

用于 MPLS 流量工程的 RSVP 信令扩展

Juniper 网络公司，爱立信公司，1999 年

目录

| | |
|---|----|
| 大纲..... | 4 |
| 观点..... | 4 |
| 介绍..... | 4 |
| RSVP 的发展..... | 7 |
| 运行总的看法：RSVP 扩展以支持 LSP 隧道..... | 8 |
| LSP 隧道..... | 9 |
| 建立一条 LSP 隧道..... | 10 |
| RSVP PATH 信息..... | 10 |
| RSVP RESV 信息..... | 11 |
| 预留类型..... | 11 |
| RSVP 隧道信息详细资料..... | 11 |
| PATH 信息..... | 12 |
| LABEL_REQUEST 对象..... | 12 |
| EXPLICIT_ROUTE 对象 (ERO) | 13 |
| RECORD_ROUTE 对象 (RRO) | 14 |
| SESSION , SENDER_TEMPLATE , FLOW_SPEC , 及 FILTER_SPEC 对象 C 型扩展... | 15 |
| SESSION_ATTRIBUTE 对象..... | 16 |

| | |
|----------------------------------|----|
| RSVP 信息..... | 17 |
| LABEL 对象..... | 17 |
| 扩展的 RSVP 是如何建立一条 LSP 隧道..... | 18 |
| PATH 信息..... | 19 |
| LSR1 (输入 LSR) 处的处理..... | 20 |
| LSR2 处的处理..... | 20 |
| LSR3 处的处理..... | 21 |
| LSR4 (输出 LSR) 处的处理..... | 21 |
| RSVP 信息..... | 21 |
| LSR4 (输出 LSR) 出的处理..... | 21 |
| LSR3 处的处理..... | 22 |
| LSR2 处的处理..... | 22 |
| LSR1 (输入 LSR) 处的处理..... | 22 |
| 业务如何通过 LSP 进行传输..... | 23 |
| 现有的 LSP 隧道如何进行重路由..... | 23 |
| 预留类型..... | 24 |
| 固定滤波器 (FF) 类型..... | 24 |
| 共享明确 (SE) 类型..... | 25 |
| LSP 隧道重路由处理..... | 25 |
| 建立初始 LSP 隧道..... | 26 |
| 将来重路由 LSP 隧道..... | 26 |
| 为 Internet 核心而增强的 RSVP 可扩展性..... | 27 |
| 捆绑信息扩展..... | 28 |
| MESSAGE_ID 扩展..... | 29 |
| MESSAGE_ID 对象..... | 29 |

| | |
|------------------------|----|
| MESSAGE_ID_ACK 对象..... | 30 |
| 更新扩展的摘要..... | 30 |
| Hello 协议扩展..... | 30 |
| 结论..... | 31 |
| 参考..... | 31 |
| Internet 草案..... | 31 |
| RFC..... | 32 |
| 参考书籍..... | 32 |
| 其它参考..... | 32 |

大纲

Internet 服务提供商 (ISP) 们不断面临来自于在管理他们的网络高速增长的同时, 维持一个用于重要任务应用的可靠基础结构方面的挑战。多协议标记交换 (MPLS), 由于其支持流量工程, 而在新型公共网络中被作为一项重要的技术。流量工程是通过将大量的用户业务转移到经过服务提供商网络中特定节点的预先设定的路径来实现的。这些预先设定的路径被称作标记交换路径 (LSP)。

这篇白皮书将阐述 IETF 的资源预留协议 (RSVP) 是如何被扩展而能够自动建立穿过服务提供商网络的 LSP。白皮书的第一部分将讨论流量工程的方法, 然后讨论传统的 RSVP 是如何被扩展—支持 MPLS 的 LSP 信令。白皮书的核心部分提供的详细的例子来说明如何使用 RSVP 建立 LSP, 用户业务如何通过 LSP, 及 LSP 在网络故障时如何进行重路由。白皮书总结了最新的 RSVP 扩展如何解决传统 RSVP“软状态”模式所面临的扩展性和稳定性问题。同时, 这篇白皮书还提供了一些 RSVP 的背景信息, 它假设您对在 RFC2205 和 RFC2209 中定义的传统 RSVP 有一个基本的了解。

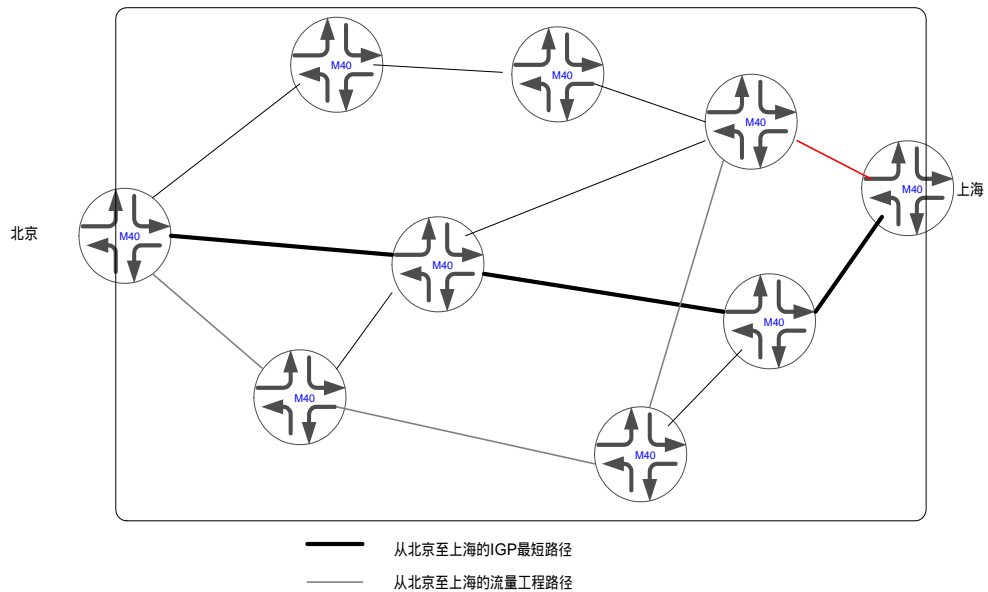
观点

由于手动配置一条 LSP 的开销非常大, 许多服务提供商希望通过使用信令协议自动完成这一处理。LSP 的明确路由可以通过定制的管理工具离线计算, 或使用基于约束的路由在线计算。信令协议在由明确路由计算处理选定的网络节点分配标记及建立 LSP 转发状态。本白皮书中所描述的 RSVP 扩展允许服务提供商动态的建立通过其网络的 LSP, 使网络对不断变化的条件作出更为有效的反映, 同时节约了时间并减少了运行费用。

介绍

流量工程参与将业务流映射到网络物理拓扑上的任务，它提供了将业务流重通过内部网关协议 (IGP) 计算出的最短路径转移到一条具有更少阻塞的路径上去的能力 (图 1)。流量工程的目的

图 1：流量工程



在于在网络中的不同链路，路由器及交换机之间均衡业务流，使这些网络组成部分不会被过分使用或未被充分使用。流量工程使 Internet 服务提供商能够充分使用他们的网络基础结构。

关于 Juniper 网络公司流量工程结构的详细描述，请参考白皮书“*新型公共网络中的流量工程*”。

因为流量工程是一个极为有力的工具，通过将处理自动化及使服务提供商可以简单地在其骨干网中实施流量工程，可以获得许多好处：

- 减少了管理机运行的费用

- 网络运行最为有效
- 在网络压力及不稳定期间进行动态的流量工程
- 为增值业务及附加业务提供可靠的基础结构

Juniper 网络公司的 MPLS 流量工程结构包括 4 个基本组成部分：

- 信息发布机制提供可用网络资源信息 - - 这一部分通过定义相关的 IGP 简单扩展来实现，使链路属性作为每个路由器链接状态广播的一部分。包括最大链接带宽，最大可预留带宽，当前预留带宽，当前带宽使用及链接着色等流量工程扩展被加到 IGP 链接状态广播中去。
- 路径选择处理部分使用 IGP 链接状态广播所发布的信息选择一条满足业务流特定需求的路径 - - 这一处理既可以通过离线计算来实现，也可以通过使用基于约束的路由在线完成。
- 信令部分用来预留资源及由 LSP 路由计算处理选择的网络节点上建立路径状态。在 Juniper 网络公司结构中，这一任务通过配置一些 IETF 认可的 RSVP 扩展来实现，这些扩展使 RSVP 能够在 MPLS 网络中作为信令协议工作。这篇白皮书将主要讨论这些扩展。
- 分组转发部分 (MPLS) 将业务沿由基于约束路由计算所得到的明确路径进行转发 - - 有关 MPLS 的详细讨论及其在服务提供商网络中的应用，请参考 Juniper 网络公司的白皮书“*多协议标记交换：新型公共网络中的增强路由*”。

很明显，负责建立穿过网络的 LSP 状态的信令协议在自动流量工程处理中扮演着重要的角色。一个成功的信令解决方案需要提供于 LSP 隧道运行有关的许多重要任务：

- 提供建立一条区别于“普通”IGP 计算路径的明确路由 LSP 的机制。一条明确路由是一系列预先配置的作为 LSP 物理路径一部分的 LSR。

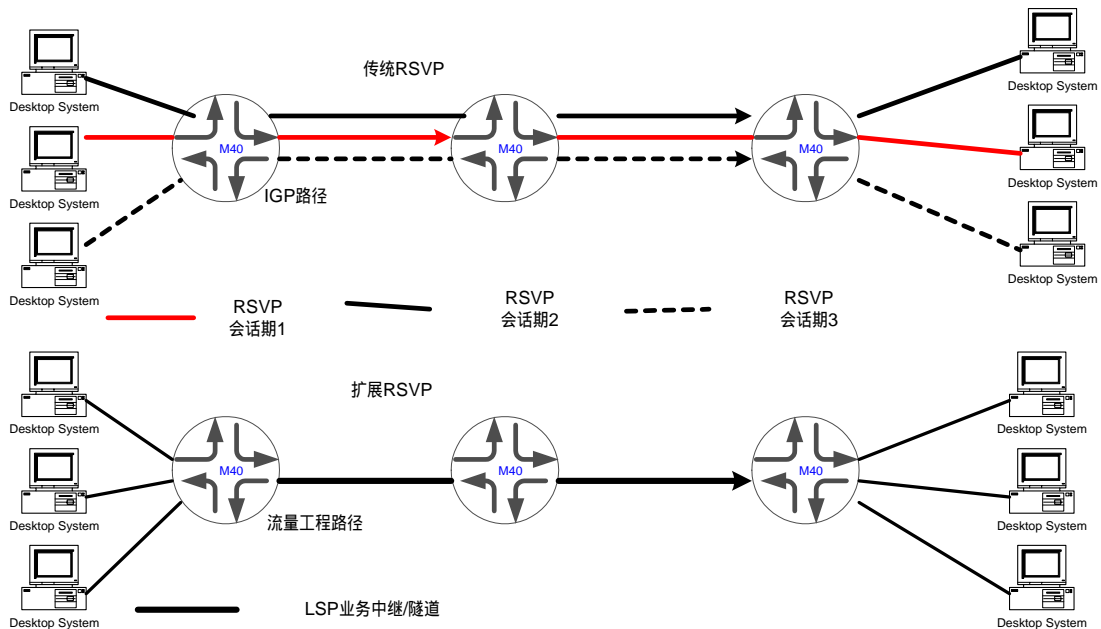
- 提供下行标记分配，发布及按路径中的 LSR 需求进行捆绑，进而在网络节点中建立路径状态。
- 沿路径随机地提供资源预留或服务等级以满足业务流的需求。
- 为网络管理员提供 LSP 使用的实际路由信息。
- 支持全局及局部修复以动态重路由一条 LSP，使其避开网络阻塞及故障。
- 重新分配以前分配的网络资源，通过管理策略控制抢占现存 LSP，以建立新的、承载更为重要业务的 LSP。
- 在 LSP 建立初始及重路由现有 LSP 时阻止循环形成。
- 监测及管理明确路由 LSP 的状态。

Juniper 网络公司流量工程结构的信令部分基于一组 IETF 认可的 RSVP 扩展，以支持所有这些功能要求。

RSVP 的发展

90 年代中期，RSVP 被开发以防止网络阻塞，它通过允许路由器事先决定它们是否能够满足应用流的需求，然后在可能的情况下预留所需资源来完成。最初，RSVP 被设计成为主机间的特定业务流安装与资源预留有关的转发状态。穿过服务提供商网络业务流的物理路径通过传统的基于目的的路由（即，IGP）来确定。到 1997，RSVP 成为被建议的标准，现在被广泛配置在 IP 网络设备中。但是，RSVP 并未在服务提供商网络中广泛应用，因为运行人员关心其需要潜在地支持数百万条主机 - 主机业务流的扩展性及开销。

图 2：传统 RSVP 与扩展的 RSVP 的比较



最初的 MPLS 设备选择扩展 RSVP 成为信令系统以支持 LSP 的建立，使其能够自动地绕开网络故障及阻塞。RSVP 通过自动进行流量工程处理提供了简化网络运行所需的重要部分。作为流量工程信令协议应用 RSVP 与其原本在 90 年代中期开发者的预想大不相同（见图 2）：

- 在基本的 RSVP 规范（RFC2205 和 RFC2209）上增加了许多扩展以支持明确路由 LSP 的建立及管理。
- RSVP 信令发生在一对作为业务中继输入及输出点的路由器（而不是主机对）之间。扩展的 RSVP 安装状态并应用于一组共享一条公用路径和共享网络资源的业务流的集合，而不是应用于单条的主机到主机的业务流。通过汇集许多主机到主机业务流至一条 LSP 隧道，扩展的 RSVP 明显地减少了服务提供商网络核心部分管理所需的 RSVP 状态数量。
- RSVP 信令安装与分组转发有关的分布的状态，包括 MPLS 标记分配。
- RSVP 的软状态模型的扩展性，延迟，及业务开销通过一系列扩展减少了刷新信息数量及相关的信息处理需求。

- 通过 RSVP 信令建立的路径并不被传统的基于目的的路由所限制，因此，它是建立流量工程中继的优秀工具。

在 1997 年初，最初的 MPLS 设备有许多原因选择扩展 RSVP 而不是设计一个全新的信令工具以支持流量工程需求。

- RSVP 原来被设计成为 Internet 资源预留协议，为在一系列多点传送或单点传送传输路径中建立及管理分布的预留状态提供了一个通用的工具。资源预留是流量工程的一个重要的部分，因此，为这一目的继续使用 RSVP 而不是从头开始设计是有意义的。
- RSVP 通过允许承载不透明对象的设计以支持扩展机制 - - RSVP 在其消息部分将对象作为一块不透明的信息承载而在路由器中提供适当的控制模块。这种基本的设计鼓励了对用于建立和维护信息分布状态的新 RSVP 对象的开发，而不是纯粹的资源预留。流量工程的设计者相信可以快速、简单地开发出一系列扩展以增强 RSVP 支持重要的流量工程需求 - - 明确路由及标记发布。
- 被推荐的扩展并未使扩展的 RSVP 实施与传统的 RSVP 实施不兼容 - - RSVP 的实施可以通过对消息部分所包含的对象进行简单地检验容易地区别 LSP 信令和标准的 RSVP 预留。
- 通过实施建议的扩展，RSVP 提供了一个无与伦比的信令系统，它提供了网络管理人员动态建立 LSP 所需的所有需求：
 - 扩展的 RSVP 沿一条明确路由建立 LSP 以支持大型服务提供商对流量工程的需求。
 - 扩展的 RSVP 通过为在 LSP 中的 LSR 分配标记捆绑信息来建立 LSP 状态。
 - 扩展的 RSVP 能够对在 LSP 中的 LSR 预留网络资源（传统 RSVP 的角色）。扩展的 RSVP 同时也允许 LSP 不做特殊的资源预留而承载最努力的业务。

运行上总的看法：RSVP 扩展以支持 LSP 隧道

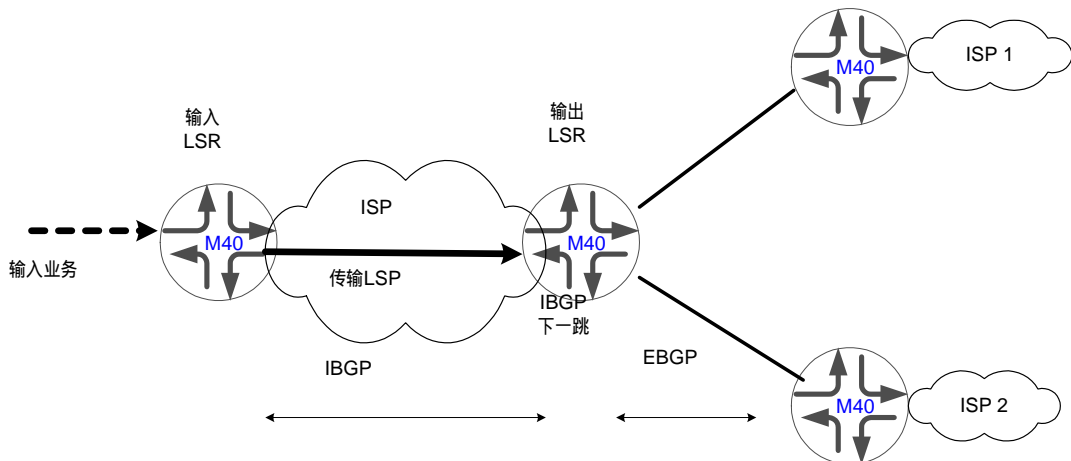
这一部分将提供一个概要的观点来阐述基本的 RSVP 规范是如何被扩展以支持 LSP 隧道的建立。它着重在以下要点：

- LSP 隧道
- RSVP 消息类型
- 预留类型

LSP 隧道

在 Juniper 网络公司的流量工程体系结构中，输入 LSR 基于 IBGP 下一跳决定哪些分组被分配到特定的 LSP。输入 LSR 通过输出 LSR 学习到远端的报头，在同一自治域 (AS) 内的路由器使用 IBGP，而不是 EBGP 来交换路由信息。输出 LSR 使用 EBGP 与不同 AS 内的输入 LSR 进行通信。

图 3：输入 LSR 基于 IBGP 下一跳选择输出 LSR



服务提供商通常希望能够设计一条在输入和输出 LSR 之间的传输路径。依靠 IBGP 下一跳是实现这一任务的方便的方法，同时也提供了将潜在的数千条 IP 报头汇集到一个 MPLS 标记中的好

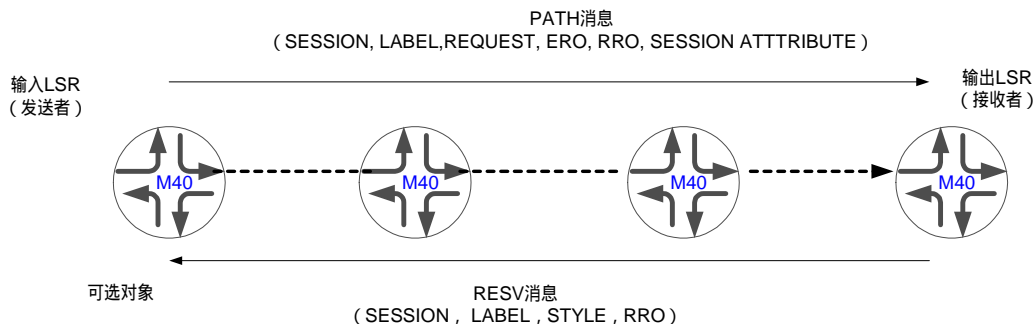
处。这种实现方法可以明显地减少建立穿过服务提供商网络所需 LSP 隧道的数量。同时，这种实现方法提供了一个强有力的工具，减少了在大型服务提供商网络互联交换节点间通信的信息量。在图 3 中，到 ISP 的输入 LSR 通过使用单一的 LSP 将位于或可通过 ISP1 或 ISP2 到达的报头的所有业务转发至到输出 LSR。

一旦输入 LSR 分配给分组一个标记，此标记将有效地定义业务流至穿过服务提供商骨干网的 LSP。LSP 通常是指一条 LSP 隧道，因为业务流通过它时对于 LSP 所包括的每个中间 LSR 是不透明的。为支持 LSP 隧道特性，扩展的 RSVP 定义了一个新的 RSVP 会话期对象，称作 LSP_TUNNEL_Ipv4。这类对象通知每个 LSR，属于特定 LSP 隧道的业务必须基于由此会话期中到上行发送者的本地 LSR 分配的带有特定标记的前一跳进行识别。这里，我们知道，基本的 RSVP 规范只定义了两个会话期 C - 类型，IPv4/UDP 会话期对象和 IPv4/UDP 会话期对象。

建立一条 LSP 隧道

为建立一条 LSP 隧道，输入 LSR 发送一个 RSVP PATH 消息下行至输出 LSR。输出 LSR 通过向输入 LSR 发送一个上行 RSVPRESV 信息对接收到 RSVP PATH 消息作出反应。当输入 LSR 接收到 RSVPRESV 消息时，LSP 被建立，输入 LSR 可以使用 LSP 隧道将业务转发至输出 LSR (图 4)。

图 4：建立一条 LSP 隧道



RSVP PATH 消息

输入 LSR 产生一个带有 LSP_TUNNEL_IPv4 类型的会话期的 RSVPPATH 消息。PATH 消息包含一个 LABEL_REQUEST 对象以请求中间 LSR 和输出 LSR 为此路径提供一个标记捆绑。如果路径中并不是每个 LSR 都支持 LABEL_REQUEST 对象，则此不支持 LABEL_REQUEST 对象路径中的第一个 LSR 会通知输入 LSR。

除 LABEL_REQUEST 对象外，RSVPPATH 消息中还包括其它一些可选对象：

- EXPLICIT_ROUTE 对象 (ERO) - - 可被增加以为穿过服务提供商网络 LSP 指定一条的预先定义路径。当 ERO 出现时，RSVPPATH 信息沿由 ERO 指定的路径向输出 LSR 方向被转发，不受 IGP 最短路径的约束。
- RECORD_ROUTE 对象 (RRO) - - 允许输入 LSR 接收 LSP 隧道穿过服务提供商网络所经过的 LSR 列表。
- SESSION_ATTRIBUTE 对象 - - 可被包含在 RSVPPATH 消息内以保证会话期鉴定及诊断。SESSION_ATTRIBUTE 对象同时业控制路径建立优先级，支持优先级，和本地重路由特性。这些特性将在下面进行讨论。

RSVP RESV 消息

当输出 LSR 接收到包含 LABEL_REQUEST 对象的 PATH 消息时，它通过传输一个包含 LABEL 对象的 RESV 消息作出反应。LABEL 对象包含下行 LSR 与其上行邻居通信所用的标记捆绑。RESV 消息向输入 LSR 的上行方向发送，其方向与 PATH 消息发送方向相反。每个处理带有 LABEL 对象的 RESV 消息的 LSR 对与特定 LSP 相关联的输出业务使用接收标记。当 RESV 消息到达输入 LSR 时，LSP 被建立。

预留类型

每个 LSP 必须采用一个明确的预留类型被建立。预留类型通过输出 LSR 单独地被确定。但是，Juniper 网络公司的实施方案可以通过设置或清除 PATH 消息内的 SESSION_ATTRIBUTE 对象所包含的“输入节点可以重路由位”来允许输入 LSR 向输出 LSR 表达其希望的预留类型（如下所述）。

输出 LSR 可以从 RSVP 固定滤波器（FF）或 RSVP 共享明确（SE）预留类型中选择。传统的 RSVP 通配符滤波器（WF）预留类型因为其合并规则及流量工程目的应用的缺陷而未被使用。正如我们将要在这篇白皮书后面看到的，预留类型是一个非常重要的角色，因为它决定了 RSVP 信令如何支持 LSP 的基本功能——对一条已有的 LSP 隧道进行重路由。

RSVP 隧道消息细节

这一部分将详细讨论为标准 RSVP_PATH 及 RESV 消息支持流量工程所作的 IETF 认可的行令扩展。

PATH 消息

当希望建立一条 LSP 隧道时，PATH 消息将由输入 LSR 向输出 LSR 方向传送。PATH 消息的地址被定义为指向输出 LSR，但是其在 IP 报头中包含了路由器告警 IP 选项（RFC2113）以表示报文需要中间路由器对其进行特殊处理。PATH 消息可以包括多种不同的 RSVP 对象：

- LABEL_REQUEST 对象
- EXPLICIT_ROUTE 对象
- RECORD_ROUTE 对象
- SESSION_ATTRIBUTE 对象

- CoS FLOWSPEC 对象

LABEL_REQUEST 对象

为建立一条 LSP 隧道，输入 LSR 将产生一条包含 LABEL_REQUEST 对象的 PATH 消息。LABEL_REQUEST 对象的存在表示这条 LSP 需要标记捆绑。LABEL_REQUEST 对象同时也包含第三层协议 ID (L3PID) 用于识别将在 LSP 隧道中传输的第三层协议。需要 L3PID 是因为假设 LSP 隧道传输 IPv4 业务是不可能的 - - L3 协议不能够从 L2 报头获得，只能简单的识别 MPLS 为更高一层的协议。

共有 3 种可能的 LABEL_REQUEST 对象类型：

- 请求不定义特定标记范围的标记 - - 这是一种普通的情况，MPLS 标记在位于数据链路层和网络层报头之间的标准 MPLS 添加报头中承载。
- 请求定义了带有最小和最大 VPI 及 VCI 值的 ATM 标记范围的标记 - - 这种类型的请求在 MPLS 标记由第二层 ATM 报头承载时十分有用。
- 请求定义了带有最小和最大 DLCI 值的帧中继标记范围的标记 - - 这种类型的请求在 MPLS 由第二层帧中继报头承载时十分有用。

当一条 PATH 消息到达一个 LSR 时，该 LSR 将 LABEL_REQUEST 对象存储在此 LSP 的本地路径状态模块中。如果定义了标记范围，则标记分配处理必须在这一范围内分配标记。

可能的错误条件包括：

- 如果接收到 PATH 消息的 LSR 识别出 LABEL_REQUEST 对象，但并不能分配标记，其将向输入 LSR 发送一条 PathErr 消息（表示一个路由问题或 MPLS 标记分配故障）。
- 如果接收方不支持 L3PID，其将向输入 LSR 发送 PathErr（路由问题/不支持 L3PID） - - 这个错误将导致 LSP 建立会话期失败。
- 如果收到信息的 LSR 未能识别出 LABEL_REQUEST 对象，其将向输入 LSR 发送 PathErr（未知的对象类别） - - 这个错误将导致 LSP 建立失败。

EXPLICIT_ROUTE 对象 (ERO)

通过对 PATH 消息增加 EXPLICIT_ROUTE 对象 (ERO)，输入 LSR 可以为 LSP 定义一条预先确定的明确路由，而与传统的 IP 路由独立。ERO 只能用于单点传送应用及必须所有明确路由上的路由器都支持 RSVP 和 ERO 的情况。

一条明确路由通过一系列包含在 ERO 内的子对象进行编码。每个子对象能够识别明确路由中的一组节点或定义沿路径执行的一个操作。因此，一条明确路由是路径中一组规定的需要通过的节点和一组需要执行的操作。

每组节点被称作一个抽象节点。如果一个抽象节点只包括一个节点，它被称作简单抽象节点。例如，一条明确路由可只由 AS 号子对象组成。同时每个 AS 可能包含多个跳转点，对于明确路由，它们对源节点是不透明的。图 5 说明了每个子对象是如何在 ERO 中进行编码的。

图 5 : ERO 子对象编码

| L | 类型 | 长度 | 子对象内容 |
|---|----|----|-------|
|---|----|----|-------|

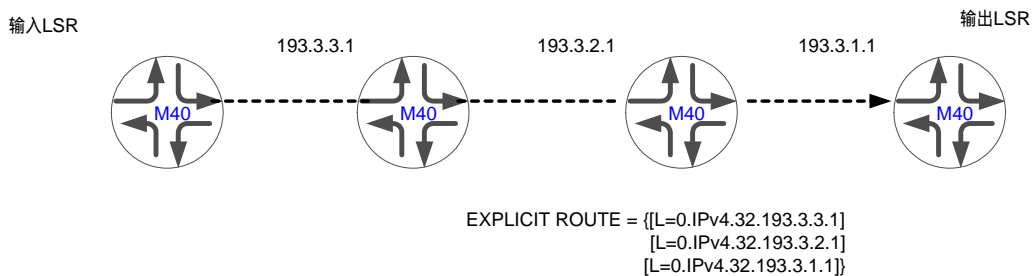
如果设置了 L 位，子对象在明确路由中将描述成疏松跳转。如果 L 位被清除，则其在明确路由中为精确跳转。

目前共定义了 4 种类型的 EXPLICIT_ROUTE 子对象：

- IPv4 报头 - - 识别一个抽象节点包含具有 IP 报头为 IPV4 的一系列节点。具有 32 位长度的报头表示一个 IPV4 节点。
- IPV6 报头 - - 识别一个抽象节点包含具有 IP 报头为 IPV6 的一系列节点。具有 128 位长度的报头表示一个 IPV6 节点。
- 自治域号码 - - 识别一个节点包含一系列属于同一自治域的节点。
- MPLSLSP 终止 - - 表明优先抽象节点应当从所有使用这一 LSP 隧道的分组中移去一级标记堆栈。

图 6 描述了使用 32 位 IPV4 报头定义的精确明确路由。

图 6：明确路由例子



明确路由中疏松节点的存在意味着短时间内在覆盖型路由协议中可能会产生转发循环，LSP 隧道中的循环可以通过 RECORD_ROUTE 对象检测到，如下面所讨论的。

RECORD_ROUTE 对象 (RRO)

通过对 PATH 消息增加 RECORD_ROUTE 对象 (RRO)，输入 LSR 可以接收到 LSP 隧道穿过的实际路由的信息。RRO 的内容是一组被称作子对象的数据项。目前定义了两种类型的子对象，IPV4 地址和 IPV6 地址。

在 RSVP 信令中，RRO 共有 3 种可能的应用：

- 发现层 3 路由循环或明确路由的固有循环，这是因为 RRO 与路径向量类似。
- 一跳接一跳的收集 LSP 建立会话期的最新详细路径信息。
- 通过较小的改变，它可以输入到 EXPLICIT_ROUTE 对象中。其将被用于下一条 PATH 消息中的 EXPLICIT_ROUTE 对象以“锁定会话期路径”。如果 LSP 被锁定，其将不允许改变，即便有更好的路径出现。

当输入 LSR 试图建立一条 LSP 隧道时，它将产生一条包含 LABEL_REQUEST 对象的 PATH 消息。PATH 消息也可以同时包含一个 RRO 对象。最初的 RRO 包含输入 LSR 的 IP 地址。当中间路由器接收到一个包含 RRO 的 PATH 消息时，路由器将在其路径状态模块中存储一个 RRO 的拷贝，并将自己的 IP 地址加到 RRO 中去。当输出 LSR 接收到一个带有 RRO 的 PATH 消息时，它将 RRO 加入到其后来的 RESV 消息中去。在交换了 PATH 和 RESV 消息之后，路径上每个路由器都将会有一个从输入到输出的 LSP 的完全的路由。能够对完全路由进行访问对网络管理的目的是非常有用的。

SESSION，SENDER_TEMPLATE, FLOW_SPEC，和 FILTER_SPEC 对象 C - 类型扩展

扩展的 RSVP 为 SESSION，SENDER_TEMPLATE, FLOW_SPEC，和 FILTER_SPEC 对象定义了新的 C - 类型。C - 类型在一个给定的对象级别内定义了特定的对象类型。例如，在 SESSION 对象级别内，共有 3 种对象 C - 类型：

- IPV4/UDP 会话期 (C - 类型 1)
- IPV6/UDP 会话期 (C - 类型 2)
- LSP_TUNNEL_IPV4 会话期 (C - 类型 7)

SESSION 对象 C - 类型 (LSP_TUNNEL_IPV4)

增加 SESSION 对象至 PATH 消息的目的是识别及诊断会话期。新的 LSP_TUNNEL_IPV4C - 类型包括隧道输出节点的 IPV4 地址和一个唯一的 16 位标识符，这一标识符在 LSP 隧道生命周期内持续保持不变。

SENDER_TEMPLATE 对象 C - 类型 (LSP_TUNNEL_IPV4)

PATH 消息被要求承载 SENDER_TEMPLATE 对象，以描述改特定发送者产生此数据分组时使用的格式。这一模板位于 FILTER_SPEC 表格中，FILTER_SPEC 通常用于在同一链路上的相同会话期中从其它发送者分组中选择出某一发送者的分组。扩展的 RSVP 定义了一个新的 SENDER_TEMPLATEC - 类型 (LSP_TUNNEL_IPV4)，它包含了发送节点的 IPV4 地址和一个唯一的 16 位标识符 LSP_ID，此标识符可以改变以允许发送者共享自己的资源。这个 LSP_ID 将在对使用共享明确 (SE) 预留类型建立的 LSP 隧道进行重路由时使用 (见“ 现有的 LSP 隧道如何进行重路由”)。

FLOWSPEC 对象 C 类型 (CLASS_OF_SERVICE)

FLOWSPEC 是一个传统的 RSVP 对象，它定义了业务流所希望的 QoS。FLOWSPEC 通常包括：

- 服务号码 - - 定义控制负载及保证的服务
- TSPEC (业务指标) - - 描述发送者希望产生的业务。TSPEC 可选择令牌桶滤波器的形式或峰值速率的上缘。
- RSPEC (服务请求指标) - - 定义希望的 QoS。RSPEC 的内容及形式只对应特定的服务。RSPEC 可以包含分配给业务流的带宽，最大迟延，或分组丢失率等信息。

在许多情况下，LSP 并不要求象在传统的综合业务模型中定义的特定带宽预留或 QoS 保证。当特定的资源并未分配给 LSP 时，服务等级 FLOWSPEC 将在 RSVPPATH 消息中显示。服务等级 FLOWSPEC 允许 RSVP 建立一条只提供最努力业务的 LSP 隧道，而无任何特定的资源预留。

当 LSR 在沿 LSP 隧道转发每个分组时，服务等级 FLOWSPEC 提供了应当使用的 CoS 值。值为 0 时表示相关的业务为最努力业务。这项工作已经在 IETF 内部开始以更新 IPTOS 位（现在被称作 DS 字节）。

FILTER_SPEC 对象 C - 类型 (LSP_TUNNEL_IPV4)

FILTER_SPEC 连同 SESSION 对象定义了一系列由 FLOWSPEC 对象定义了服务的数据分组（业务流）。扩展的 RSVP 定义了一个新的 FILTER_SPEC - 类型 (LSP_TUNNEL_IPV4)，它包含了发送者节点的 IPV4 地址和一个唯一的 16 位标识符 - - LSP_ID，此标识符可以被更改以允许发送者能够共享自己的资源。这个 LSP_ID 将在对使用共享明确 (SE) 预留类型建立的 LSP 隧道进行重路由时使用（见“现有的 LSP 隧道如何进行重路由”）。

SESSION_ATTRIBUTE 对象

SESSION_ATTRIBUTE 对象可被添加到 PATH 消息内以控制 LSP 优先级，抢占，及快速重路由功能。

目前在 SESSION_ATTRIBUTE 的 8 为标志位中定义了 3 位：

- 本地保护位 - - 如果设置，传输 LSR 可以使用本地修复机制，其可能导致妨碍 EXPLICIT_ROUTE 对象。通过对这一位进行设置，输入 LSR 通知传输 LSR 它们可以提供 JUNOS“快速重路由”功能。

- 合并允许位 - - 如果设置，传输 LSR 能够将此会话期与其它 RSVP 会话期合并以减少下行传输 LSR 的资源消耗。这一位并未在当前的 MPLS/RSVP 实现中使用。
- 输入节点可以重路由位 - - 如果设置，输入节点可在不拆除 LSP 的情况下对其进行重路由。通过对这一位进行设置，输入 LSR 通知输出 LSR 当对相应的 RESV 消息作出反应时使用共享明确 (SE) 预留类型。

建立优先级定义了此 LSP 隧道在获取其它现有 LSP 资源时的优先级。该值的范围从 0 到 7，0 为最高优先级。建立优先级在决定一条 LSP 隧道是否能够抢占另外一条 LSP 时被使用。

保持优先级定义了此 LSP 隧道在其它 LSP 试图占用其资源时保持资源的优先级。该值范围从 0 到 7，0 为最高优先级。保持优先级在决定一条 LSP 隧道是否能够被另一条 LSP 抢占时使用。

RESV 消息

RESV 消息从输出 LSR 向输入 LSR 传输以响应 PATH 消息的接收。RESV 消息通过发布标记捆绑，研路径请求资源预留，及定义预留类型 (固定滤波器或共享明确) 在每个 LSR 中建立路径状态。

RSVPRESV 消息包含多种不同的 RSVP 对象：

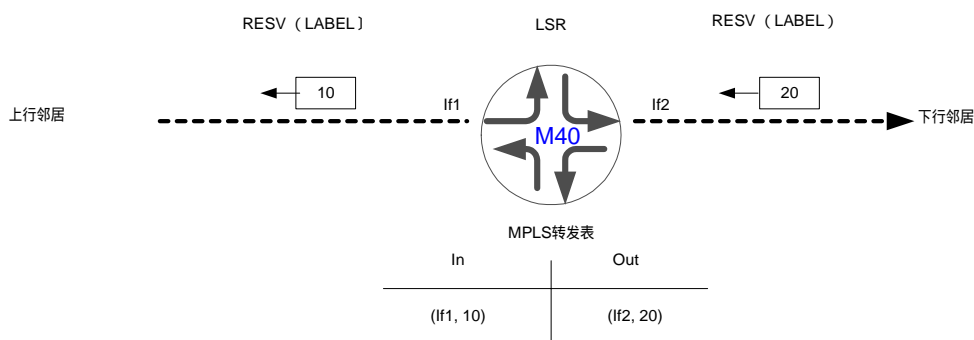
- LABEL 对象 - - 提供“ 按需上行” 的标记分配处理
- RECORD_ROUTE 对象 - - 将 LSP 路径还给 PATH 消息的发送者
- SESSION 对象 - - 单独识别建立的 LSP
- STYLE 对象 - - 定义预留类型 (固定滤波器或共享明确)

LABEL 对象

LABEL 对象在 RESV 消息中承载。这一对象可以包含单一的 MPLS 标记或标记堆栈。如果 MPLS 实施方案不支持标记堆栈，则只有顶端的标记被检验。对于 FF 和 SE 预留类型，标记将被提供给 LSP 的每个发送者。

图 7 描述了一个 LSR 是如何处理 RESV 消息中的 LABEL 对象的：

图 7：LSR 处理 LABEL 对象



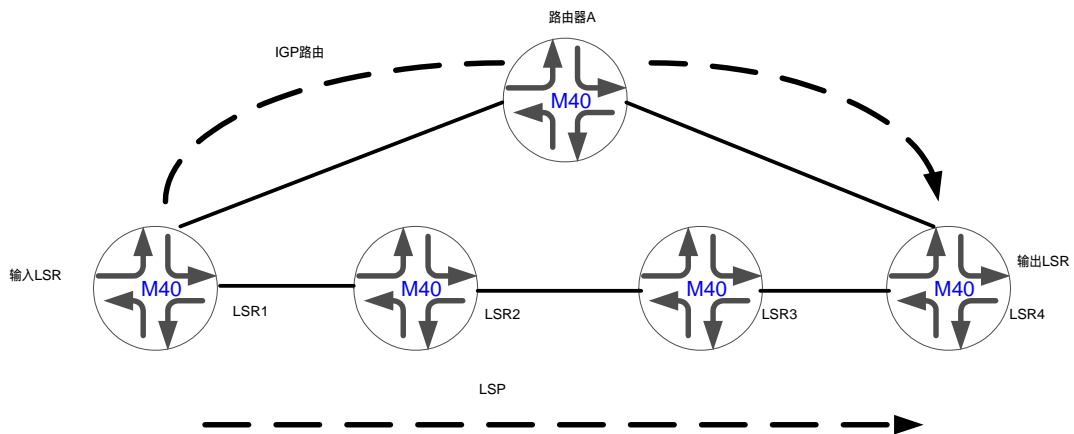
- 当 LSR 收到一个与以前 PATH 消息相关的 RESV 消息时，它确定该 RESV 消息是由 LSP 下一跳发送。LSR 将输入标记与接收接口进行捆绑，并使用这一捆绑将与 LSP 相关的业务转发到其下行邻居。图 7 中，LSR 将所有通过 LSP 隧道的业务流由接口 2 转发出去，并同时带有标记值 20。
- 当 LSR 接收到 RESV 消息时，它将一个本地分配的标记（在此例中为 10）捆绑至 LSP 的输入接口（接口 1）。输入接口是指 LSR 接收到与此 LSP 有关的 PATH 消息的接口。
- LSR 使用值 10 建立一个新的 LABEL 对象，使用新的标记值替换接收到的 RESV 消息中的顶端标记，然后将 RESV 转发到 LSP 中的前一跳。LSR 使用存储在 LSP 路径状态模块中的信息确定前一跳，路径状态模块是使用接收到的初始 PATH 消息建立的。

在不同接口上接收到的 RSVP 消息中的标记通常被认为是不同的，即使标记值是相同的，因为每个标记都只有本地链接范围，而不是全局范围。

扩展 RSVP 如何建立 LSP 隧道

这一部分将讲述扩展的 RSVP 如何将来一条 LSP 隧道。图 8 描述了这一例子将使用的拓扑。

图 8：网络拓扑



假设：

- LSR1，LSR2，LSR3，和 LSR4 上都配置了 MPLS 和 RSVP 并且都被使能。
- 通过某些机制，LSR1 知道 LSP 需要遵循明确路由（LSR1 至 LSR2 至 LSR3 至 LSR4）。

- ERO 的每个抽象节点的 L 位都被清除 (明确路由中的精确跳转) , 并且是一个简单抽象节点 (只由一个通过 32 位 IPV4 报头定义的节点组成) 。

我们希望建立一条 LSP 以传输从 LSR1 进入服务提供商网络并由 LSR4 离开服务提供商网络的业务。传输的业务应当采用 LSP 的物理路径而不是通过 IGP 计算得到的路径 (LSR1 至路由器 A 至 LSR4) 通过网络。其结果是 , 所有从 LSR1 进入服务提供商网络的传输业务 (LSR4 作为 IBGP 下一跳) 都将沿 LSP 进行转发。

LSP 物理路径通过另一种处理被选择是为了 :

- 减少 IGP 路由上的业务流总量。
- 优化网络资源的总体使用效率。
- 增强业务流的业务导向性能参数。
- 增强整个网络的业务导向性能参数。

PATH 消息

LSR1 负责初始化 LSP 的建立。遵从标志 RSVP 信令的过程 , LSR1 发送 PATH 消息至 LSR4。PATH 消息在去往 LSR4 的途中经过 LSR2 和 LSR3。

LSR1 (输入 LSR) 处的处理

1.为建立 LSP , LSR1 建立 PATH 消息 , 其包含 :

- EXPLICIT_ROUTE 对象 (ERO) - - 描述为建立 LSR1 与 LSR4 间的 LSPPATH 消息所应遵循的物理路径。

- LABEL_REQUEST - - 表明路径上所有 LSR 都要求进行 LSP 的标记捆绑，及 LSP 需承载由 L3PID 定义的协议。

PATH 消息同时也可包含下列 RSVP 信令扩展：

- RECORD_ROUTE 对象 (RRO) - - 允许输入 LSR 接收实际路由路径的信息。这一信息对于循环检测及诊断很有用处。
- SESSION 对象 - - 唯一定义此 LSP 隧道。
- SESSION_ATTRIBUTE - - 控制 LSP 优先级，抢占，及快速重路由。

PATH 消息同时也包括下面一些标准的 RSVP 对象：

- SENDER_TEMPLATE - - 包含发送者的 IP 地址及可能的一些识别 PATH 消息发送者的信息。
- SENDER_TSPEC - - 描述将沿 LSP 发送的业务流的业务参数。LSR4 使用这一信息建立适当的 RECEIVER_TSPEC (描述业务流) 和 RSPEC (定义希望的 QoS)。TSPEC 及 RSPEC 的格式和内容对 RSVP 不透明，其由 IETF 的综合业务工作组定义。

2.如果 LSP 希望承载最努力业务，不要求分配资源，则控制负责服务被要求具有零突发及零速率。

3.PATH 消息将沿 ERO 定义的路由向 LSR4 传输。这里，PATH 消息的目的为输出 LSR，但其包含路由告警 IP 选项以表示此数据报需要中间路由器进行特殊处理。

LSR2 处的处理

- 1.当 PATH 消息到达 LSR2 时，它在其路径状态模块中记录 LABEL_REQUEST 对象及 ERO。路径状态模块同时也包括前一跳的 IP 地址，会话期，发送者，及 TSPEC。这些信息用于将相关的 RESV 消息路由回 LSR1。
- 2.LSR2 沿 ERO 定义的路径将 PATH 消息向 LSR4 转发。
- 3.如果 LSR2 不能为 LSP 分配一个标记，它将通过发送一个带有“未知对象级别”的路径错误信息给 LSR1 作出反应。

LSR3 处的处理

- 1.当 PATH 消息到达 LSR3 时，它在其路径状态模块中记录 LABEL_REQUEST 对象及 ERO。路径状态模块同时也包含前一跳，会话期，发送者，和 TSPEC。这些信息将用于把相关的 RESV 消息路由回 LSR2。
- 2.LSR3 沿 ERO 定义的路径将 PATH 消息向 LSR4 转发。
- 3.如果 LSR3 不能为 LSP 分配一个标记，它将通过发送一个路径错误消息给 LSR1 作出反应。

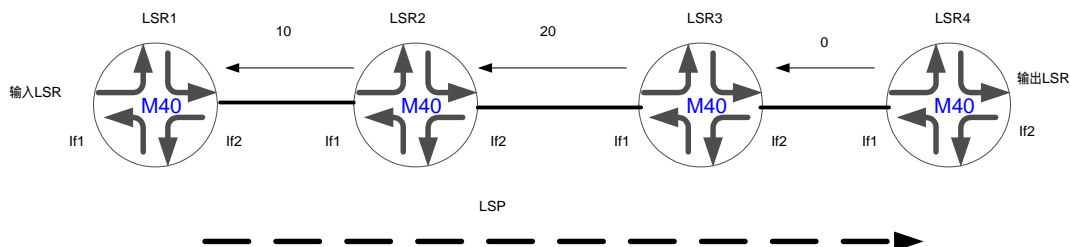
LSR4 处的处理 (输出 LSR)

- 1.当 PATH 消息到达 LSR4 时，它将从 LABEL_REQUEST 对象中得知其为 LSP 的输出 LSR。

RESV 消息

遵循标准的 RSVP 处理过程，LSR4 为会话期产生一个 RESV 消息以分配标记和为 LSP 隧道建立转发状态 (图 9)。RESV 消息的 IP 目的地址是前一跳节点的单点传送地址，从 LSR 的本地路径状态模块中获得。

图 9 : RESV 消息发布标记



在 LSR4 处的处理

1. LSR4 分配给标记的值为 0，并将其放置在 RESV 消息中的 LABEL 对象内。值为零对 LSR4 有特殊意义。当 LSR4 接收到一个标记值为零的分组时，它知道其为 LSP 的输出 LSR。LSR4 简单地弹出标记并依据包含在 IP 报头内的目的 IP 地址对分组进行转发。
2. 如果 LSR1 在 PATH 消息内插入了一个 TSPEC，LSR4 将使用这一信息建立一个适当的接收者 TSPEC 及 RSPEC。
3. RESV 消息通过 LSR3 向回传送。RESV 消息并不承载一个“反向”ERO 去寻找回到 LSR1 的路径。作为替代，RESV 将依照 RSVPPATH 消息在路径状态模块中建立的反向路径进行传输。在某种程度上，PATH 消息留下了一些痕迹以允许 RESV 消息循着此反向路径返回 LSR1。

LSR3 处的处理

1. LSR3 接收到包含由 LSR4 分配的标记的 RESV 消息。
2. LSR3 将标记 (0) 作为 LSP 预留状态的一部分存储。LSR3 在沿 LSP 向 LSR4 转发输出业务时使用这一标记。

3.LSR3 分配一个新的标记 (20) 并将其放置在 RESV 消息的 LABEL 对象中 (替换接收到的标记) , 然后将其上行发送向 LSR2。LSR3 使用这个标记来识别 LSP 上来自 LSR2 的输入业务。

LSR2 处的处理

- 1.LSR2 接收到包含由 LSR3 分配的标记的 RESV 消息。
- 2.LSR2 将标记 (20) 作为 LSP 预留状态的一部分进行存储。LSR2 在沿 LSP 向 LSR3 转发输出业务时使用这一标记。
- 3.LSR2 分配一个新的标记 (10) 并将其放置在 RESV 消息的 LABEL 对象中 (替换接收到的标记) , 然后将其上行发送向 LSR1。LSR2 使用这一标记来识别 LSP 上来自 LSR1 的输入业务。

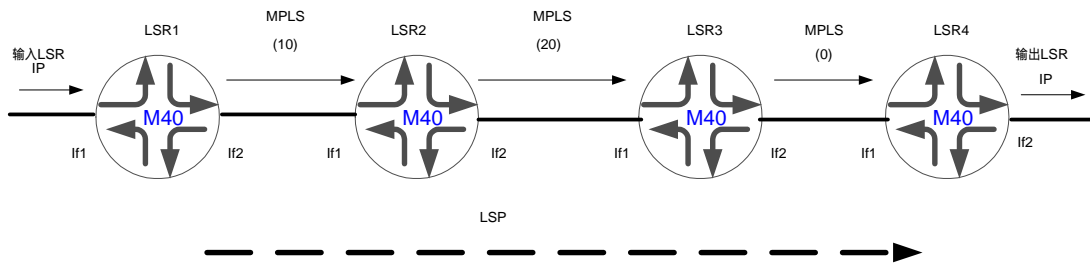
LSR1 处的处理

- 1.LSR1 接收到包含由 LSR2 分配的标记的 RESV 消息。它对所有映射到这一 LSP 的输出业务使用这个标记。
- 2.作为这些操作的结果 , LSP 依照 ERO 内定义的确切路由路径从 LSR1 到 LSR4 被建立。LSR1 通过将来自 LSR2 接收到的标记 (10) 压入标记报头对报头为 X 的业务转发向 LSR2。

传输业务如何穿过一条 LSP

假设 LSP 象例 1 中描述的那样被建立。这个例子描述了传输业务是如何通过 LSP 被转发的 (图 10) 。

图 10 : 通过 LSP 转发业务



1. 一个标准 IP 分组到达 LSR1 的接口 1。在其 IP 路由表中执行完最长匹配查询后，LSR1 发现最佳的匹配为前缀 X，而且所有与前缀 X 匹配的业务应在接口 2 进行转发并带有标记 = 10。LSR1 将 IP 分组封装至一个 MPLS 帧，将标记 10 压入标记头，然后将 MPLS 分组从接口 2 转发出去。
2. LSR2 从接口 1 接收到一个标记 = 10 的 MPLS 分组。它基于 (接口，标记) 对进行固定长度查询，得到分组应被从接口 2 转发出去并带有标记 = 20。LSR2 交换现有的标记 (10)，从接口 2 将 MPLS 分组传送出去并带有标记 = 20。
3. LSR3 从接口 1 接收到标记 = 20 的 MPLS 分组。它基于 (接口，标记) 对进行固定长度查询，得到分组应从接口 2 转发出去并带有标记 = 0。LSR3 交换现有的标记 (20)，从接口 2 将 MPLS 分组传送出去并带有标记 = 0。
4. LSR4 从接口 1 接收到标记 = 0 的 MPLS 分组。因为 MPLS 帧具有标记值为 0，LSR4 知道它为 LSP 隧道的输出 LSR，它必须基于包含在分组 IP 报头内的目的地址作出转发决定，而不是执行 MPLS 查询。LSR4 在其 IP 路由表中执行最长匹配查询，发现此 IP 目的地址最佳匹配为前缀 Y，所有匹配前缀 Y 的业务都将作为普通的 IP 业务在接口 2 上进行转发。

如何对现存的 LSP 隧道进行重路由

流量工程的基本要求之一是对一条已建立的 LSP 隧道进行重路由。对于流量工程，可能有许多原因需要对 LSP 隧道进行重路由：

- 管理策略在一条“更优”的路由出现时可能需要一条 LSP 被重路由。

- LSP 路径中的链路或路由器故障通常需要 LSP 被全局或本地重路由。
- 当发生故障的链路或路由器恢复运行时，管理策略可能要求一条曾被重路由的 LSP 恢复到原来的路径。

当一条 LSP 被重路由时，用户的业务流不被中断是非常重要的。平稳的转换要求对一个被称为“中断前产生”概念的支持 - - 新的 LSP 隧道必须被建立，业务必须在旧的 LSP 隧道被拆除前被转移。RSVP 信令的好处之一就是传统的共享明确预留类型为这一具有挑战性的问题提供了极佳的解决方案。

在被旧的和新的 LSP 隧道共享的链路上，很基本的，由旧 LSP 隧道使用的资源不能在业务转换到新的 LSP 隧道前被释放。但是，至关重要的一点是，预留不能在共享链路上被计数两次，因为这样可能导致 RSVP 管理控制由于资源缺乏而拒绝新的 LSP 隧道。共享明确预留类型允许旧的和新的 LSP 隧道共享它们共同所在链路的单一预留，使新的 LSP 隧道不必要因为链路资源的缺乏而必须等到旧 LSP 隧道清除后才能完成。

预留类型

每个 LSP 必须使用一个明确的预留类型被建立，此预留类被输出 LSR 单独地确定。输出 LSR 可以从任何一种 RSVP 预留类型中选择。这一部分将阐述这些可选预留类型的操作参数。

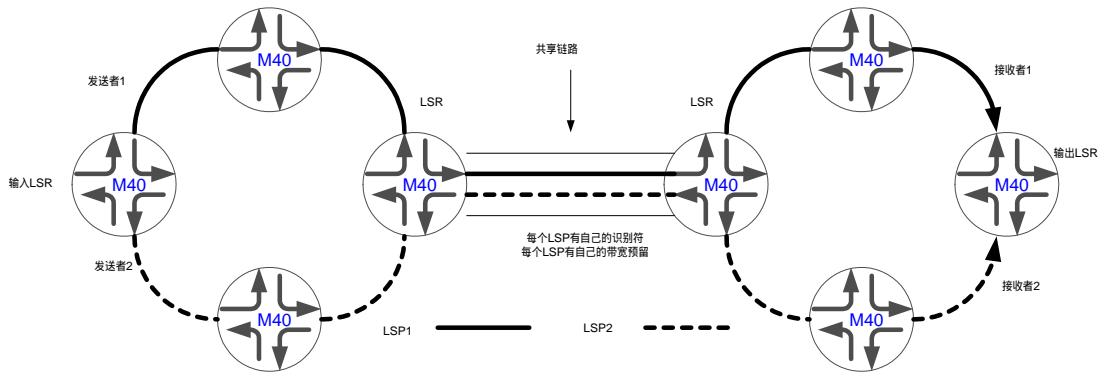
固定滤波器 (FF) 类型

固定滤波器 (FF) 预留类型定义了一个明确的发送者列表，并对每个发送者定义单独的预留。每个发送者具有一个单独的预留，不与其他发送者共享。每个发送者通过 IP 地址及一个本地标识号码 - - LSP_ID - - 被定义。因为每个发送者有其自己的预留，可以为每个发送者 - 接收者对建

立一个唯一的标记和一个单独的 LSP。对于传统的 RSVP 应用，FF 预留类型对于视频发送应用是非常理想的，在这种应用中，每个频道（或源）需要为每个不同的视频流提供单独的管道。

在扩展的 RSVP 中，不同的发送者这一术语可能是十分令人困惑，直到您意识到每个发送者和接收者指的是路由器上不同的接收者和发送者，而不是不同的终端系统（见图 11）。FF 预留类型允许建立多条，并行，单点传送，点对点 LSP。如果 LSP 穿过一条公用链路，这条共享链路路上的总的预留带宽是每个单独发送者所做预留的总和。FF 预留类型适用于并行发生并相互独立的来自不同发送者的业务。

图 11：固定滤波器（FF）预留类型



共享明确（SE）类型

共享明确（SE）预留类型在链路上建立一个单一预留，这一预留可被一系列明确列出的发送者共享。因为每个发送者都被明确地列在 RESV 消息中，不同的标记能够被分配到不同的发送者 - 接收者对，因此建立分离的 LSP。

图 12：共享明确（SE）预留类型

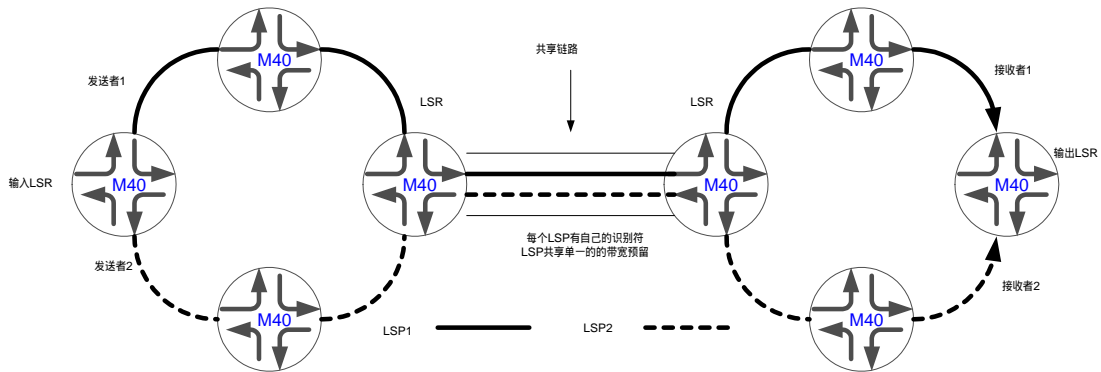


图 12 描述了 SE 预留类型在一条共享链路上的运行状况。每个 LSP 都有其穿过共享链路的自己的标识，但两条 LSP 共享了最大的带宽请求。如我们将在下一部分了解到的，SE 预留类型的适用是一个非常重要的工具，因为它允许 RSVP 信令在对一个已建立的 LSP 隧道进行重路由时完美地支持“中断前产生”。

LSP 隧道重路由过程

这一部分描述了 RSVP 如何在 LSP 被重路由时或在其被试图增加其可用带宽时建立一条维持资源预留（不双重计数）的 LSP。

建立初始 LSP 隧道

在初始的 PATH 消息中，输入 LSR：

- 组成一个 LSP_TUNNEL_IPV4 会话期对象，唯一地识别 LSP 隧道。LSP_TUNNEL_IPV4 会话期对象包含：
 - LSP 隧道双重节点的 IPV4 地址
 - 输入输出 LSR 间 LSP 隧道的 TUNNEL_ID，其将在此 LSP 整个生命周期中保持不变。
 - 识别隧道输入节点的 Extended_Tunnel_ID。（即，输入 LSR 的 IPV4 地址）

- 在 SESSION_ATTRIBUTE 对象中设置“输入节点可以重路由位”以要求输出 LSR 适用 SE 预留类型。
- 组成 SENDER_TEMPLATE 对象，其包括：
 - 发送者（输入）节点 IPV4 地址
 - LSP_ID，其可在将来被更改以允许输入 LSR 以不同的发送者出现，因此其可以在如果或当 LSP 需要被重路由时共享自己的资源（见 SENDER_TEMPLATE 对象和 FILTER_SPEC 对象中 LSP_TUNNEL_IPV4C 类扩展所包含的 LSP_ID 部分）。

接收到 PATH 消息之后，输出 LSR 向输入节点发送一个带有 SE 预留类型的 RESV 消息。

当输入 LSR 接收到 RESV 消息时，带有 SE 预留类型的初始 LSP 便被建立了。

建立重路由 LSP 隧道

当输入 LSR 想要对一条已有的 LSP 增加带宽或更改其路径时，它发送一个新的 PATH 消息。在重路由运行过程中，输入 LSR 必须以两个不同的发送者出现在 RSVP 会话期中。此种情况是通过在 SENDER_TEMPLATE 和 FILTER_SPEC 对象中包含新的 LSP_ID 实现的。

在新的 PATH 消息中，输入 LSR：

- 为新的 LSP 隧道建立一个 EXPLICIT_ROUTE 对象（ERO）
- 适用已有的 LSP_TUNNEL_IPV4 会话期对象识别将要被重路由的 LSP
- 选取新的 LSP_ID 并建立新的 SENDER_TEMPLATE。通过为 SENDER_TEMPLATE 选择新的 LSP_ID，输入 LSR 以不同的发送者出现在 RSVP 会话期中。

输入 LSR 向输出 LSR 发送新的 PATH 消息。但是，输入 LSR 继续使用旧的 LSP 隧道对业务进行转发并继续刷新原来的 PATH 消息。

输出 LSR 通过 RESV 消息对接收到的新 PATH 消息作出反应，RESV 消息包括一系列

RSVP 对象：

- 用来支持“按需上行”标记分配处理的 LABEL 对象
- SE 预留类型对象

在未被旧的和新的 LSP 隧道共享的链路上，新的 PATH/RESV 消息对被作为常规的 LSP 建立对待。但是，在被旧的和新的 LSP 隧道共同穿过的链路上，LSP_TUNNEL_IPV4 会话期对象及 SE 预留类型允许新的 LSP 隧道被建立，这样，它可以与旧的 LSP 隧道共享资源。这样便避免了在共享链路上的“双重计数”问题。在输入 LSR 接收到新 LSP 的 RESV 消息后，它可以开始使用新的 LSP 隧道转发业务。输入 LSR 应当为旧的 LSP 隧道发送一条 PATH_TEAR 消息以拆除中间 LSR 中就 LSP 隧道的状态。

增强 RSVP 扩展性，支持 Internet 核心网

一些 Internet 团体对在 Internet 核心网中使用 RSVP 作为信令协议的稳定性和扩展性表示关心。他们的关心主要表现在两个方面：

- LSR 的开销（业务量和 CPU 使用量），因为 RSVP 是一个“软状态”协议，不是“硬状态”协议 - - 在软状态协议中，RSVPPATH 及 RESV 消息必须在 LSP 隧道路径上的 LSR 中周期地进行刷新。如果刷新消息未被传输，LSP 状态将自动超时而最终被删除。对 RSVP 持批评态度的团体关心当 LSR 试图处理经常的更新消息时，LSR 将经历的开销。
- RSVP 信令的时延及稳定性，因为 RSVP 并未运行在一个稳定的传输上（即，TCP） - - 时延和稳定性问题可能在非刷新 RSVP 消息（PathErr，PathTear，ResvTear，或 ResvConf）在传输过程中丢失时出现。如果这类消息丢失，基于 LSR 刷新闻隔的 RSVP 信

令端对端响应将经历丢失。当响应时间由刷新闻隔限制时，建立或修改 LSP 的时间可能会超出特定应用所能接受的范围。这种关心可以通过降低刷新闻隔来解决。但是，它将明显增加业务量及处理开销，进而引起刷新开销的问题。

为增强 RSVP 信令的扩展性，时延，和稳定性，定义了一系列扩展：

- 捆绑消息扩展
- MESSAGE_ID 扩展
- 概要刷新扩展
- Hello 协议扩展

这些扩展解决了用户所关心的问题，而无需对刷新闻隔进行调整。它们有效地将 RSVP 从其传统的“软状态”模型转移到“固态”模型。在一个固态模型中，仍传输刷新消息，在支持稳定性的同时，业务量，CPU 使用量，及响应时延都被降低。没有任何被建议的扩展导致与传统 RSVP 实施不兼容的问题。

捆绑消息扩展

捆绑消息扩展降低了必须周期的发送及接收的 RSVP 消息总量。捆绑消息包含一个捆绑报头，接下来是一个包含了多种标准 RSVP 消息的主体部分（见图 13）。捆绑消息用于汇聚多条 RSVP 消息至一个单独的协议数据单元（PDU）内。

图 13：RSVP 捆绑消息



其它子消息

一条 RSVP 捆绑消息必须至少包含一个子消息。一个子消息可以包含除捆绑消息以外的其它 RSVP 消息。现在可用的子消息类型包括 PATH, PathErr, RESV, ResvErr, ResvConf, ACK, 或 Hello。ACK 及 Hello 消息将在下面讨论。

捆绑消息的地址被直接指向 RSVP 邻居, 并使用协议号 46 (即, RSVP) 作为“自然” IP 数据报被发送。捆绑报头直接遵循 IP 报头, 并没有中间传输报头。当 RSVP 路由器接收到一个并未指向其本地 IP 地址之一的捆绑消息时, 它将转发此消息。非 RSVP 路由器对待 RSVP 捆绑消息就象对待其它 IP 数据报一样。

支持 RSVP 捆绑消息是可选的。当捆绑消息帮助扩展 RSVP 和减少处理开销及带宽占用时, 节点并不需要使用捆绑消息传送每个 RSVP 消息。节点必须时刻准备接收及处理标准的 RSVP 消息。

MESSAGE_ID 扩展

两个新的对象作为 MESSAGE_ID 扩展的一部分被定义:

- MESSAGE_ID 对象
- MESSAGE_ID_ACK 对象

这些对象通过实施承认机制以支持 RSVP 消息的可靠发送。而且, 当刷新 (使用 PATH 及 RESV 消息) 被正常产生时, MESSAGE_ID 可被用于对消息何时表现一个新的状态提供标识。接收节点可使用这一信息减少其花费在处理刷新消息上的时间总量。

MESSAGE_ID 对象

MESSAGE_ID 对象通过允许接收者简单地识别一个包含未改变状态信息的信息来降低刷新消息的处理。当路由器传送一个包含 MESSAGE_ID 的刷新消息时，其将传送相同的 MESSAGE_ID 值，该值在最初广播状态被刷新时放置在 RSVP 消息中。当一个节点需要传送一个表达了新的或修改了的状态的消息时，MESSAGE_ID 的值被更改为一个比以前使用的值更大的值。如果一个 LSR 能够处理 MESSAGE_ID_ACK 对象，则在其传输包含 MESSAGE_ID 对象的刷新消息时，它可以选择设置 ACK_DESIRED 位。

当一个路由器接收到一条包含 MESSAGE_ID 的刷新消息时，它首先识别 RSVP 会话期，然后检验以前存储在它本地状态模块中的此 RSVP 会话期的值。如果在本地状态中不能发现有关 MESSAGE_ID 的值，则接收者必须对该消息进行完全的处理，因为它表示了一个新的或修改了的状态。但是，如果 PATH 或 RESV 消息中包含的 MESSAGE_ID 与同一会话期接收到的最近的消息所使用的相同，接收者则假设入站消息为一个状态刷新。

MESSAGE_ID_ACK 对象

传输 MESSAGE_ID_ACK 对象用于通知所接收的含有 MESSAGE_ID 对象的消息在发送时设置了 ACK_DESIRED 位。ACK 消息承载了一个或多个 MESSAGE_ID_ACK 对象，但可能不包含任何 MESSAGE_ID 对象。这种机制可以确保在网络丢失时对错误及确定消息可靠地传输，并支持快速刷新。

摘要刷新扩展

摘要刷新扩展支持在无需传输传统的 PATH 和 RESV 消息的情况下沿 LSP 隧道刷新 RSVP 状态。摘要刷新扩展的主要好处是它减少了为维持 RSVP 状态同步所需传输及处理的信息量。

摘要刷新消息承载了一系列 MESSAGE_ID 对象用于识别需要被刷新的 PATH 及 RESV 状态。当一个 RSVP 节点接收到一条摘要刷新消息时，它使用本地安装的 PATH 或 RESV 状态与每个接收到 MESSAGE_ID 对象进行匹配。如果 MESSAGE_ID 对象与带宽状态匹配，状态就象接收到一个标准的 RSVP 刷新消息那样被更新。但是，如果一个 MESSAGE_ID 对象未能与接收者本地状态匹配，接收者将发送一个刷新 NACK 来通知摘要刷新消息的发送者。一个刷新 NACK 通过在 MESSAGE_ID_ACK 对象中对 REFRESH_NAK 位进行设置来指出。

当使用摘要刷新消息更新 RSVP 会话期时，常规的刷新消息传输应被抑制。摘要刷新扩展不能用于包含与以前广播状态有改变的 PATH 或 RESV 消息。同时，只有那些在以前广播时包含哟 MESSAGE_ID 对象的 PATH 或 RESV 消息才能通过摘要刷新消息进行刷新。

Hello 协议扩展

Hello 协议用于检测邻居节点的丢失或重新设置邻居的 RSVP 状态信息。在标准的 RSVP 中，邻居监测作为 RSVP 软状态模型的一部分出现。预留状态向缓冲信息一样被维持，最初被安装，然后被输入和输出 LSR 周期性地刷新。如果状态未能在一个特定的时间间隔内被刷新，LSR 将丢弃这个状态，因为它假设或者邻居节点丢失，或者它的 RSVP 状态信息被重新设置。

Hello 协议扩展由 Hello 消息，HELLOREQUEST 对象和 HELLOACK 对象组成。两个邻居间的 Hello 处理指出对故障检测间隔的独立选择。每个邻居可以自动发出 HELLOREQUEST 对象。每个 HELLOREQUEST 对象通过 HELLOACK 对象来回答。

结论

多年来，Juniper 网络公司一直积极地参与 IETF 及和世界最大的 IP 网络运行商一起设计 MPLS/流量工程应用的信令协议。这些服务提供商正是那些少数真正了解如何建设并管理大型 IP 基

基础结构，凭借他们在公共 Internet 上多年的实际经验。在通过 IETF 严格的测试及争论后，世界上最大和最具经验的 ISP 们确信扩展的 RSVP 已经足够成熟，可以在他们现在的实施计划中扮演重要的角色。

扩展的 RSVP 提供 MPLS/流量工程应用信令所需支持的所有功能。它允许建立明确路由的 LSP，提供标记分配功能，在 LSP 中支持资源预留或服务等级，并提供 LSP 穿过的物理路由信息。扩展的 RSVP 在对已有 LSP 进行重路由时可完美地支持“中断前产生”的概念，在建立一条新的 LSP 时允许对以前分配的网络资源进行重新分配，并在 LSP 建立时提供循环防止/检测。最后扩展的 RSVP 解决了由于传统 RSVP 的“软状态”模型所引出的扩展性，时延，及稳定性等问题。

参考

Internet Drafts

Awduche, D., J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, *Requirements for Traffic Engineering over MPLS*, draft-ietf-mpls-traffic-eng-01.txt, June 1999.

Awduche, D., L. Berger, D-H Gan, T. Li, G. Swallow, and V. Srinivasan, *Extensions to RSVP for LSP Tunnels*, draft-ietf-mpls-rsvp-lsp-tunnel-02.txt, March 1999.

Callon, R., A. Viswanathan, and E. Rosen, *Multiprotocol Label Switching Architecture*, draft-ietf-mpls-arch-05.txt, April 1999.

Callon, R., G. Swallow, N. Feldman, A. Viswanathan, P. Doolan, and A. Fredette, *A Framework for Multiprotocol Label Switching*, draft-ietf-mpls-framework-03.txt, June 1999.

Yuhara, M and M. Tomikawa, *RSVP Extensions for ID-based Refreshes*, draft-yuhara-rsvp-refresh-00.txt, April 1999.

Request for Comments

RFC 2205, *Resource ReserVation Protocol—Version 1 Functional Specification*, R. Braden ED., L. Zhang, S. Berson, S. Herzog, and S. Jamin, September 1997.

RFC 2207, *RSVP Extensions for IPSEC Ipv4 Data Flows*, L. Berger and T. O'Malley, September 1997.

RFC 2208, *Resource ReSerVation Protocol (RSVP) Version 1 Applicability Statement – Some Guidelines on Deployment*, A. Mankin, Ed., F. Backer, B. Braden, M. O'Dell, A. Romanow, A. Weinrib, L. Zhang, September 1997.

RFC 2209, *Resource ReSerVation Protocol (RSVP) – Version 1 Message Processing Rules*, R. Braden and L. Zhang, September 1997.

Textbooks

Davie, B., P. Doolan, and Y. Rekhter, *Switching in IP Networks: IP Switching, Tag Switching, and Related Technologies*, Morgan Kaufmann, 1998 (ISBN 1-55860-505-3).

Metz, Christopher, *IP Switching: Protocols and Architectures*, McGraw-Hill, New York, 1999 (ISBN 0-07-041953-1).

Other References

Zhang, L., S. Deering, D. Estrin, S. Shenker, and D. Zappala, *RSVP: A New Resource ReSerVation Protocol*, IEEE Network, September 1993.