

doi:10.11835/j.issn.1000-582X.2022.09.011

软件定义广域网中基于 IPv6 分段路由的双栈流量调度算法

肖漫漫, 刘骥琛, 李艳丽, 马 迎

(中国人民大学 信息技术中心, 北京 100872)

摘要:针对当前软件定义广域网在应对企业园区部署双协议栈出口链路、园区之间支持双协议栈通信的网络架构下,存在园区任意协议栈出口链路失效时,园区间双栈流量不可通信的问题,提出了一种适用于软件定义广域网的 IPv4/IPv6 双协议栈环境下的流量调度算法。该算法基于快速重路由机制,通过结合基于 IPv6 的分段路由技术分为路径编码算法和报文处理算法 2 部分实现。在广域网双栈出口任一链路失效时将不同园区的双栈流量进行调度。通过在 Mininet 中的仿真模拟验证了算法的可行性,并利用带宽测量工具 iPerf 对链路性能进行测量,测量数据表明利用双栈流量调度算法进行双栈流量调度极大地提高了园区间双栈链路的利用率和服务质量。

关键词:双协议栈;流量调度;基于 IPv6 的分段路由;软件定义广域网

中图分类号:TP311

文献标志码:A

文章编号:1000-582X(2022)09-115-11

Dual stack traffic scheduling algorithm and implementation based on segment routing over IPv6 in software defined wide area network

XIAO Manman, LIU Jichen, LI Yanli, MA Ying

(Information Technology Center, Renmin University of China, Beijing 100872, P. R. China)

Abstract: In order to solve the problem that the dual stack traffic can not be communicated when any protocol stack link fails in software defined wide area network, a dual stack traffic scheduling algorithm for software defined wide area network is proposed. The algorithm is based on fast rerouting mechanism. By combining segment routing over IPv6, it schedules the dual stack traffic of different parks when one of the wide area network dual stack link fails. Through the simulation in mininet, the feasibility of the algorithm is verified, and the link performance is measured by using the bandwidth measurement tool iPerf. The data shows that dual stack traffic scheduling algorithm greatly improves the quality of service and utilization rate of dual stack links between different parks.

Keywords: dual stack; traffic scheduling; segment routing over IPv6; software-defined wide area network

近年来,随着大数据、人工智能、IPv6、在线视频、移动互联网等业务的快速发展,网络流量变得复杂多样,应用种类也越来越多,互联网的带宽资源越来越丰富,企业对网络的弹性、灵活性和 QoS (quality of

收稿日期:2021-08-24

基金项目:赛尔网络下一代互联网技术创新项目(NGII20190101)。

Supported by the CERNET Innovation Project (NGII20190101).

作者简介:肖漫漫(1987—),女,主要从事计算机网络体系结构、软件定义网络、无线网络、边缘计算方向研究,(E-mail) mxiao@ruc.edu.cn。

service)的需求也越来越高。因此业界提出了软件定义广域网(SD-WAN, software-defined wide area network),通过部署 SD-WAN,可将网络资源虚拟化,降低流量成本,提高带宽利用率,加速网络服务分发效率^[1],实现广域网自动网络部署和流量调度。为保障 SD-WAN 可靠性及利用率,SD-WAN 通过部署多链路备份方案,同时接入专线、互联网、LTE/卫星线路,利用 overlay 技术将不同线路混合捆绑,从而实现多链路间的流量调度和复杂服务质量^[2]。而目前业界部署的 SD-WAN 多链路间的流量调度方案均为基于 IPv4 协议栈技术,支持 IPv4/IPv6 的双栈流量调度还没有统一的解决方案。

另一方面,分段路由(SR, segment routing)技术以及软件定义网络(SDN, software defined network)结合受到了人们的青睐。分段路由支持无状态源路由,可以减轻控制器和中间节点的开销,对路径的管理和控制也十分灵活^[3],因此也有一些工作研究了如何基于分段路由技术解决 SDN 中的流量调度问题^[4-7]。其中文献[4]将流量路径进行分解,在考虑 2 段段列表的情况下,开发了离线和在线的流量调度优化算法。文献[5]提出了一种基于分段路由路径变量的数据结构,通过记录经过节点和可能的下一跳节点,达到流量调度时减小消耗计算资源的目的。文献[6]分析了基于分段路由技术的网络流量模型,考虑在流转发时使用单路径而不是会加大计算资源消耗的等价多路径模型。文献[7]提出了一种 SDN 网络中基于分段路由的多路径流量调度算法,该算法在进行路径权重计算后,根据路径权重选择最优路径,并基于分段路由技术构造分段流表下发到边缘交换机。但以上所述流量调度算法仅针对 IPv4 流量,还不能解决双栈流量的调度问题。

当前,随着 IPv6 和分段路由技术结合,演进出的 SRv6(基于 IPv6 的分段路由, segment routing over IPv6)技术作为新一代 IP 网络承载协议,是解决 IPv6 网络流量工程新的方向。笔者提出了针对 SD-WAN 的双栈流量调度(DSTS, dual stack traffic scheduling)算法,在双栈链路失效时将不同园区的双栈流量进行调度。DSTS 算法基于快速重路由机制,通过 SRv6 技术与 SD-WAN 结合,构建了一种复杂且高效的双栈广域网多链路流量调度的解决方案,可以帮助企业在广域网链路失效时快速连通双栈网络,建立智能的“端-边-网-云”的网络架构,并支持基于 SRv6 的功能拓展。

1 关键技术

1.1 SRv6 简介

SRv6 是一种基于 IPv6 数据平面实现的分段路由协议的网络架构,分段路由技术^[8]通过将报文转发路径切割为不同的分段,并在路径的起始点往报文中插入分段信息指导报文转发。这样的路径分段通过 SID(段标识符, segment identifier)来标识。路径的起始节点把这些路径分段组合起来形成段序列(segment list),指引报文按照段列表在网络上将数据包进行转发。为在 IPv6 报文中实现 SRv6 转发,依据 IPv6 原有的路由扩展报文头定义了 SRv6 扩展报文头 SRH(段路由扩展报文头, segment routing header)^[9],SRH 通过携带段列表等信息显式地指定一条 SRv6 路径。由于对双栈流量进行了调度,与文献[4-7]针对 IPv4 的流量调度研究中压入 SID 信息引导流量转发的方式不同,通过在恢复路径中压入 SRH 信息引导流量转发,进而通过在处理节点指定行为指令对流量报文进行处理,最终实现流量调度。在 IETF 的 SRv6 Network Programming^[10]文稿中定义了很多 SRv6 段端节点(Endpoint)行为的指令,SRv6 常见指令功能介绍如表 1 所示。

表 1 SRv6 Endpoint 常见指令功能

Table 1 Function of SRv6 Endpoint

指令	功能简述
End	把下一个 SID 复制到 IPv6 目的地址,进行查表转发
End.X	根据指定出接口转发报文
End.T	在指定的 IPv6 路由表中进行查表并转发报文
End.DX6	解封装报文,向指定的 IPv6 三层邻接转发
End.DX4	解封装报文,向指定的 IPv4 三层邻接转发
End.B6.Encaps	封装外层 IPv6 报文头和 SRH,应用指定的 SRv6 Policy

1.2 FRR 技术

快速重路由(FRR, fast reroute)技术是指在链路故障发生时,能够快速切换到备份路由路径的机制^[11]。基于 FRR 技术,当链路状态失效无法正常传输流量数据时,SD-WAN 控制器会对备份路径进行动态计算,并将计算好的备份路径转发规则部署到相应的网络设备上,以调度中断的流量。

基于以上 2 种技术原理,由于网络发生故障时和故障恢复后会产生环路现象,而 SRv6 通过创建无环的 SRv6 段列表就可以有效地消除网络环路^[12],因此分段路由技术结合 FRR 技术也越来越多地被用来解决链路失效时的路径恢复问题^[13-15]。其中文献[13]分别针对单链路失效和多链路失效提出基于分段路由的快速重路由技术,2 种算法都是在检测到链路失效后,计算出拼接路径,并通过分段路由技术引导业务流量的恢复路径。文献[14]提出基于分段路由的主动式链路故障恢复策略,该策略利用分段路由的方法,通过将链路信息压入数据包头部进行数据包的转发,从而达到链路恢复时减少网络设备消耗流表项资源的目的。

2 DSTS 算法

2.1 系统场景

在多园区网络中,各园区保留独立的园区接入广域网的出口链路,而当园区网络支持双协议栈架构时,各园区接入广域网出口多部署 IPv4/IPv6 双链路,多园区双栈网络拓扑可以归结为如图 1 所示拓扑模型。

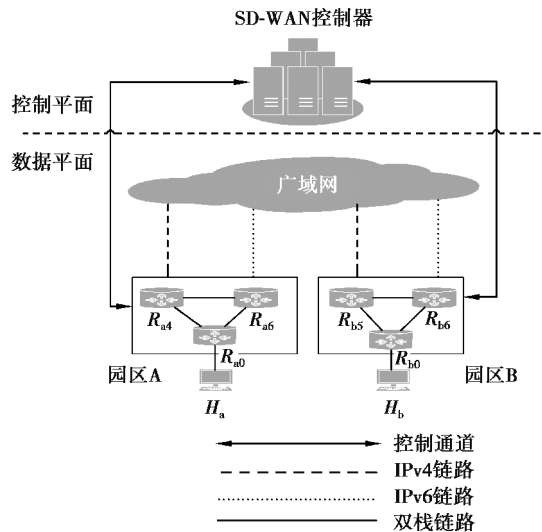


图 1 多园区双栈网络拓扑模型

Fig. 1 Topology of dual stack in multi-branches

如图 1 所示,园区 A 任意主机 H_a 和园区 B 任意主机 H_b 分别通过双栈链路接入广域网,在任意园区 IPv4 和 IPv6 链路有效状态下, H_a 发往 H_b 的 IPv4 流量传输路径为 $H_a \rightarrow R_{a0} \rightarrow R_{a4} \rightarrow \text{WAN} \rightarrow R_{b4} \rightarrow R_{b0} \rightarrow H_b$ 、IPv6 流量传输路径为 $H_a \rightarrow R_{a0} \rightarrow R_{a6} \rightarrow \text{WAN} \rightarrow R_{b6} \rightarrow R_{b0} \rightarrow H_b$ 。假设园区 A IPv4 链路出现故障,即 $R_{a4} \rightarrow \text{WAN}$ 链路失效,此时需在 R_{a4} 节点进行快速重路由,按照现有拓扑结构,恢复路径应为 $R_{a4} \rightarrow R_{a6} \rightarrow \text{WAN}$ 。但是当 IPv4 数据流到达节点 R_{a6} 时,由于没有匹配的路由转发表,最终会导致数据包的丢失。同理,当园区 A IPv6 链路出现故障时,当 IPv6 数据流到达节点 R_{a4} 时,仍会面对数据包丢失的问题。

针对上述应用场景提出 DSTS 算法,该算法在 SD-WAN 的基础上,通过触发 SD-WAN 控制器为双栈数据流重新计算一条路径,实现路由快速恢复,并分为路径编码算法和报文处理算法 2 部分实现双栈流量调度,进而达到双栈业务流量快速恢复的效果。

2.2 数学模型

将计算流量路径描述为单源最短路径模型,其数学模型描述为带权有向图 $G(N, A)$,定义节点 $s \in N$ 为源节点,其他节点为非源节点,路径长度为该路径所包含链路长度之和。计算单源最短路径问题就是找出源

节点 s 到每一个非源节点 j 的有向最短路径,由于流量转发设计仅支持单路径,进一步将原优化模型转换为一个整数线性规划(ILP, inter linear programming)模型,即优化目标是恢复单源路径长度。该模型用到的变量如表 2 所示。

表 2 符号含义

Table 2 Notation meaning

符号	含义
$G(N, A)$	网络拓扑
N	网络节点集合
A	链路集合
s	源节点
t	目的节点
(i, j)	节点 i 到节点 j 的链路, i 表示该链路上游节点, j 表示该链路下游节点
x_{ij}	业务在链路 (i, j) 所占的流量比
c_{ij}	链路 (i, j) 的权重

优化目标函数

$$\text{Minimize } \sum_{(i,j) \in A} c_{ij} x_{ij}, \quad (1)$$

$$s.t.$$

$$\sum_{\{j:(i,j) \in A\}} x_{ij} - \sum_{\{j:(j,i) \in A\}} x_{ij} = 1, i, j \in A \text{ 且 } i = s, \quad (2)$$

$$\sum_{\{j:(i,j) \in A\}} x_{ij} - \sum_{\{j:(j,i) \in A\}} x_{ij} = -1, i, j \in A \text{ 且 } i = t, \quad (3)$$

$$\sum_{\{j:(i,j) \in A\}} x_{ij} - \sum_{\{j:(j,i) \in A\}} x_{ij} = 0, i, j \in A \text{ 且 } i \neq s, t, \quad (4)$$

$$0 \leq x_{ij} \leq 1, i, j \in A, \quad (5)$$

$$\sum_{\{j:(i,j) \in A\}} x_{ij} \leq 1, i, j \in A \text{ 且 } i \neq s, t, \quad (6)$$

$$\sum_{\{j:(j,i) \in A\}} x_{ij} \leq 1, i, j \in A \text{ 且 } i \neq s, t, \quad (7)$$

目标函数是最小化单源恢复路径长度,公式(1)表示 i, j 为链路 (i, j) 的上下游节点,如果链路 (i, j) 是在业务最短路径上,则为 1,否则为 0。公式(2)(3)(4)表示链路流量守恒,若限制数据流转发仅支持单路径,则遵守式(2)(4)。公式(5)表示如果链路 (i, j) 在最短路径上,则值为 0,否则为 1。公式(6)(7)可防止产生环路。

优化目标函数求解算法流程如下:

1)SD-WAN 控制器记录有向图 $G(N, A)$,记录各链路权重 c_{ij} ,并采用 Dijkstra 算法计算最短路径并保存。

2)当出现链路故障时,更新失效链路权重 $c_{st} = 0$,控制器删除故障节点和失效链路,更新网络拓扑为 $G(N, A')$ 。

3)重复步骤 1),根据更新后的网络拓扑,采用最短路径算法重新计算最优路径并保存。

2.3 DSTS 算法设计与分析

在快速重路由的应用基础上,基于 SRv6 提出双栈流量调度算法。设计思路是在园区网络拓扑中单协议栈出口链路失效的场景下,对于受影响的业务流量,SD-WAN 控制器分别以失效链路 2 端的上下游节点为源节点和目的节点计算一条恢复路径,通过路径编码算法计算出并压入 SRH 的内容、通过报文处理算法计算出双栈流量的报文转发方式,进而将业务流量绕过失效链路引导至下游节点。基于 SRv6 的双栈流量调度算法如表 3 所示。

表 3 DSTS 算法

Table 3 Algorithm of DSTS

Algorithm1:DSTS 算法
输入:网络拓扑 $G(N,A)$ 、失效链路以及双栈数据流量
输出:双栈数据流量的调度路径
1 控制器实时监控网络状态,如果发现失效链路
2 更新网络拓扑为 $G(N,A) \leftarrow G(N,A')$
3 以丢包率和跳数为权重采用最短路径算法计算回复路径 L_r
4 通过路径编码算法得到段列表
5 通过报文处理算法在节点计算出双栈流量的报文转发方式
6 在 Segment List 和报文转发方式的引导下将双栈流量绕过失效链路引导至下游节点

在该场景下,SD-WAN 控制器监控整个网络拓扑状态,当链路失效后,控制器更新网络拓扑,将失效链路删除,进而根据最短路径算法计算出恢复路径。如图 2 所示,当链路 $R_{a4} \rightarrow \text{WAN}$ 链路失效时,园区间 IPv4 流量会切换到重新计算的恢复路径 $H_a \rightarrow R_{a0} \rightarrow R_{a4} \rightarrow R_{a6} \rightarrow R_{b6} \rightarrow R_{b4} \rightarrow R_{b0} \rightarrow H_b$ 。假设园区网络内所有网络节点均支持 SRv6,可以把节点 R_{a4} 作为入口节点、节点 R_{b4} 作为出口节点。SD-WAN 控制器监控整个网络拓扑状态,当链路失效后,控制器更新网络拓扑,将失效链路删除,在园区出口 R_{a4} 节点通过 Segment List 引导流量绕过失效链路,并通过 SRv6 作为 underlay 技术穿越广域网,在出口节点(即故障链路的下游节点)弹出所有路径分段并恢复原始报文进行转发,保证双栈流量通过快速重路由生成的替代路由进行转发。采用基于编码算法的方式进行路径编码,以此确定段列表压入哪些 Segment。

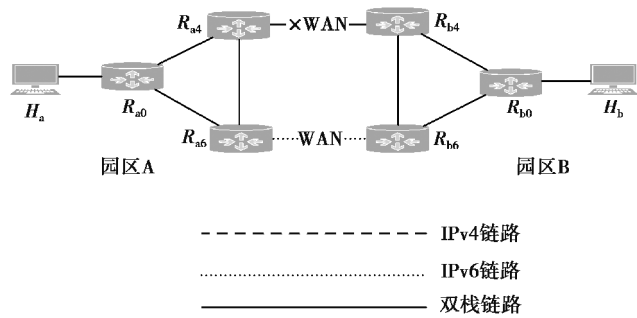


图 2 DSTS 算法拓扑

Fig. 2 Topology of DSTS algorithm

2.4 DSTS 算法实现

针对图 2 所示场景的恢复路径,设计的 DSTS 算法包括路径编码算法和报文处理算法 2 部分,算法伪代码描述分别如表 4、表 5 所示,表 6 为报文处理算法使用的符号及其含义。

表 4 路径编码算法

Table 4 Algorithm of path coding

Algorithm2:路径编码算法
输入:网络拓扑 $G(N,A)$ 和恢复路径 L_r
输出:段列表 Segment List
1 从目的节点开始遍历恢复路径 $L_r \{s, a_1, a_2, \dots, a_k, t\}$
2 for a_i in L_r do
3 if $C_{tai} \leq C_{ait}$ & $C_{tai+1} \leq C_{ai+1t}$ do
4 if $a_i \neq s$ do
5 $SL \leftarrow a_i$ break
6 end for
7 $SL \leftarrow t$
8 end for
9 return SL

表 4 描述了图 2 场景所示的恢复路径编码算法。针对不同协议流量类型,在该网络拓扑中,每条链路的权重值 c_{ij} 是不均等的,采用逆向遍历解法。从失效链路下游节点开始,其中恢复路径源节点 S (即图 2 中 R_{a4}) 为失效链路上游节点、目的节点 T (即图 2 中 R_{b4}) 为失效链路下游节点,节点 X 为恢复路径的中间节点。逆向遍历恢复路径直到找到节点 X 使其路径权重和满足公式(8), C_{TX} 为节点 T 到节点 X 的路径权重和,而 C_{XT} 为节点 X 到节点 T 的路径权重和, Y 为 X 下一跳节点,其他变量同理。

$$\begin{cases} C_{TX} < C_{XT}, \\ C_{TY} \geq C_{YT}, \end{cases} \tag{8}$$

若 X 节点与源节点重合,则压入目的节点 Segment,若 X 节点与源节点不重合,则压入该节点与目的节点 Segment,之后各节点依据表 5 所示的报文处理算法进行报文调度,最终遵循路由协议在网络中进行最短路径转发,图 6 为报文处理处理算法的符号及含义。

表 5 报文处理算法

Table 5 Algorithm of packet

Algorithm3:报文处理算法	
输入:Packet 报文方向、类型、源地址、目的地址四元组 R_{Packet}	
输出:Packet 调度方式	
# 假设报文从园区 A 发送到园区 B	
1 向 R_{a4} 下发流表:	
2 match: $R_{\text{Packet}} = (\text{out}, \text{IPv6}, \text{any}, \text{Hb})$	
3 action: T.Encap(Packet, SL)	
4 match: $R_{\text{Packet}} = (\text{in}, \text{IPv4}, R_{b4}, \text{any})$	
5 action: T.Encap(Packet, SL)	
6 向 R_{a6} 下发流表:	
7 match: $R_{\text{Packet}} = (\text{out}, \text{IPv4}, \text{any}, \text{Hb})$	
8 action: SRH(6over4,Encap(Packet, R_{b4}),SL)	
9 match: $R_{\text{Packet}} = (\text{in}, \text{IPv6}, R_{b6}, \text{any})$	
10 action: END.DX4(Packet)	

表 6 报文处理算法符号及其含义

Table 6 Notation meanings of algorithm 2

符号	含义
Forward(Packet)	该操作不修改接受到的报文,直接根据目的地址进行三层转发
SRH(Packet,SL)	该操作会对 IPv6 报文增加 SRH 报文头,以 SL 作为段列表
T.Encaps(Packet,SL)	该操作会在 IPv4 数据包外层新加一个 IPv6 报头以及 SRH 报头,并可以定义新的 Segment 列表,数据包将首先按照新 IPv6 报头中的 SRH 进行转发。
END.DX4(Packet)	该操作会去掉外层的 IPv6 报头,并将内部的 IPv4 数据包转发给指定的下一跳地址
6over4,Encap(Packet,DA)	该操作会对 IPv6 数据包进行 6over4 封装,封装后的数据包的目的地址设为 DA

通过上述解决方案,可以基于 SRv6 实现受影响的广域网双栈流量的快速恢复,该方案主要解决了在单协议栈广域网链路失效的网络场景下,基于绕过失效链路的原则,以失效链路的上游节点作为源节点针对受影响数据流计算一条恢复路径,并利用路径编码算法得到 Segment List 压入数据包首部,进而依据报文处理算法对数据流报文及其转发方式进行处理,然后通过 Segment 的引导将受影响的数据流通过恢复路径调度至失效链路的下游节点,以此实现在尽可能减少网络重新配置的前提下,数据流仍沿着原有广域网链路进行传输。

3 实验验证与分析

通过实验环境下的仿真与验证,得出该方案可以解决 SD-WAN 单协议栈链路失效时的双栈流量调度问题,采用仿真平台 mininet 搭建测试拓扑,使用 iproute2 进行节点配置,控制器基于 Python2.7 开发实现,实验平台选择 Ubuntu18.04(Linux 内核 4.15.0),流量发生和性能测试工具采用 Iperf 和 Wireshark 软件。

3.1 系统仿真验证园区 IPv4 出口链路失效场景

场景仿真网络拓扑如图 3 所示。

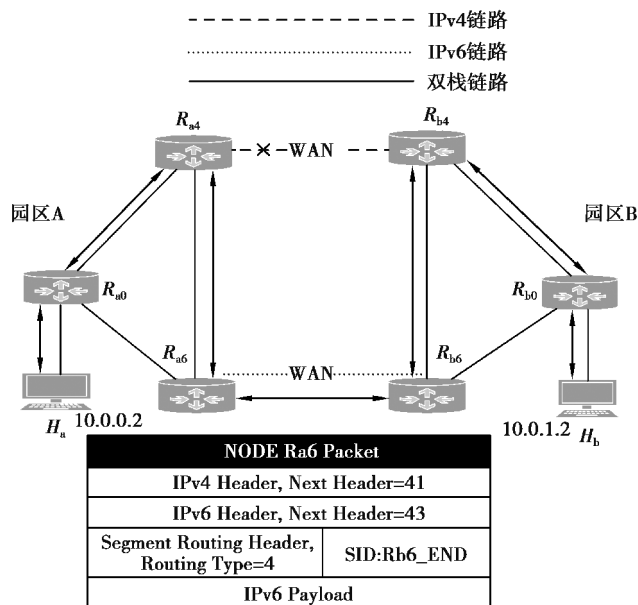


图3 “园区 IPv4 出口链路失效场景”仿真拓扑

Fig. 3 Simulation topology of IPv4 link failure scenario

如图 3 所示,“园区 IPv4 出口链路失效场景”的仿真拓扑由 H_a 和 H_b 2 台主机、6 台路由器和 10 条链接通过运行生成在 mininet 中,其中 2 台主机、6 台路由器均支持 IPv4/IPv6 双协议栈,10 条链接中除 R_{a4} 和 R_{b4} 链接构成 IPv4 广域网链路、 R_{a6} 和 R_{b6} 链接构成 IPv6 广域网链路外,其他均为双栈链路。

在该仿真场景中,断开 R_{a4} 和 R_{b4} 之间链接, H_a 执行至 H_b 的 Ping 命令,数据包在 R_{a4} 处进行 SRv6 的 T.Encaps 压入 Segment,通过仿真 IPv6 广域网后,向 R_{a6} 发送,并转发至 R_{b4} 进行 END.DX4 处理弹出 Segment,还原为 IPv4 报文,向 H_b 进行转发, H_b 收到来自 H_a 的 ICMP-Echo-Request,向 H_a 发送 ICMP-Echo-Reply。同理,Reply 报文按相反方向进行处理转发, R_{a4} 收到 ICMP-Echo-Reply,即收到正确的 Ping 回复。通过在 R_{a4} 、 R_{b4} 进行 Wireshark 抓包分析验证了恢复路径的可行性,并验证了恢复路径流量引导和报文处理,实现了流量调度的目标。

为验证恢复路径网络转发的性能,在 H_a 和 H_b 之间通过 iperf 对网络进行测速,如图 4 测量结果显示链路失效时,基于 SRv6 的恢复路径的网络吞吐率约为 19Gbps,满足中型企业园区间的网络需求。

```

root@ubuntu:/opt/mininet# ./run.py -s 10.0.0.1/24 -t 10.0.1.1/24 -m 3
*** Iperf test
*** h1 : ('ifconfig h1-eth1 mtu 1376 up',)
*** h2 : ('ifconfig h2-eth1 mtu 1376 up',)
*** h2 : ('iperf -s -p 5566 -t 7 &',)
*** h1 : ('iperf -c 10.0.1.2 -p 5566 -t 5',)
-----
Client connecting to 10.0.1.2, TCP port 5566
TCP window size: 85.3 KByte (default)
-----
[ 3] local 10.0.0.2 port 59388 connected with 10.0.1.2 port 5566
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 5.0 sec   11.3 GBytes  19.4 Gbits/sec
*** Starting C.T.

```

图4 “园区 IPv4 出口链路失效场景”iperf 吞吐率

Fig. 4 Throughput of IPv4 link failure scenario

3.2 系统仿真验证园区 IPv6 出口链路失效场景

场景仿真网络拓扑如图 5 所示。

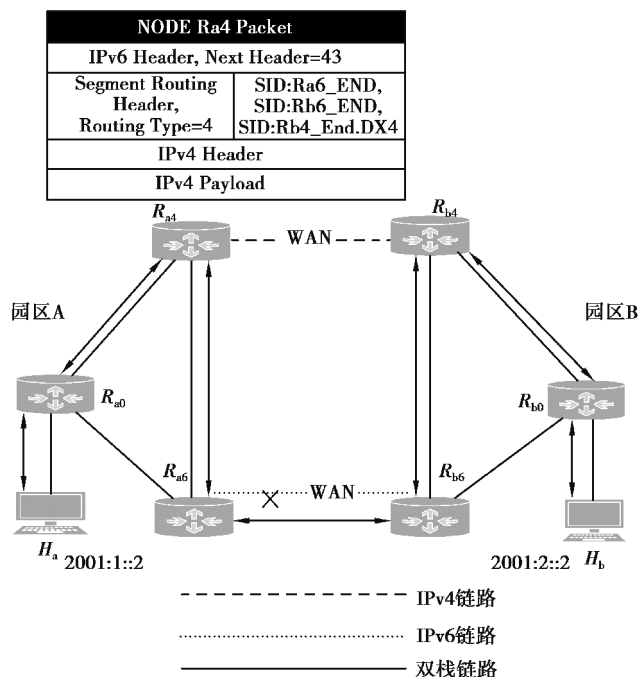


图 5 “园区 IPv6 出口链路失效场景”仿真拓扑

Fig. 5 Simulation topology of IPv6 link failure scenario

在该仿真拓扑中,仿真网络节点设备与图 3 相同,在该仿真场景中,断开 R_{a6} 和 R_{b6} 之间链接, H_a 执行至 H_b 的 Ping 命令,数据包在 R_{a6} 处进行 SRv6 的 T.Encaps 压入 Segment,转发至 R_{a4} 节点后,通过 6over4 隧道封装,最外层以 IPv4 报文头向 R_{b4} 发送,通过仿真 IPv4 广域网后,在 R_{b4} 节点对 6over4 报文头解封装,还原为 SRv6 报文,转发至 R_{b6} 后进行 END.DX4 弹出 Segment,还原为 IPv4 报文,向 H_b 进行转发, H_b 收到来自 H_a 的 ICMP-Echo-Request 请求,向 H_a 发送 ICMP-Echo-Reply,即收到正确的 Ping 回复。通过在 R_{a6} 、 R_{b6} 进行 Wireshark 抓包分析验证了恢复路径的可行性,并验证了恢复路径流量引导和报文处理的处理流程实现了流量调度的预期目标。

为验证恢复路径网络转发的性能,在 H_a 和 H_b 之间通过 iperf 对网络进行测速,如图 6 测量结果显示链路失效时,基于 SRv6 的恢复路径的网络吞吐率约为 1.4Gbps,保障了单协议栈链路失效时企业园区间的双栈网络需求。

```
[ 3] 0.0- 5.0 sec 10.2 Gbytes 17.4 Gbits/sec
*** h2 : ('iperf -s -p 5566 -t 7 &')
-----
Server listening on TCP port 5566
TCP window size: 85.3 KByte (default)
-----
[ 4] local 2001:2::2 port 5566 connected with 2001:1::2 port 53944
[ ID] Interval      Transfer      Bandwidth
[ 4] 0.0- 5.0 sec 10.2 Gbytes 17.4 Gbits/sec
[2] 21478
*** h1 : ('iperf -c 10.1.1.2 -p 5566 -t 5')
-----
Client connecting to 10.1.1.2, TCP port 5566
TCP window size: 85.3 KByte (default)
-----
[ 3] local 10.1.0.2 port 40436 connected with 10.1.1.2 port 5566
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 5.0 sec 832 Mbytes 1.39 Gbits/sec
*** Starting C.T.
```

图 6 “园区 IPv6 出口链路失效场景”iperf 吞吐率

Fig. 6 Throughput of IPv6 link failure scenario

3.3 实验分析

在 Mininet 系统仿真实验中,不同路径和处理方式的报文转发或封装转发吞吐率如图 7 所示、数据包往返时延如图 8 所示、流量丢包率如图 9 所示,仿真数据分析得出如下结论:

1)链路失效时恢复路径只使用 SRv6 处理报文的转发速率与链路正常时 IPv6 路由转发的吞吐率接近,而链路失效时恢复路径通过 SRv6over4 处理转发的吞吐率低于另 2 种方式,但也基本满足中型企业园区的网络需求。

2)链路失效时恢复路径的 3 种报文处理方式的往返时延差异不大,经 SRv6 处理报文的往返时延与链路正常时 IPv6 路由转发的往返时延非常接近,甚至略优,而通过 SRv6over4 处理转发的往返时延稍大于其他两种,但平均值在 0.25 ms 以下,在合理范围内。

3)链路失效恢复路径的 3 种报文处理方式的丢包率均为 0,测试数据表明单协议栈链路失效时,恢复路径的流量引导和报文处理达到了流量调度的目的,提高了双栈 SD-WAN 链路的服务质量。

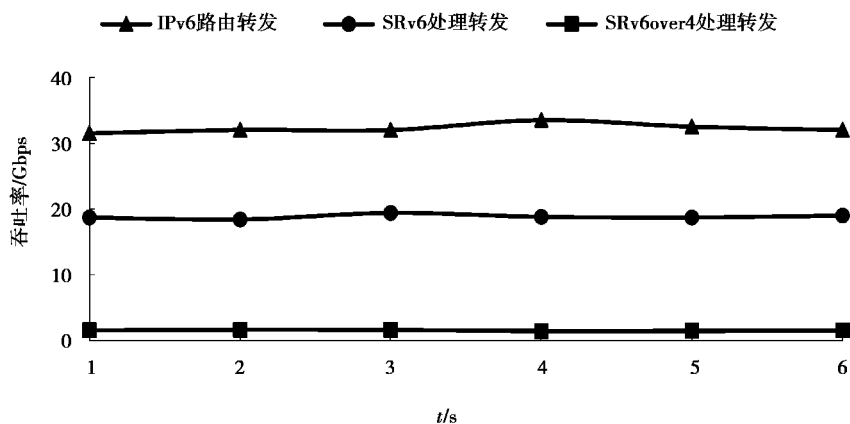


图 7 不同方式下 iperf 吞吐率对比

Fig. 7 Throughput comparison of different forward method

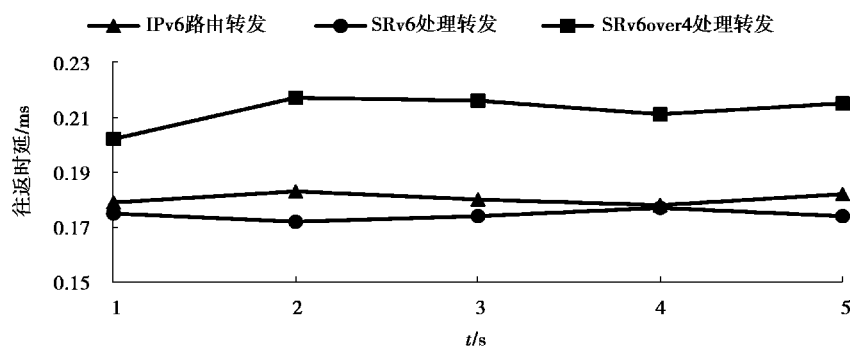


图 8 不同方式下往返时延对比

Fig. 8 RTT comparison of different forward method

由于报文处理流程会导致数据流报文 IP 报头长度增加,可能存在报文长度超过路径最大传输单元 (PMTU, path maximum transfer unit) 的问题,进而导致报文在传输中被分片,大幅影响传输速率。因此,可以在网络接入层对 MTU 进行适当限制,或通过 G-SRv6 (SRv6 头压缩, Generalized-SRv6)^[15] 技术对 SID 长度进行压缩,以减小处理后的报文长度。

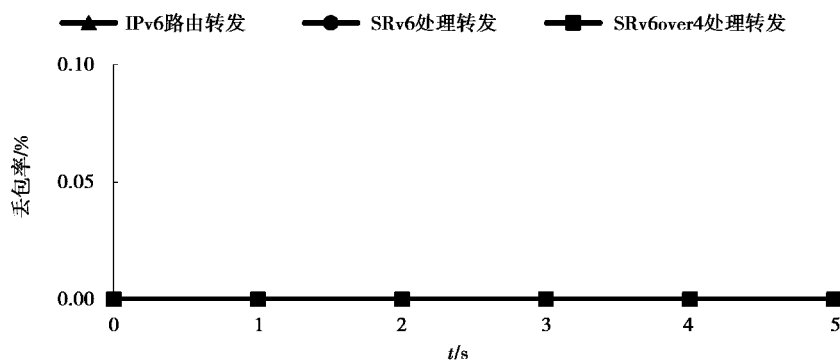


图 9 不同方式下丢包率对比

Fig. 9 Packet loss rate comparison of different forward method

4 结 语

针对现有园区 SD-WAN 网络架构下,单协议栈广域网链路失效时园区双栈流量的互联互通问题,提出了一种 SD-WAN 架构下基于 SRv6 的双栈流量调度算法,该算法基于快速重路由机制,结合 SRv6 技术分为路径编码算法和报文处理算法两部分实现。在链路失效时,通过快速重路由机制确定恢复路径,在恢复路径的基础上,首先通过路径编码算法得出数据流添加 Segment 的类型,引导数据流在恢复路径正确转发传输,然后在网络各节点根据数据流报文信息利用报文处理算法确定报文的转发方式,最终保证双栈流量通过快速重路由生成的替代路由进行转发。最终,在 Mininet 中运行仿真实验,并使用 Iperf 工具进行性能测试,实验仿真结果表明,提出的 DSTS 算法实现了链路失效时的双栈流量调度,增加了企业园区广域网的冗余度和可靠性,提升了企业带宽利用率和服务质量。

参考文献:

- [1] 软件定义广域网(SD-WAN)研究报告[R]. 北京: 中国互联网协会, 2018.
Research on software defined wide area network[R]. Beijing: Internet society of China, 2018. (in Chinese)
- [2] 董炳泉. SD-WAN 三种技术方案的研究[J]. 广东通信技术, 2019, 323(11):68-71.
Dong B Q. Research on technique solutions on SD-WAN[J]. Guangdong Communication Technology, 2019, 323(11):68-71. (in Chinese)
- [3] 董谦, 李俊, 马宇翔, 等. 软件定义网络中基于分段路由的流量调度方法[J]. 通信学报, 2018(11):23-35.
Dong Q, Li J, Ma Y X, et al. Traffic scheduling method based on segment routing in software-defined networking INFOCOM[J]. Journal on Communications, 2018(11):23-35. (in Chinese)
- [4] Bhatia R, Hao F, Kodialam M, et al. Optimized network traffic engineering using segment routing[C]// 2015-IEEE International Conference on Computer Communications. HongKong, China: IEEE, 2015: 657-665.
- [5] Hartert R, Schaus P, Vissicchio S, et al. Solving segment routing problems with hybrid constraint programming techniques[C]// International Conference on Principles and Practice of Constraint Programming. Berling: Springer, 2015: 592-608.
- [6] Schüller T, Aschenbruck N, Chimani M, et al. Traffic engineering using segment routing and considering requirements of a carrier IP network[C]// 2017 IFIP Networking Conference (IFIP Networking) and Workshops. Stockholm, Sweden: IEEE, 2018: 1851-1864.
- [7] 李艺, 唐宏, 马枢清. 软件定义网络基于分段路由的多路径调度算法[J/OL]. 计算机应用研究. <https://doi.org/10.19734/j.issn.1001-3695.2020.07.0183>.
Li Y, Tang H, Ma S Q. Multi-path scheduling algorithm based on segment routing in software defined network[J/OL]. Application Research of Computers Application Research of Computers. <https://doi.org/10.19734/j.issn.1001-3695.2020.07.0183>. (in Chinese)

- [8] Filsfils C, Dukes D, Previdi S, et al. IPv6 Segment Routing Header (SRH)[EB/OL]. (2020-03-14)[2020-03-25]. RFC 8754.
- [9] Filsfils C, Camarillo P, Keddy H, et al. SRv6 Network Programming[EB/OL]. (2019-12-05)[2020-12-27]. Draft-ietf-spring-srv6-network-programming-07.
- [10] Filsfils C, Previdi S, Ginsberg L, et al. Segment routing architecture[EB/OL]. (2018-12-19)[2020-03-25]. RFC 8402.
- [11] Shand M, Bryant S. IP fast reroute framework[EB/OL]. (2015-10-14)[2020-03-25]. RFC 5714.
- [12] 李振斌, 胡志波, 等. SRv6 网络编程: 开启 IP 网络新时代[M]. 北京: 人民邮电出版社, 2020: 178.
Li Z B, Hu Z B, et al. SRv6 network programming: ushering in a new era of IP networks[M]. Beijing: Posts & Telecom Press, 2020: 178. (in Chinese)
- [13] 沈少辉. 基于 SDN 的分段路由技术研究[D]. 成都: 电子科技大学, 2018.
Shen S H. Research on SDN-based segment routing[D]. Chengdu: University of Electronic Science and Technology of China, 2018. (in Chinese)
- [14] 王世双. 基于 SDN 的链路故障恢复技术的研究与实现[D]. 合肥: 中国科学技术大学, 2020.
Wang S S. Research and implementation of link failure recovery technology based on SDN[D]. Hefei: University of Science and Technology of China, 2020. (in Chinese)
- [15] 程伟强, 刘毅松, 姜文颖, 等. G-SRv6 头压缩优化技术研究与应用[J]. 电信科学, 2020, 36(8): 22-27.
Cheng W Q, Liu Y S, Jiang W Y, et al. Research and application of G-SRv6 head compressions optimization technology [J]. Telecommunications Science, 2020, 36(8): 22-27. (in Chinese)

(编辑 侯 湘)