所以这种情形发生的几率很小. 图 8 展示了场景 2 和场景 3 接收到的视频播放截图, 反映了两种场景下视频传输质量的直观的对比.

图 9 是 IP 融合切换实验的抓包过程. 图 9(a) 是视频在进行 IP 切换之前的数据包截图, 我们可以看到在 IP 切换之前, 视频数据包是封装在 IPv4 数据包内的; 图 9(b) 是视频进行 IP 切换之后的数据包截图, 注意到所有的 IPv4 数据包被封装在了 IPv6 数据包内, 从而实现了 IP 切换.

文件传输的带宽测试结果如图 10 所示. 在这 3 种场景下, 动态流量调控仍然提供了最大的文件传输带宽保证, 其次是静态流量调控. 我们也注意到在第 28 s 之后文件传输带宽下降, 这是因为文件

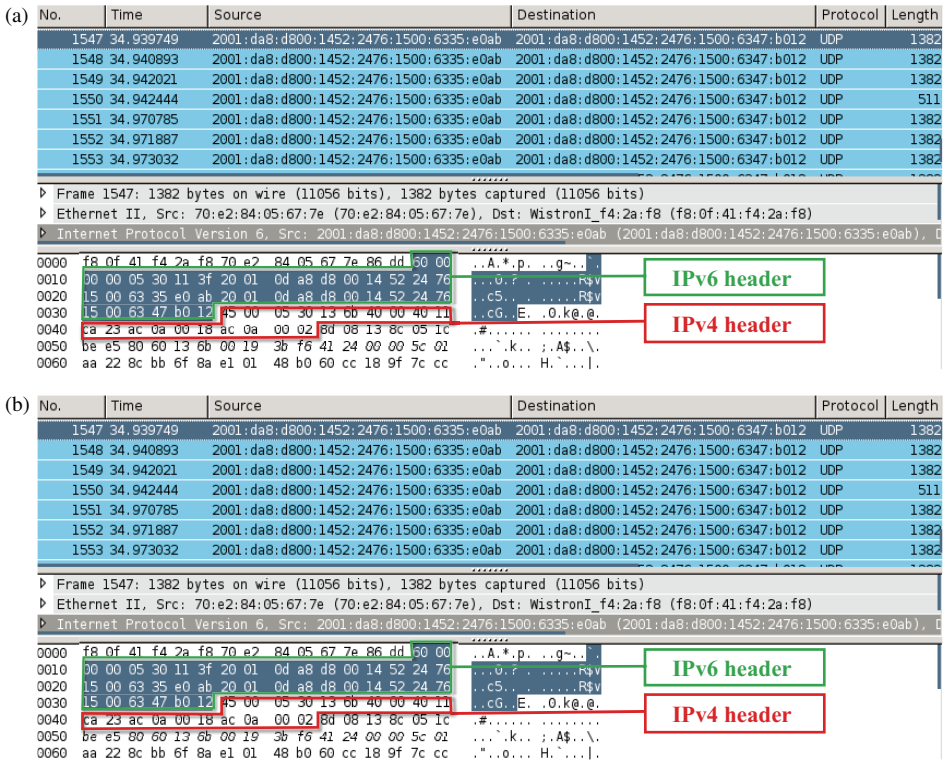


图 9 (网络版彩图) IP 融合切换视频数据包抓包

Figure 9 (Color online) Video packets. (a) Before IP-forwarding interchanging; (b) after IP-forwarding interchanging

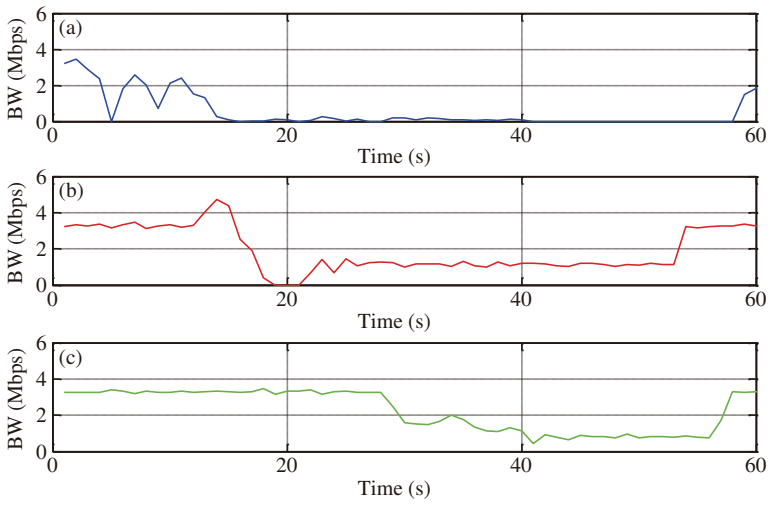


图 10 (网络版彩图) 文件传输带宽结果

Figure 10 (Color online) Results on file transfer bandwidth. (a) Without TE; (b) Static TE; (c) F-TE

的传输优先级小于视频传输优先级, 所以文件传输在特定的情况下 (如视频和文件发生传输竞争时) 必须给视频串流让路, 以保证视频传输的 QoS 需求, 这也证实了动态流量调控策略是 QoS 感知的.

4 结论

本文基于软件定义网络 (SDN) 的概念, 提出了一种借助于 OpenFlow 技术的新颖的 IP 融合切换技术. 在 IPv4 与 IPv6 域共存的异构网络里, 传统的 IP 转发很难实现服务质量 (QoS) 感知的动态流量调控, 而基于 OpenFlow 技术的 IP 融合切换技术可根据当前网络状况对 IP 数据流的下一跳转发方式进行选择, 实现 IPv4 与 IPv6 之间动态高效的转发切换. 为此, 我们设计了一个 OpenFlow 系统, 拓展了 OpenFlow 协议, 并利用服务器和网络交换机搭建了一个真实的网络实验平台, 通过在平台上进行视频与文件同步传输实验, 成功验证了基于 OpenFlow 的 IP 融合切换技术在实现服务质量 (QoS) 感知的动态流量调控的优越性.

参考文献

1 Deering S, Hinden R. Internet protocol, version 6 (IPv6) specification. RFC 2460. https://tools.ietf.org/html/rfc2460. 1998

2 Hu G, Xu K, Wu J, et al. A general framework of source address validation and traceback for IPv4/IPv6 transition scenarios. IEEE Netw, 2013, 27: 66–73

3 Wu P, Cui Y, Wu J, et al. Transition from IPv4 to IPv6: a state-of-the-art survey. IEEE Commun Surveys Tuts, 2013, 15: 1407–1424

4 Ma S, Hu D, Li S, et al. QoS-aware flexible traffic engineering with OpenFlow-assisted agile IP-forwarding interchanging. In: Proceedings of IEEE International Conference on Communications (ICC), London, 2015. 6887–6892

5 Goth G. Software-defined networking could shake up more than packets. IEEE Internet Comput, 2011, 15: 6–9

6 McKeown N, Anderson T, Balakrishnan H, et al. Openflow: enabling innovation in campus networks. ACM SIGCOMM Comput Commun Rev, 2008, 38: 69–74

7 Xia W, Tsou T, Lopez D, et al. A software defined approach to unified IPv6 transition. ACM SIGCOMM Comput Commun Rev, 2013, 43: 547–548

8 Govindarajan K, Setapa S, Meng K C, et al. Interoperability issue between IPv4 and IPv6 in OpenFlow enabled network: IPv4 and IPv6 transaction flow traffic. In: Proceedings of International Conference on Computer, Control, Informatics and Its Applications (IC3INA), Bandung, 2014. 58–63

9 Nordmark E, Gilligan R. Basic transition mechanisms for IPv6 hosts and routers. RFC 4213. https://tools.ietf.org/html/rfc4213. 2005

10 Caria M, Jukan A, Hoffmann M. A performance study of network migration to SDN-enabled traffic engineering. In: Proceedings of IEEE Global Communications Conference (GLOBECOM), Atlanta, 2013. 1391–1396

11 Ghobadi M, Yeganeh S H, Ganjali Y. Rethinking end-to-end congestion control in software-defined networks. In: Proceedings of the 11th ACM Workshop on Hot Topics in Networks. New York: ACM, 2012. 61–66

12 Fraleigh C, Moon S, Lyles B, et al. Packet-level traffic measurements from the Sprint IP backbone. IEEE Netw, 2003, 17: 6–16

OpenFlow-controlled QoS-aware dynamic traffic engineering for differentiated service provisioning in hybrid IPv4/IPv6 networks

Shoujiang MA, Daoyun HU, Shengru LI, Nana XUE, Yan SHAO, Suoheng LI & Zuqing ZHU*

School of Information Science and Technology, University of Science and Technology of China, Hefei 230027*, China*
*E-mail: zqzhu@ieee.org

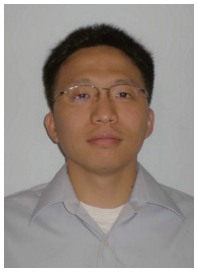
Abstract In this paper, we propose a solution to realize IP-forwarding interchanging based on a software-defined networking (SDN) infrastructure facilitated by OpenFlow (OF). Specifically, we propose to decide an IP packet's next-hop forwarding scheme according to the network status, *i.e.*, switching the packets from an IPv4 domain to an IPv6 one, or vice versa. Then, dynamic traffic engineering can be realized in a hybrid network where IPv4 and IPv6 domains coexist. By extending the OF protocol properly, we design an OF system to realize the proposed IP-forwarding interchanging, propose a dynamic traffic engineering algorithm for providing differentiated services, and implement the algorithm in the OF system. The OF system is then demonstrated with an experimental test bed that consists of real switches and servers to show its advantages. The experiments that include both real-time video streaming and TCP file transfer verify the effectiveness of the proposed system, and show successful QoS-aware dynamic traffic engineering.

Keywords SDN, OpenFlow, traffic engineering, IP-forwarding interchanging, QoS

Shoujiang MA was born in 1989. He received his B.S. degree from the Department of Information Science and Technology, Southwest Jiaotong University (SWJTU), Chengdu, China in 2013. He is working toward his M.S. degree at the School of Information and Technology, University of Science and Technology of China (USTC). His research interests include software-defined networking and next generation network architecture.

Daoyun HU was born in 1990. He received his B.S. degree from the Department of Information Science and Technology, Southwest Jiaotong University (SWJTU), Chengdu, China in 2014. He is working toward his M.S. degree at the School of Information and Technology, University of Science and Technology of China (USTC). His research interest is software-defined networking.

Shengru LI was born in 1989. He received his B.S. degree from the School of Information Science and Engineering, Shenyang Ligong University (SYLU), China in 2013. Currently, he is working toward his Ph.D. degree at the School of Information and Technology, University of Science and Technology of China (USTC). His research interests include software-defined networking and network architecture.



Zuqing ZHU received his Ph.D. degree from the Department of Electrical and Computer Engineering, University of California, Davis in 2007. From 2007 to 2011, he worked at the Service Provider Technology Group of Cisco Systems, San Jose, California, as a senior R&D engineer. In January 2011, he joined the University of Science and Technology of China, where he is currently a full professor. His research interests are software-defined networking and optical networking.