ICS 33.180.01
M 33

中华人民共和国国家标准

GB/T 21645.5—2012

自动交换光网络(ASON)技术要求 第5部分:用户-网络接口(UNI)

Technical requirements for automatically switched optical network—Part 5: User-Network Interface(UNI)

2012-06-29 发布　　2012-10-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

中华人民共和国
国家标准
自动交换光网络(ASON)技术要求
第5部分:用户-网络接口(UNI)
GB/T 21645.5—2012
*
中国标准出版社出版发行
北京市朝阳区和平里西街甲2号(100013)
北京市西城区三里河北街16号(100045)
网址 www.spc.net.cn
总编室:(010)64275323 发行中心:(010)51780235
读者服务部:(010)68523946
中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销
*
开本 880×1230 1/16 印张 5.25 字数 153千字
2012年11月第一版 2012年11月第一次印刷
*
书号:155066·1-45708

目　　次

前　言

GB/T 21645《自动交换光网络(ASON)技术要求》分为以下几个部分：

——自动交换光网络(ASON)技术要求　第1部分：体系结构与总体要求

——自动交换光网络(ASON)技术要求　第2部分：术语和定义

——自动交换光网络(ASON)技术要求　第3部分：数据通信网(DCN)

——自动交换光网络(ASON)技术要求　第4部分：信令技术

——自动交换光网络(ASON)技术要求　第5部分：用户-网络接口(UNI)

——自动交换光网络(ASON)技术要求　第6部分：管理平面

——自动交换光网络(ASON)技术要求　第7部分：自动发现

——自动交换光网络(ASON)技术要求　第8部分：路由

——自动交换光网络(ASON)技术要求　第9部分：外部网络-网络接口(E-NNI)

本部分是GB/T 21645的第5部分。

本部分按照GB/T 1.1—2009给出的规则起草。

本部分主要参考了OIF UNI1.0 R2《UNI1.0信令规范版本2》和IETF GMPLS系列协议。

本部分由中华人民共和国工业和信息化部提出。

本部分由中国通信标准化协会归口。

本部分起草单位：华为技术有限公司、工业和信息化部电信研究院、上海贝尔股份有限公司、中兴通讯股份有限公司、武汉邮电科学研究院、中国联合网络通信有限公司。

本部分主要起草人：高建华、张国颖、徐云斌、孙俊、黄峰、柯明、宋然、王健全。

自动交换光网络(ASON)技术要求 第5部分:用户-网络接口(UNI)

1 范围

GB/T 21645的本部分规定了自动交换光网络(ASON)用户-网络接口(UNI)技术要求,包括:ASON UNI接口所支持的连接业务类型、用来调用UNI接口服务的信令协议以及支持UNI信令协议的辅助功能。

本部分适用于基于同步数字体系(SDH)和光传送网(OTN)的ASON。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

GB/T 15837—2008 数字同步网接口要求

GB/T 21645.1—2008 自动交换光网络(ASON)技术要求 第1部分:体系结构与总体要求

GB/T 21645.3—2009 自动交换光网络(ASON)技术要求 第3部分:数据通信网(DCN)

GB/T 21645.4—2010 自动交换光网络(ASON)技术要求 第4部分:信令技术

YD/T 1462—2006 光传送网(OTN)接口

IETF RFC2961 RSVP对减少刷新开销的扩展(RSVP Refresh Overhead Reduction Extensions)

IETF RFC3473 GMPLS RSVP-TE信令扩展(GMPLS Signaling RSVP-TE Extensions)

IETF RFC3474 GMPLS RSVP-TE扩展支持ASON及应用(GMPLS RSVP-TE Usage and Extensions for ASON)

IETF RFC3477 RSVP-TE对无编号链路的支持(Signalling Unnumbered Links in RSVP-TE)

IETF RFC3945 GMPLS架构(GMPLS Architecture)

IETF RFC4003 GMPLS信令支持出口标签控制(GMPLS Signaling Procedure for Egress Control)

IETF RFC4204 链路管理协议(Link Management Protocol)

IETF RFC4206 GMPLS对层次LSP的支持(LSP Hierarchy with GMPLS TE)

IETF RFC4208 RSVP对Overlay模型扩展(RSVP-TE Support for the Overlay Model)

IETF RFC4328 GMPLS支持对G.709 OTN网络控制的扩展(GMPLS Signaling Extensions for G.709)

IETF RFC4606 GMPLS支持对SDH网络控制的扩展(GMPLS Extensions for SONET & SDH Control)

IETF RFC4974 GMPLS RSVP-TE信令支持呼叫的扩展(GMPLS RSVP-TE Signaling Extensions in Support of Calls)

IETF RFC4990 GMPLS网络地址的使用(Use of Addresses in GMPLS Networks)

IETF RFC5150 GMPLS对拼接LSP的支持(LSP Stitching with GMPLS TE)

IETF [eth-tspec] 以太网流量参数(Ethernet Traffic Parameters)

IETF [vcat-lcas] GMPLS支持VCAT & LCAS的扩展(Operating VCAT and LCAS with

GMPLS)

IETF [gmpls-esvcs] GMPLS 支持 MEF&G.8011 以太业务交换(GMPLS Support for MEF and G.8011)

OIF-UNI-1.0-R2-Common 用户-网络接口 1.0 公共部分规范(版本 2)(UNI 1.0 Signaling Specification, Release 2: Common Part)

OIF-UNI-2.0-Common 用户-网络接口 2.0 公共部分规范(UNI 2.0 Signaling Specification: Common Part)

OIF-SEP-1.1 UNI/NNI 安全扩展规范(Security Extension for UNI and NNI)

3 术语和定义、缩略语

3.1 术语和定义

下列术语和定义适用于本文件。

3.1.1

客户层 client layer

相对于提供传送服务的服务层,作为一个传送网服务的客户网络层次。如,IP 层是传送网络的客户层。

3.1.2

客户层地址 client layer address

客户层协议所使用的地址。如,连接到传送网络的 IP 客户层设备所使用的 IP 地址。

3.1.3

纤内信令 in-fiber signaling

纤内信令是指通过和数据绑定的嵌入式通信通道来传递信令消息的信令方式。

3.1.4

IP 控制通道 ip control channel

两个设备之间用来传送 IP 包的通信通道。

3.1.5

纤外信令 out-of-fiber signaling

纤外信令是指通过独立的通信链路来传递信令消息的信令方式,其中通信链路和相关的数据链路分离。

3.1.6

端口 port

用户或光网络网元内部用来在网元之间终结双向链路的硬件接口。如 TNE 的 STM-16 端口。

3.1.7

服务层路径或踪迹 service path or trail

路径是一个传送实体,包括一对相互关联的单向路径,能够同时向两个相反的方向传送信息。单向路径负责在一个路径终端源的输入端和一个路径终端宿的输出端之间传送信息,同时监视被传送信息的完整性。单向路径由路径终端功能和一个网络连接组成。

3.1.8

传送网络分配的地址 transport network assigned address;TNA

TNA 地址用于标识 UNI 连接终端,由传送网络分配。每一个 TNA 地址是全局唯一的地址,地址类型可以是 IPv4、IPv6 或 NSAP。

3.1.9

传送网络地址　transport network address

传送网络内部实体(如 TNE)的地址。

3.1.10

传送网元　transport network element;TNE

具有光接口的网元(在传送网络内部),如,光分插复用设备 ADM。

3.1.11

UNI 信令通道　uni signaling channel

UNI-C 和 UNI-N 之间的逻辑通信通道,通信通道用来传送 UNI 信令消息。

3.1.12

用户或客户　user or client

连接到传送网络来使用光传送服务的网络设备。客户设备包括 IP 路由器、ATM 交换机、以太交换机、SDH 交叉连接设备等。

3.2　缩略语

下列缩略语适用于本文件。

ACK:应答(Acknowledgement)

ASON:自动交换光网络(Automatically Switched Optical Network)

ATM:异步传输模式(Asynchronous Transfer Mode)

BGP:边界网关协议(Border Gateway Protocol)

CC:控制通道(Control Channel)

CCID:控制通道标识(Control Channel ID)

DCC:数据通信通路(Data Communications Channel)

DCN:数据通信网络(Data Communications Network)

DS:分集(Diversity)

EPL:以太网专线(Ethernet Private Line)

ESP:封装安全有效载荷(Encapsulating Security Payload)

EVPL:以太网虚拟专线(Ethernet Virtual Private Line)

FA:转发邻接(Forwarding Adjacency)

FF:固定过滤(Fixed Filter)

GCC:通用通信通道(General Communication Channel)

GFP:通用成帧规程(Generic Framing Procedure)

GMPLS:通用多协议标记交换(Generalized Multi-Protocol Label Switching)

G-PID:通用静荷标识(Generalized Payload ID)

GSMP:通用交换管理协议(Generic Switch Management Protocol)

HMAC:基于哈希的消息验证代码(Hash Message Authentication Code)

NCC:连续级联成员数量(Number of Contiguous Components)

NNI:网络-网络接口(Network-to-Network Interface)

NSAP:网络服务接入点(Network Service Access Point)

NVC:虚级联成员数量(Number of Virtual Components)

IETF:因特网工程任务组(Internet Engineering Task Force)

IPCC:IP 控制通路(IP Control Channel)

ISI:内部信令接口(Internal Signaling Interface)

ITU：国际电信同盟(International Telecommunications Union)
LAP-D：D 信道上的链路访问规程(Link Access Procedure，D-Channel)
LAPS：SDH 上的链路访问规程(Link Access Procedure-SDH)
LCAS：链路容量调整方案(Link Capacity Adjustment Scheme)
LMP：链路管理协议(Link Management Protocol)
LSP：标签交换路径(Label Switched Path)
MEF：城域以太网论坛(Metro Ethernet Forum)
MS：复用段(Multiplex Section)
MT：倍数(Multiplier)
MTU：最大传输单元(Maximum Transmission Unit)
OAM：操作维护管理(Operations，Administration，and Management)
OCh：光通道(Optical Channel)
ODU：光通道数据单元(Optical Channel Data Unit)
OIF：光互联网论坛(Optical Internetworking Forum)
OTN：光传送网络(Optical Transport Network)
PDP：策略决定点(Policy Decision Point)
PDU：协议数据单元(Protocol Data Unit)
PEP：策略执行点(Policy Enforcement Point)
PK：公开密钥基础设施(Public Key Infrastructure)
PPP：点到点协议(Point to Point Protocol)
RCC：连续级联请求(Requested Contiguous Concatenation)
RSVP：资源预留协议(Resource ReSerVation Protocol)
RSVP-TE：支持流量工程的资源预留协议(RSVP-Traffic Engineering)
SCN：信令通信网(Signaling Communication Network)
SDH：同步数字体系(Synchronous Digital Hierarchy)
SE：共享显式(Shared Explicit)
SHA：安全散列算法(Secure Hash Algorithm)
SLA：服务等级协议(Service Level-Agreement)
SPC：软永久连接(Soft Permanent Connection)
SRLG：共享风险链路组(Shared Risk Link Group)
STM：同步传送模块(Synchronous Transport Module)
TDM：时分复用(Time Division Multiplexing)
TLV：类型-长度-值(Type Length Value)
TNA：传送网分配地址(Transport Network Assigned Address)
TNE：传送网设备(Transport Network Element)
UNI：用户-网络接口(User-Network Interface)
UNI-C：UNI 客户侧(User-Network Interface-Client)
UNI-N：UNI 网络侧(User-Network Interface-Network)
VC：虚容器(Virtual Container)
VCAT：虚级联(Virtual Concatenation)
VCG：虚级联组(Virtual Concatenation Group)

4 UNI 接口功能定义

4.1 UNI 提供的功能

传送网络通过 UNI 接口所提供的主要功能是按需实现客户连接业务的建立请求和删除请求处理。

UNI 所支持的业务连接是指在传送网络的入口和出口接入点之间具有固定带宽的电路连接，并且是具有特定帧格式的单向或双向连接。UNI 接口所提供的连接类型包括：SDH 连接、OTN 连接、以太网连接。UNI 接口所提供的功能包括：

a) 基于 RSVP-TE 信令的连接业务创建、删除和修改，其中连接业务修改为可选支持。支持的连接业务包括：

1) SDH 连接业务，其中 VC4 连接为必选支持，VC12、VC3、连续级联 VC-4-Xc，(X = 4，16，64，256)、虚级联 VC-3/4-Xv，(X = 1 ... 256)为可选支持；

2) 以太网连接业务，其中支持的以太网物理接口可以是：10 Mbit/s(10Base)、100 Mbit/s(100Base)、1 Gbit/s(1 000Base)、10 Gbit/s(10Gbase-R)；

3) OTN 连接业务，其中 ODU_k(k=1、2)、OCh 连接为必选支持，ODU_k(k=0、3、4)以及级联业务为可选支持。

b) 控制通道维护；

c) 基于 RSVP-TE 信令的呼叫控制(可选)；

d) 基于 LMP 的邻居发现(可选)；

e) UNI 策略和安全控制(可选)。

4.2 UNI 支持的客户业务

4.2.1 UNI 支持的 SDH 业务

UNI 允许在 SDH 网络以及其他服务层网络(ITU-T G.805 中定义的客户/服务层网络)中承载 SDH 业务。业务模型如图 1 所示。

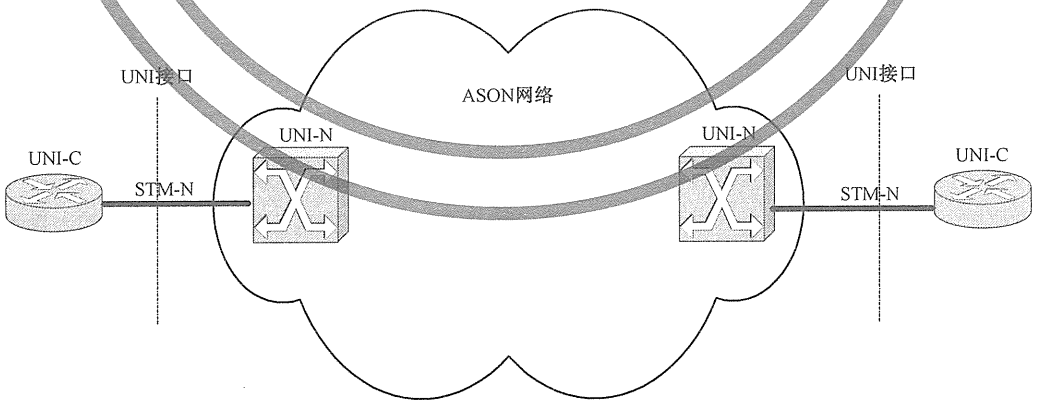


图 1 UNI 支持 SDH 业务传送的模型

UNI-C 和 UNI-N 之间的接口链路为 SDH 链路，物理接口速率可以是：STM-1、STM-4、STM-16、STM-64 或 STM-256。

UNI 接口支持的 SDH 业务连接包含以下几种标准的信号类型：

a) VC-4；

b) VC-12、VC-3(可选)。

UNI 接口支持的业务连接还可以包含以下几种 SDH 级联信号类型:

a) 连续级联信号:VC-4-Xc,(X = 4,16,64,256)(可选);

b) 虚级联:VC-3/4-Xv(X = 1 ... 256)(可选)。

UNI 接口支持的帧结构:支持 ITU-T G.707 中定义的标准 SDH 帧结构信号;

UNI 接口对 SDH 业务信号透明传送的支持:支持通道开销透明,即传送网络不能修改客户信号的通道开销字节(缺省)。

表 1 总结了 UNI 所支持的 SDH 业务连接类型:

表 1 UNI 接口所支持的 SDH 业务连接类型

SDH 信号类型	透明级别	级联类型	备注
VC-12	通道	非级联	VC-12(仅 VC 透明)
VC-3	通道	非级联	VC-3(仅 VC 透明)
	通道	虚级联	VC-3-Xv(仅 VC 透明)
VC-4	通道	非级联	VC-4(仅 VC 透明)
	通道	连续级联	VC-4-Xc(仅 VC 透明)
	通道	虚级联	VC-4-Xv(仅 VC 透明)

4.2.2 UNI 支持的以太网业务

UNI 允许在 SDH 网络以及其他服务层网络(ITU-T G.805 中定义的客户/服务层网络)中承载以太网数据业务。业务模型如图 2 所示。

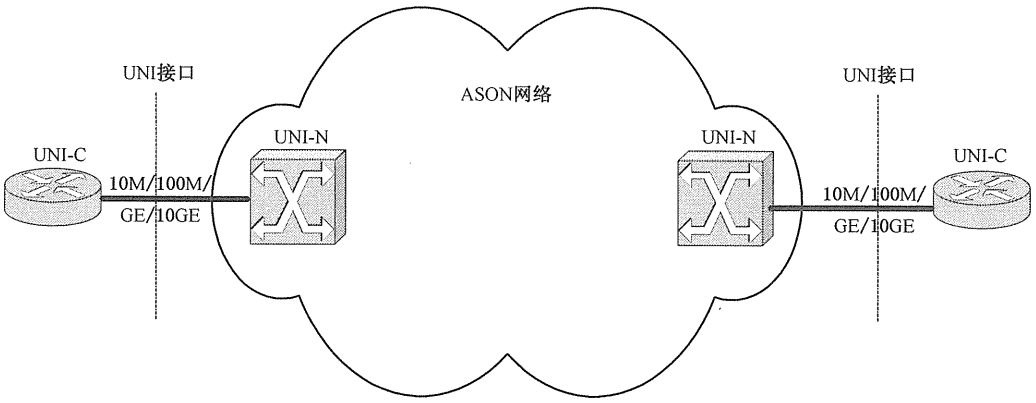


图 2 UNI 支持以太网业务传送的模型

UNI-C 和 UNI-N 之间的接口链路为以太网链路,物理接口速率可以是:10 Mbps、100 Mbps、1 Gbps、或 10 Gbps。

UNI 接口支持的以太网业务连接为以太网信号类型。网络连接范围为点到点的以太网专用线业务(EPL)和以太网虚拟专用线业务(EVPL)。

表 2 总结了 UNI 所支持的以太网物理接口类型:

表 2　UNI 接口所支持的以太网物理接口类型

以太网接口类型	备注
10 Mbit/s	10Base
100 Mbit/s	100Base
1 Gbit/s	1000Base
10 Gbit/s	10Gbase-R

4.2.3　UNI 支持的 OTN 业务

UNI 允许在 ITU-T G.709 所定义的 OTN 网络以及其他服务层网络(ITU-T G.805 中定义的客户/服务层网络)中承载 OTN 业务。业务模型如图 3 所示。

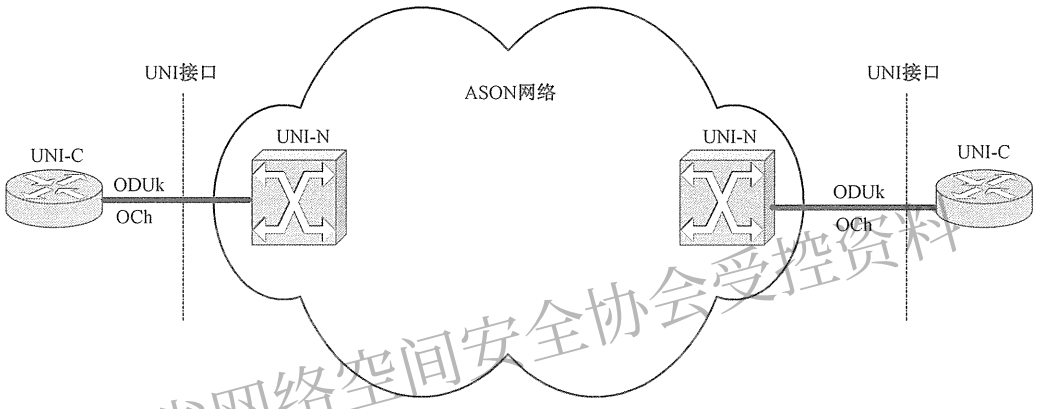


图 3　UNI 支持 OTN 业务传送的模型

UNI-C 和 UNI-N 之间的接口链路为 ITU-T G.709 中所定义的标准的 OTN 接口链路。

尽管 ITU-T G.709 定义了多个网络层次,但仅将 ODUk 和 OCh 定义成了交换层。因此,UNI 连接类型将支持 ODUk 和 OCh 层的连接业务。

基于 ITU-T G.709 的 UNI 接口信令传送配置可以使用纤外信令配置、也可以使用非关联的 OSC 开销来传送信令、也可以使用带内的基于 OCh 关联的开销配置来传送信令信息。

表 3 总结了 UNI 所支持的 OTN 业务连接类型:

表 3　UNI 支持 OTN 业务连接的类型

G.709 OTN 信号类型	级联类型	备注
ODU1	非级联	
	虚级联	可选
ODU2	非级联	
	虚级联	可选
ODU3	非级联	可选
	虚级联	可选
OCh	无	

4.2.4 UNI 业务的信令行为

UNI 信令指 UNI-C 和 UNI-N 实体之间的消息交换，信令行为用来激活传送网络服务。UNI 接口支持如下的信令行为：

a） 业务创建

该行为根据客户连接请求所指定的属性在两个接入点之间创建一条业务。业务创建应该满足网络所定义的策略（如用户组连接限制）和安全处理流程。业务的创建过程可以包含呼叫和连接的建立过程，也可以只包含连接的建立过程。

b） 业务删除

该行为删除一条存在的业务。业务的删除过程可以包含呼叫和连接的删除过程，也可以只包含连接的删除过程。

c） 业务状态查询

该行为用来查询某个呼叫或连接参数的状态。

d） 业务修改（可选）

该行为用来对某个呼叫或连接的参数进行修改。

4.3 UNI 业务的支撑过程

4.3.1 概述

本部分定义了支撑 UNI 信令行为的其他支撑处理过程。这些处理过程中，邻居发现处理为可选的处理，控制通道维护为必选功能。UNI 完成 4.2.4 的信令行为时，可以不包含下面这些可选的处理过程。这种情形下，下面这些可选的处理流程所提供的信息应由手动配置完成。

4.3.2 UNI 邻居发现

邻居发现功能为可选功能。邻居发现处理可以在客户和传送网络设备之间动态建立接口映射关系。还可以验证传送网络设备和客户设备之间的端口连通性和一致性。也允许开始启用并维护 UNI 信令控制通道。对不同信令参考配置的邻居发现处理见 6.1.4 的描述。

4.3.3 控制通道维护

控制通道维护功能为必选功能。UNI 信令交互需要在客户侧和网络侧信令实体之间存在控制通道。如 5 章中的描述，可以存在不同的控制通道配置。UNI 支持在所有这些配置的情形下对控制通道进行维护，如 6.1.1 中的描述。另外，控制通道维护功能可以是基于 LMP 协议的控制通道维护功能来实现，也可以通过基于 RSVP-TE 协议的 Hello 消息功能来实现。

4.4 UNI 业务调用的参考配置

4.4.1 概述

本条所描述的参考配置指定了调用传送网络服务的不同方法。共有两种服务调用模型，一种为直接调用，另外一种为间接调用。这两种模式下，客户侧和网络侧 UNI 信令代理分别为 UNI-C 和 UNI-N。直接调用模式下，客户直接调用传送网络的服务。UNI-C 功能和客户设备本身绑定在一起。间接调用模式下，一个名为代理的 UNI-C 执行 UNI 功能，并负责一个或多个客户设备。客户设备不需要和代理 UNI-C 驻留在一起。本节中描述了四种基本的配置，其中直接模式有两种配置，间接模式有两种配置。其他配置可以是这四种基本配置的不同组合。对于每种配置方式，UNI-C 和 UNI-N 之间通过 IP 控制通道来对信令消息进行传送。UNI 信令协议是采用 RSVP-TE 的扩展协议。

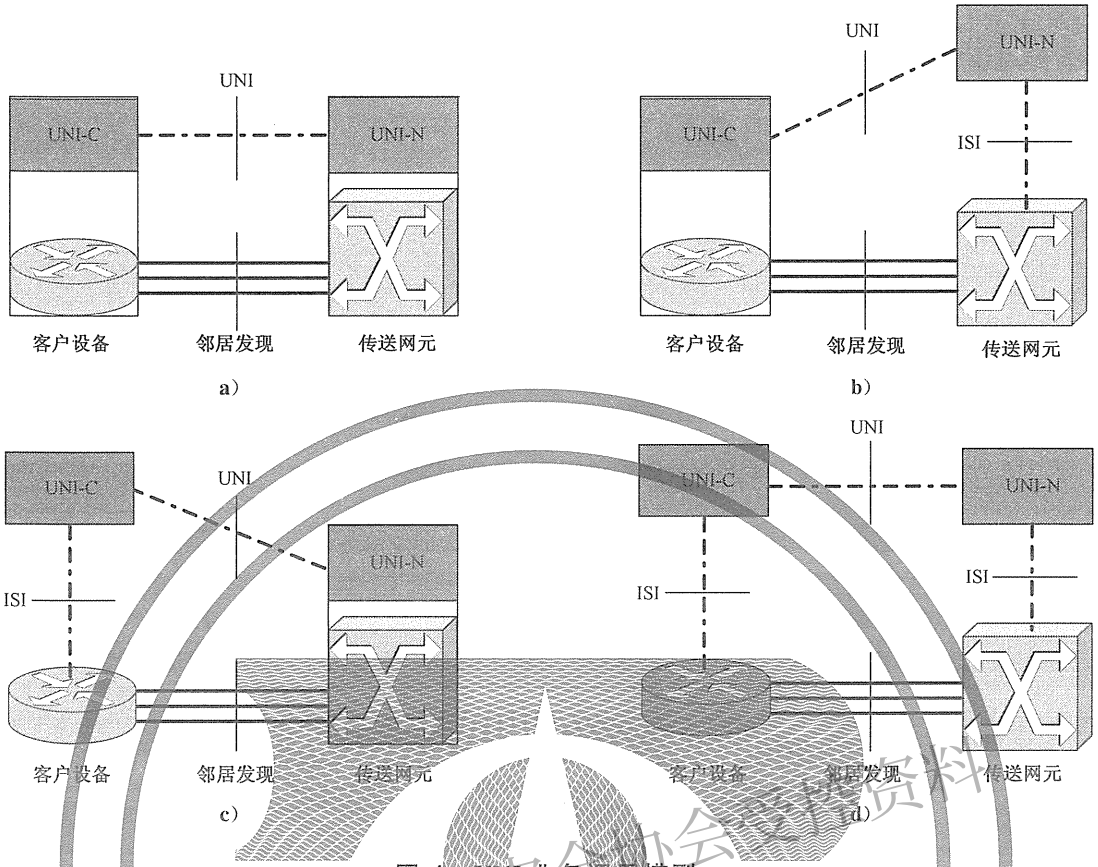


图 4 UNI 业务调用模型

4.4.2 直接调用模型

这种模式下，UNI-C 功能由客户设备自身来实现。有两种基本的配置，如图 4 中的 a)、b)所示。这两种配置的不同之处如下：第一种配置中(图 4 中的 a))，UNI-N 功能由传送网络网元设备自身来实现。而第二种配置中(图 4 中的 b))UNI-N 功能由一个代理来实现。当 UNI-N 功能由一个代理来实现时，对代理的驻留位置没有限制。传送网络的内部信令接口(ISI)用来承载 UNI-N 和 TNE 之间的信令消息。内部接口和 UNI 信令规范无关。

在直接调用模式下，UNI 邻居发现功能可以通过客户和 TNE 之间的每个数据接口来完成。如第 5 章中的描述，UNI-C 和 UNI-N 之间的 IP 控制通道可以是纤内的(In-Fiber)，也可以是纤外的(Out-of-Fiber)。客户调用传送网络的服务可以通过客户网络的管理系统来发起，也可以通过客户自身所定义的流量工程机制来发起。在传送网络内部，来自 UNI 接口的服务请求可以使用集中系统来提供，也可以通过分布式协议来完成。

4.4.3 间接调用模型

这种模型如上图 4 中的 c)、d)所示，客户使用信令代理来调用传送网络服务。代理 UNI-C 实现 UNI 信令功能，该代理 UNI-C 负责一个或多个客户设备的 UNI 信令功能。客户设备可以不需要和代理 UNI-C 驻留在一起。包括：

a) 客户可以不实现邻居自动发现处理。这种情形下，UNI-N 和代理 UNI-C 应手动配置邻接点信息；

b) 代理 UNI-C 执行它所负责的客户设备的 UNI 功能；

c) 代理 UNI-C 和它所代表的客户之间可以使用私有的或标准的内部信令接口-内部信令接口(ISI)来进行通信。ISI 接口信令(如 GSMP)不是 UNI 接口规范的研究范围。

代理 UNI-C 和 UNI-N 之间运行 RSVP-TE 信令协议。代理 UNI-C 和 UNI-N 之间的 IP 控制通道为纤外信令配置方式，如第 5 章中的描述。UNI-N 实体应知道被客户设备授权的代理 UNI-C 的相关信息。

每个代表客户的代理 UNI-C 应配置的参数包括：

a) 客户层地址，可以是第 7 章中所描述的任意一种；

b) 分配给客户设备的 TNA 地址；

c) 客户设备连接传送网络设备的端口 ID 和节点 ID(如果有邻居自动发现功能，不用实现该配置)；

d) 信令控制通道：纤外 IPCC 对应的地址，该地址用来发送信令消息；

e) 客户设备的服务能力(如 SDH 能力)。

代理 UNI-C 可以支持多个客户设备。并且，光纤连接的两个端点可以由下面不同的组合来处理：

a) 代理 UNI-C 和 UNI-C；

b) 两个分离的代理 UNI-C，端点中的一个；

c) 相同的代理 UNI-C 同时处理两个端节点。

这种模型下，有两种基本的配置，如图 4 中的 c)、d)。图 4 中的 c)显示 UNI-N 的功能实现限制在 TNE 中来实现。图 4 中的 d)显示在 TNE 设备之外实现 UNI-N 的功能，并通过内部信令接口 ISI 来实现 UNI-N 和 TNE 之间的通信。

4.4.4 服务调用配置

本部分允许在同一个网络使用多个 UNI 业务调用模式。如：一些客户使用直接调用模式，另外一些客户使用间接调用模式；类似地，在网络侧允许有些 TNE 有 UNI 信令功能，另外的 TNE 通过代理来实现 UNI 信令功能；最后，每个连接终端的客户允许使用不同的业务调用模式。

4.5 无中断业务的参数修改(可选)

4.5.1 带宽修改

4.5.1.1 带宽修改概述

有几种方式可支持无中断业务参数修改，这些方式都涉及到对呼叫的修改。对于呼叫的修改可通过增加或删除呼叫中的连接或修改呼叫中已有连接的参数实现。存在两种业务架构的带宽修改：一种是对以太网业务的呼叫带宽修改，一种是对 SDH 或 OTN 业务的呼叫带宽修改。对于以太网业务的带宽修改，不需进行适配方式的修改。然而对于 SDH 或 OTN 业务的带宽修改可能需要修改适配方式。本部分定义的呼叫带宽修改不涉及到对适配能力的修改。

4.5.1.2 带宽修改原则

带宽修改可由 UNI-C 发起，发起的带宽修改要求不能涉及到对适配能力的修改。UNI-C 可进行如下方式的带宽修改：

a) 直接修改连接速率，只对以太网业务有效；

b) 通过修改呼叫中已有连接的 Multiplier 域来修改连接带宽；

c) 在已有呼叫中增加连接；

d) 删除已有呼叫中的连接。

4.5.1.3 SDH 业务

当 UNI 传送 SDH 业务时，有以下数种带宽修改方式：

a) VCn 改变。由于 VCn 的改变涉及到适配方式的改变，这种方式不被允许，不能从 VC3 改变为 VC4，如果要进行改变，则认为是业务的改变，需要新的呼叫支持。

b) VCn-xc 改变。不允许 x 改变，由于这种改变意味着适配方式的改变，例如从 VC4-4c 改变为 VC4-16c。如同 a)所述，对于 n 的改变不被允许。

c) m * VCn 的改变。允许 m 改变，例如从 1 * VC4 改变为 5 * VC4，可通过在呼叫中增加/删除连接实现，并不涉及到对适配方式的修改。如同 a)所述，对于 n 的改变不被允许。

d) m * VCn-xc 的改变。允许 m 改变，例如从 1 * VC4-4c 改变为 3 * VC4-4c，对 n，x 的改变不被允许。

图 5 描述了通过增加呼叫中的连接数增加呼叫带宽。例如一个呼叫中已存在一个 VC3 的连接，UNI-C 和 UNI-N 间的承载链路为 STM-16，可通过增加一个 VC3 的连接增加呼叫的带宽。该 VC3 可用 UNI-C 和 UNI-N 间的任何一条承载链路。客户侧可将两条连接放入一个 VCG 组，这对 UNI 是不可见的。

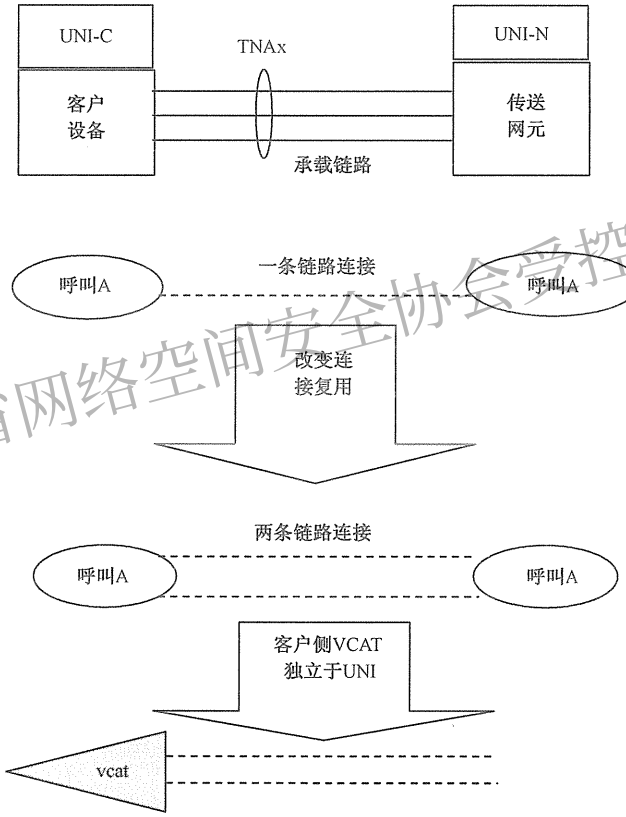


图 5 SDH 业务带宽修改

4.5.1.4 OTN 接口

对于 OTN 客户侧而言，连接的数目可通过 OTN 业务参数属性中的倍增指示(MT)字段进行修改。

4.5.1.5 以太网业务

以太网业务在信令中支持以 1 Mbit/s 为增减粒度的带宽编码。修改以太网呼叫的带宽，有两种方式：增删以太网呼叫中已有的连接，直接修改已有连接的带宽值。如图 6 所示：

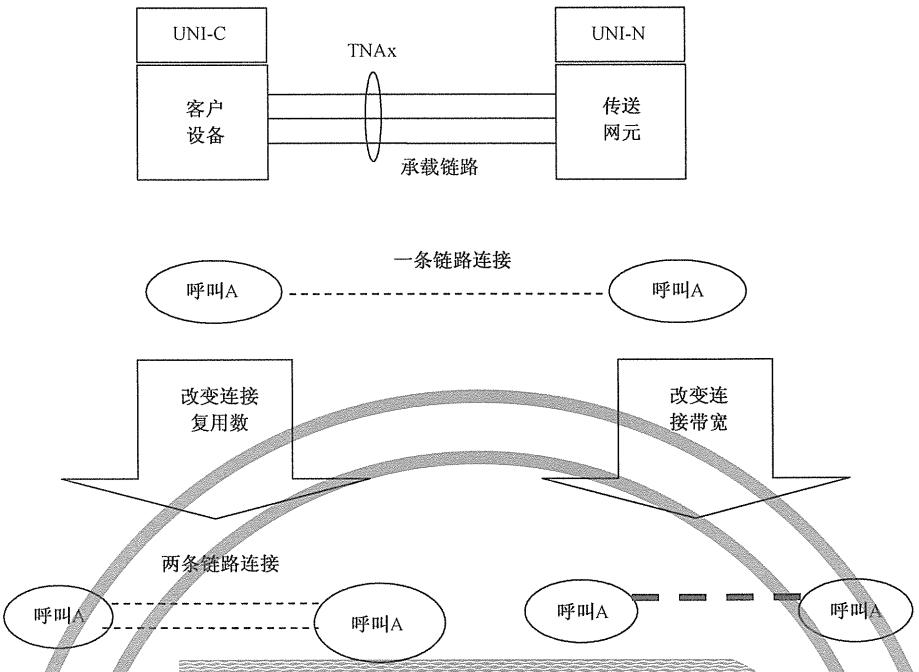


图 6　以太网业务带宽修改

5　UNI 信令传送配置

5.1　UNI 信令传送配置概述

按照 4.4 所描述的 UNI 业务调用参考配置，UNI-C 和 UNI-N 之间需要控制通道来传送信令消息。本部分支持如下的信令传送配置类型：

a)　纤内(In-Fiber)

承载信令消息的通信通道嵌入在客户和 TNE 之间承载数据业务的光纤链路中。这种配置只能应用于直接服务调用模型中，如图 4 中的 a)所示。

b)　纤外(Out-of-Fiber)

信令消息由 UNI-C 和 UNI-N 之间的专用通信通道来承载，通信通道和承载对应数据业务的光纤链路分离。这种信令配置可以应用于直接服务调用模型，也可以应用于间接服务调用模型，如图 4 中的 a)、b)所示。

这两种配置方式下，控制通道应支持 IP 包的传输。因而，该控制通道为 IP 控制通道或 IPCC。UNI 信令消息由 IP 包来承载。如：UNI 信令协议采用 RSVP-TE 协议，这时携带 RSVP-TE 消息的 IP 包会在 IPCC 上传输。

UNI 中，IPCC 可以通过下列方式之一来实现：

- SDH/OTN 纤内信令配置；
- 以太网 OAM 帧纤内信令；
- 纤外信令。

5.2　SDH/OTN 纤内信令配置(可选)

纤内信令配置见图 7。如图 7 所示，客户设备和 TNE 之间通过一个或多个双向 SDH/OTN 链路进行连接。每个双向链路由一对单向链路构成，其中一条从客户设备到 TNE，另外一条从 TNE 到客户

设备。一条或多条双向链路中的某些 SDH DCC 或 OTN GCC 开销字节用来实现双向 IPCC。

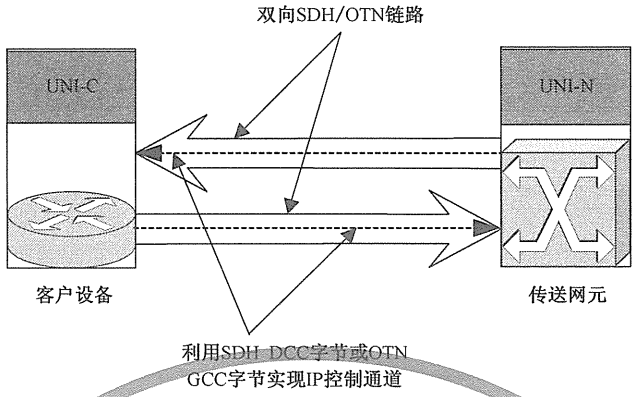


图 7 SDH 纤内嵌入式信令通道配置

纤内 IPCC 和点到点链路逻辑上是等同的。如果在客户设备和 TNE 之间有多条这样的 IPCC 链路可用，根据本地的策略，IP 包可以在任何一条 IPCC 上传送。

本部分中，基于 SDH 接口的 UNI 纤内 IPCC 应利用 SDH DCC 字节来实现，基于 OTN 接口的 UNI 纤内 IPCC 应利用 OTN GCC 字节来实现。

将 IP 包封装到 DCC 或 GCC 字节时本部分定义了两种选项。如图 8 所示，第一种选项将 IP 包封装到 PPP 包，PPP 包封装到 HDLC 链路帧并对帧进行比特填充。第二种选项将 IP 包封装到 PPP 包，PPP 包封装到使用 LAP-D 帧的 OSI 9577 包中。其中，选项 1 为必选支持，选项 2 为可选支持。客户设备和传送网元应能够至少支持这两种封装方式中的一种，也可以同时支持两种封装方式。对于一个给定 IPCC，当客户设备和传送网元设备同时支持两种封装方式时，客户设备和/或传送网元应能够支持通过手动配置来选择某一种封装方式。

客户和 TNE 对邻居设备的 IP 地址（用来发送 UNI 信令报文）的自动发现以及对 IPCC 的维护见第 6 章中的描述。

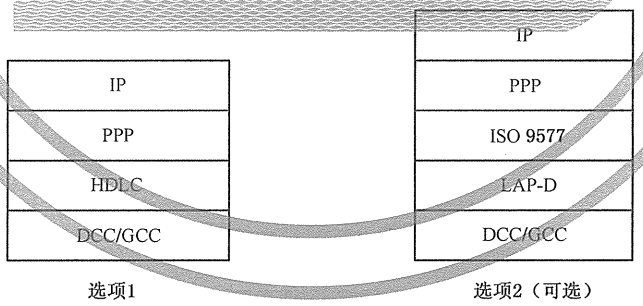


图 8 DCC/GCC 封装选项

5.3 以太网 OAM 帧纤内信令

当 UNI 的客户端是以太网设备时，信令消息可使用纤内的 OAM 帧实现传送。这些 OAM 消息在链路级别范围内有效，总体的格式可参考 IEEE 802.3ah 中的 5.7（以太网 OAM）。

5.4 纤外信令

5.4.1 纤外信令配置

纤外信令应用于 4.4 所描述的全部四种参考配置。本部分描述了图 9 所示的两种可选的纤外信令

方式。对于这两种选项，通过在 UNI-C 和 UNI-N 之间建立 IP 连接来实现纤外 IPCC，这种 IP 连接的建立和相对应的客户和 TNE 之间的数据链路数量相互独立。

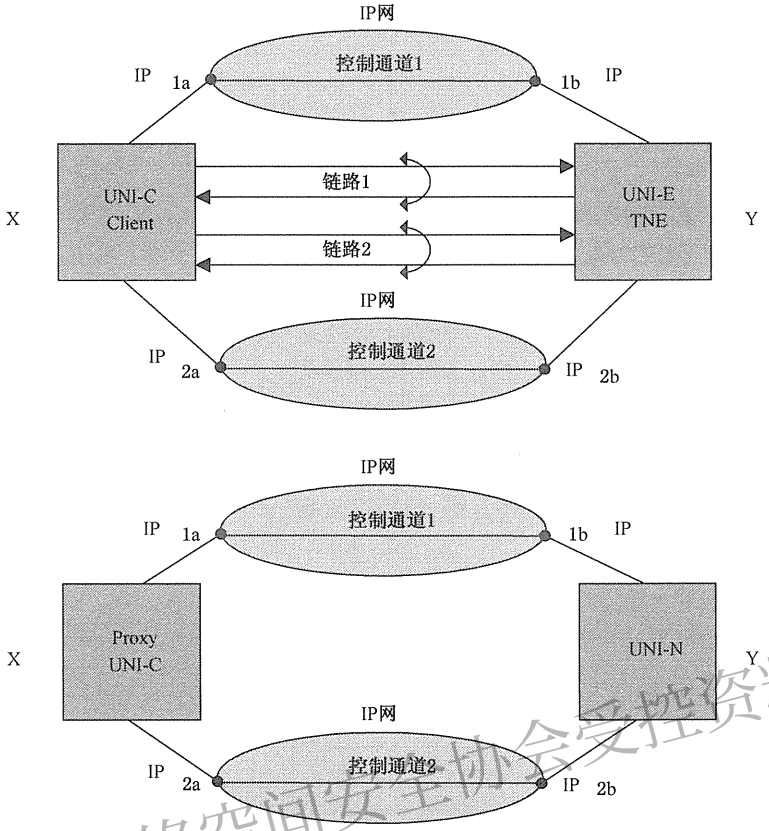


图 9　纤外信令配置

5.4.2　外部 IP 传送网络实现

这种配置情形下，UNI 信令消息由外部 IP 传送网络来承载。如图 9 所示，UNI-C 和 UNI-N 有一个或多个点接入 IP 网络。这些网络的特性在本部分中不指定。控制消息可以通过每个网络来进行发送而不需要进行协调。但对于每个 UNI-C 和 UNI-N 实体应配置对端邻居的 IP 地址以便能够发送控制消息。本部分假定实现 IPCC 的 IP 网络是安全的，即不要求额外的处理来保证信令消息在外部网络的安全传送。

5.4.3　专用信令网络实现(可选)

这种配置情形下，可以在 UNI-C 和 UNI-N 之间建立专有的双向连接业务。这种专用的连接作为 IPCC 来使用，信令消息可以在连接的负载中发送。本部分中，连接的建立和使用可以在对等实体上通过手工配置来控制。通过这种方式可以建立多条 IPCC 控制通道。任何一条可以用来发送特定的信令消息。在专用连接实现的 IPCC 上所发送的 IP 包要使用 PPP 到 SDH 的封装技术，见 IETF RFC2615 中的相关定义。

5.5　信令传送的实现

尽管对信令传送有三种不同的选择，本部分不排除在客户设备和 TNE 之间使用多种传送链路类型来实现。如：客户设备和 TNE 之间可以同时使用纤内和纤外 IPCC 来实现。类似地，TNE 可以和不同客户设备之间有不同的 IPCC 类型。因此，客户设备和 TNE 之间需要配置指定对应于一组数据链路

集合的 IPCC 实现类型。客户和 TNE 之间的 IPCC 状态跟踪维护处理见 6.1.2 中的描述。

6 UNI 链路管理功能

6.1 UNI 链路自动管理

6.1.1 概述

本节所定义的邻居发现处理允许 TNE 和直接连接的客户设备之间了解彼此之间的节点标识以及和本地端口相连的远端端口标识。本节中所定义的 IP 控制通道维护处理允许客户和 TNE 连续监控和维护可用的 IP 控制通道列表。

UNI 中，邻居发现和 IPCC 维护处理是可选的。如果客户设备和 TNE 之间不实现如本节所描述的邻居发现功能，相关的 UNI-C 和 UNI-N 应手动配置邻居节点标识和远端端口标识相关的信息。如果不实现如本节所描述的 IPCC 控制通道维护处理，应在所使用的信令协议中实现该功能（如，RSVP-TE 协议中需要支持 HELLO 机制）。本节所描述的邻居发现功能的实现可以避免手工配置邻居和端口连接相关的信息。因此，自动发现功能可以排除依靠手动配置而导致的潜在的错误。而且，邻居发现还能自动检测出物理连接的错连情形（如，某个端口的发送和接收错误地连接到远端的两个不同的端口上）。

TNE 可以和多个客户设备相连，客户设备也可以和多个 TNE 相连，但对于邻居发现和 IPCC 维护处理的定义，考虑单个 TNE 以及和它直接相连的客户设备就已经足够。单个 TNE 和客户之间由一组数据链路进行连接，如 SDH 或以太网链路。而且，客户和 TNE 之间可以存在一条或多条 IPCC。IPCC 可以以纤内方式或纤外方式来实现（如第 5 章中的定义）。在所有的情形下，本节所定义的处理需要 TNE 和客户设备从数据链路中发送或接收某些纤内消息。

本节所定义的协议机制是基于 IETF RFC4204 所定义的链路管理协议（LMP）。LMP 中 IPCC 维护定义了两个基本的步骤：

a) 初始配置：TNE 和客户设备中为他们之间的每条 IPCC 配置基本的参数。

b) Hello：TNE 和客户设备中为他们之间的每条 Hello 协议实例进行初始化。该协议负责监控控制通道的可用性。

在由 DCC 字节实现的纤内信令的情形下（IPCC 嵌入在每条数据链路中），端口的连通性信息在 IPCC 配置处理过程中可以进行交换。在纤外信令的配置情形下，为验证端口的连通性，LMP 定义了如下的第三个步骤。

c) 链路验证：为确定端口的连通性，每条数据链路都需要发送纤内测试信息。在 IPCC 控制通道中所发送的控制消息用来配合测试处理。

TNE 和客户之间多个端口的连接期望聚合到一条或多条逻辑链路，每条逻辑链路由一个 TNA 地址来标识。对于聚合链路相关参数的验证处理非常有用，如链路带宽、链路编码类型、交换能力等。LMP 中的链路属性关联处理可以验证链路相关的参数。LMP 术语中聚合的逻辑链路指流量工程链路（TE Link）。

UNI LMP 处理过程中用到如下的标识：

- 节点标识（Node ID）
  用来标识节点（TNE 或客户）的 IP 地址。如路由器客户的节点 ID 是它的路由器 ID，见 7.2；
- 接口 ID
  用来标识每个端口的 32 位逻辑标识；
- 控制通道 ID
  节点应为每条 IPCC 分配的本地唯一的标识。该标识为 CCID，为 32 位数并在节点 ID 范围内唯一。TNE 和客户之间相同的控制通道在两侧可以分配不同的 CCID，见 7.2.1。

- 链路 ID

  UNI 中和 TNA 地址相对应。由于 LMP 仅支持 IPv4 和 IPv6 链路 ID 格式。链路属性关联功能不支持使用 NSAP TNA 地址格式作为链路 ID 信息，见 6.1.3。

所有 LMP 消息使用 UDP 协议，并使用 LMP 端口号“701”(受带内消息传送机制限制的测试消息除外)。除标准的 IP 头和 UDP 报文头之外，所有的 LMP 消息(受带内消息传送机制限制的测试消息除外)有如下的公共报文头“见 LMP”。

LMP 消息可以通过纤内或纤外的信令通道进行发送。为支持 UNI 的邻居发现和 IPCC 维护，只需要部分 LMP 消息。表 4 总结了 UNI 支持的 LMP 消息。编号为 1～4 的消息为 LMP 控制通道维护消息，它们在 IPCC 上发送，并携带相应 IPCC 的 CCID。编号为 5～16 的消息为 LMP 和数据链路相关的消息。除测试消息之外(消息编号为 10)，所有消息由 IPCC 来承载。测试消息总是在每条 TNE 和客户之间的每条数据链路的纤内进行发送。

表 4 LMP 邻居发现和控制通道维护消息

LMP 消息类型编号	LMP 消息类型
1	Config …… 配置消息
2	ConfigAck …… 配置确认
3	ConfigNack …… 配置出错
4	Hello …… 握手消息
5	BeginVerify …… 开始验证
6	BeginVerifyAck …… 验证确认
7	BeginVerifyNack …… 验证出错
8	EndVerify …… 结束验证
9	EndVerifyAck …… 结束验证确认
10	Test …… 测试
11	TestStatusSuccess …… 测试成功
12	TestStatusFailure …… 测试失败
13	TestStatusAck …… 测试确认
14	LinkSummary …… 链路属性关联
15	LinkSummaryAck …… 链路属性关联确认
16	LinkSummaryNack …… 链路属性关联出错

下面的章条中，描述了用来做邻居发现和 IPCC 维护的 LMP 消息以及对应的处理。LMP 状态移机制、LMP 消息格式、LMP 对象定义分别参考 IETF RFC4204 中 11、12、13 的内容，本部分不做详细描述。

6.1.2 控制通道管理

6.1.2.1 概述

邻居发现处理被客户或 TNE 用来确定和自己通过数据链路相连的远端 TNE 或客户的标识，以及数据链路互连的方式。邻居发现处理允许客户和 TNE 交换节点 ID 信息、确定本地和远端端口的映射关系(指本地接口 ID 和远端接口 ID)、关联相关数据链路的配置参数等。IPCC 维护处理允许对等的信

令实体之间建立和维护 UNI 信令控制通道的连通性。图 10 描述了设备运行 UNI 邻居发现协议时所维护的端口映射表。当设备提供了端口后，本地接口 ID 和“其他信息”要进行配置。其中后者也可以通过 LMP 链路属性关联消息交换的一部分进行交换。所有远端信息可以自动确定。

节点 ID	本地接口 ID	其他信息（如、端口速率、端口交换能力等）	远端节点 ID	远端接口 ID	其他信息
32 位地址	1		32 位地址	5	
32 位地址	2		32 位地址	8	
32 位地址	3		未知	未知	

图 10　UNI 两端端口配置信息表

除端口映射表之外，IPCC 状态（包括本地/远端 CCID）应进行维护。

6.1.2.2　控制通道参数协商

控制通道配置步骤由交换 3 个 LMP 消息来构成：配置消息、配置确认和配置出错消息。发起对 IPCC 配置的节点（TNE 或客户）应通过 IPCC 发送配置消息到对等实体。配置消息包含一个或多个 LMP 对象。UNI 中，配置消息应包含 HelloConfig 对象，该对象中包括 HelloInterval 和 HelloDeadInterval（见 IETF RFC4204 LMP 协议中的定义）。这些参数是否能够协商可以通过 HelloConfig 对象中的指示来确定。节点收到配置消息后应以配置确认或配置出错消息进行响应，并指示接受或拒绝配置消息中的配置参数。但发送配置消息的节点收到配置确认消息后完成配置的处理。当发送配置消息的节点收到配置出错消息后，可以终结不成功的配置处理步骤，或发送参数经过调整的其他的配置消息。

客户和 TNE 可能会同时向对端发送配置消息。这时，客户应停止发送配置消息并响应所接收到的配置消息。TNE 应忽略所接收到的配置消息并等待配置确认或配置出错消息。这种情形下，LMP 协议中所定义的竞争解决的处理，可以等同于 TNE 分配了比客户更大的节点 ID。配置消息的发送方式依靠信令传送机制：

a）　纤内信令

配置消息应该以多播地址（224.0.0.1）为目的地址进行发送。配置确认或配置出错消息应以收到的配置消息的源 IP 地址为目的地址进行发送。

b）　纤外信令

配置消息应以邻居节点 IPCC 地址为目的地址进行发送，IPCC 地址在控制通道两端都会进行配置。

6.1.2.3　Hello 协议和 IPCC 维护

当客户和 TNE 之间的 IPCC 被激活后，两个节点通过交换 Hello 消息来维持 IPCC 的连通性。执行 Hello 协议的每个节点应通过 IPCC 周期性的发送 LMP Hello 消息。IP 包的目的地址应从配置处理过程中获取。Hello 消息的周期长短由配置节点建立的 HelloInterval 值来控制。

如果节点发送 Hello 消息，但在 HelloDeadInterval 周期内一直收不到任何 Hello 消息，相关的 IPCC 应声明为 down 状态。这种情形下，需要再次尝试配置步骤。

6.1.2.4　IP 控制消息发送通道的选择

LMP 所维护的端口映射表允许节点从给定的实例中确定一组可用的 IPCC。和一个邻居之间可能

有多个 IPCC 的状态为“Up”。为向该邻居发送 IP 包，通道管理者应首先将目的 IP 地址和邻居节点地址进行匹配，并选择一条状态为“Up”的 IPCC 来发送 IP 包。选择哪条 IPCC 的具体算法不具体指定。在两个相邻节点之间发送信令消息时双方对 IPCC 的选择不需要保持一致。

节点选择到邻居节点的 IPCC 并一直使用该 IPCC 直到该通道状态发生改变。原先选择的 IPCC 状态发生改变后，节点可以任意选择和邻居节点相连的状态为“Up”的其他 IPCC。如果没有这样的 IPCC 时，UNI 信令消息无法发送到邻居节点，节点应产生网络管理事件来上报控制通道失效的信息。

6.1.3 链路属性关联

链路属性关联功能用来在 TNE 和客户之间帮助对链路属性进行配置。该功能不是必须功能，可以根据需要来调用。UNI 中，链路属性关联处理只有在使用 IPv4 和 IPv6 作为 TNA 地址格式时才能够调用。当使用 NSAP 格式的 TNA 地址时不能调用。链路属性关联功能是通过在 LMP 协议中使用链路属性关联(LinkSummary)消息来实现。如果使用链路属性关联处理应该在链路正常启用之前，也可以在链路状态为“UP”后的任意时间进行，但不能在链路验证处理过程中进行。

交互链路属性关联消息用来：

- 将多条数据链路聚合到一条逻辑链路(通过 TNA 来标识)；
- 交换、关联、或改变链路参数；
- 交换、关联、或改变接口 ID。

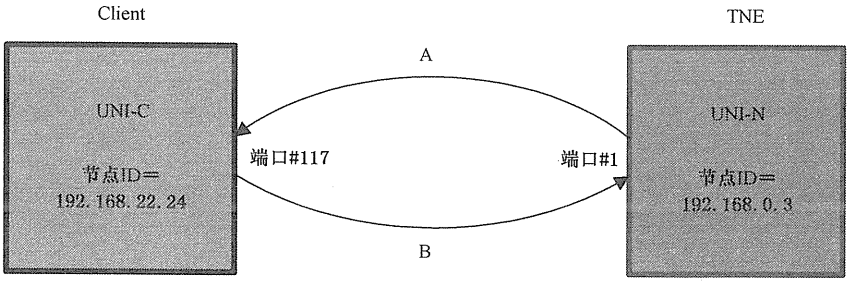
使用链路属性关联、链路属性关联确认和链路属性关联出错(见 LMP 协议中定义的消息格式)消息来定义链路属性关联消息交换过程。UNI 中这些消息的处理流程见 IETF RFC4204 LMP 协议中的定义。

6.1.4 物理连通性验证(邻居发现)

端口连通性验证是邻居发现的重要部分。用来验证数据通道所提供的物理连通性，并用来交换接口 ID 信息。在纤内配置方式下，端口连通性验证融合在 LMP 配置处理过程中。在纤外配置方式下，需要单独的处理。具体如下面的描述：

a) 纤内配置

这种配置方式下，控制通道和数据链路端口为一一对应的映射关系。因此，IPCC 应该分配和接口 ID 相同的值。同样，端口连通性验证可以在配置处理过程中完成，如图 11 所示。这里，TNE 通过发送配置消息来发起配置处理。除 HelloConfig TLV 之外，配置消息还包括 TNE 本地节点 ID 和 CCID(和接口 ID 相同)。客户设备返回配置确认消息作为响应，其中包含它自己的本地节点 ID 和 CCID 以及远端的节点 ID 和远端 CCID。这些消息允许 TNE 和客户设备填充端口映射表中的“远端节点 ID”和“远端端口 ID”域的内容(如图 11 所示)。



A:配置(本地节点 ID=192.168.0.3,本地 CCID=1)

B:配置确认(本地节点 ID=192.168.22.24,本地 CCID=117
远端节点 ID=192.168.0.3,远端 CCID=1)

图 11 纤内信令方式下端口连通性验证过程

b） 纤外配置

这种配置方式下，控制通道和数据端口之间没有一一对应的映射关系。因此，IPCC 配置处理不能确定数据端口的连通状态，需要其他的处理步骤来完成连通性验证。如图 12 所示，为完成客户和 TNE 之间的关联处理，在端口连通性验证发起之前应在它们之间建立 IPCC 通道。通过 IPCC 通道，节点（客户或 TNE，名为“发起者”）向远端节点发送开始验证消息发起端口连通性验证处理。该消息的本地链路 ID 域应置 0。而且，消息中开始验证消息对象的其他域应设置如下：

- Flag 域：第 0 个比特位（LSB）应置 1 用来指示验证所有未分配的链路。第一个比特位应置 1 来指示 端口。其他比特位应置 0；
- VerifyInterval 域：该域应设置成已经配置的值；
- 数据链路编号域：该域应设置成发送节点未分配链路的编号值。
- 其他域的设置参见 IETF RFC4204 LMP 协议中的定义。

为响应开始验证消息，远端节点可以向发起者发送验证确认或验证出错消息。如果发送验证确认消息，应包括 32 位的验证码（VerifyId）用来唯一标识要验证的和远端节点连接链路的具体实例。验证码拷贝到所有其他由发起者或远端节点所发送的与验证相关的消息中。远端节点发送验证出错消息时，表示接收者不能够或不愿意做链路验证处理。对每条数据链路的测试可以依次进行，也可以多条数据链路同时测试。这与节点的处理能力有关。为终结处理过程，发起者向远端节点发送结束验证消息，远端节点返回结束验证确认消息。所有这些消息中的域中值应按 LMP 协议规定进行设置。

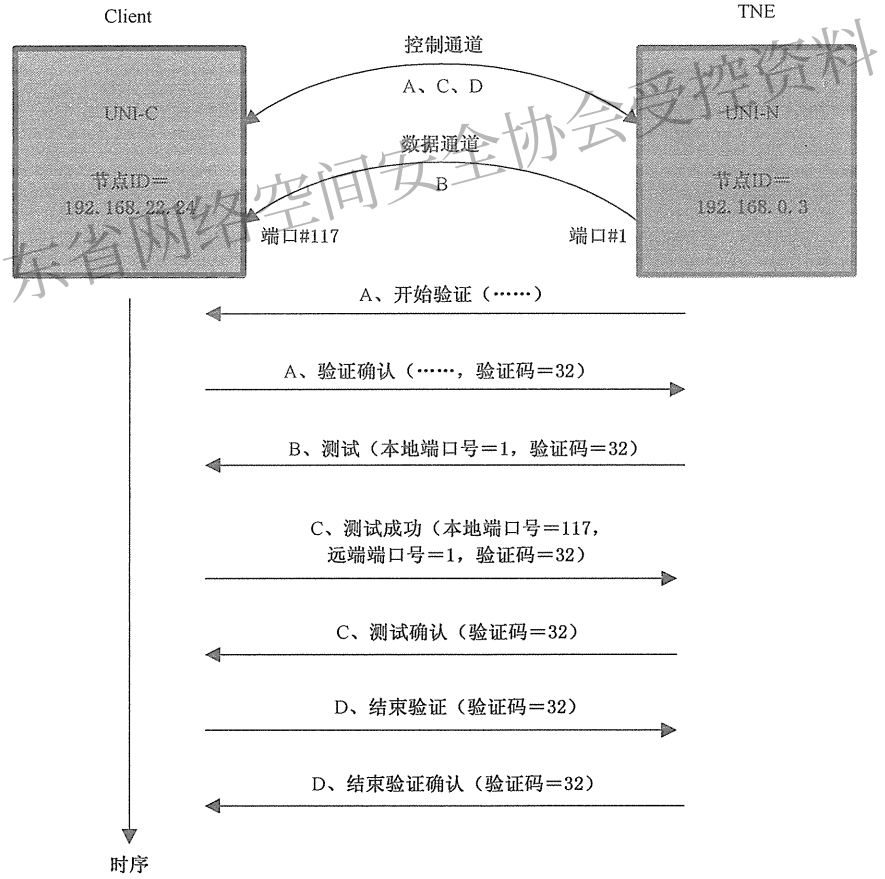


图 12 纤外信令方式下的端口连通性验证过程

端口连通性验证处理过程如下：当发起节点和远端节点同意开始测试端口后，发起节点通过每条单向光纤链路发送“测试”消息。当接收到“测试”消息后，接收者使用消息中的 VerifyID 域中的值将测试消息关联到特定的要验证的处理实例中，并将所接收到的远端接口 ID 映射到对应的本地值。然后，接

收者通过控制通道向发起者发送测试成功消息，其中包含本地接口标识和远端接口 ID。如果接收者在特定的时间间隔内无法接收到测试消息，会向发起者发送测试失败消息。最后，发起者返回测试确认消息给它的对等实体用来应答成功或失败的消息。UNI 中，所有消息的域值应按 LMP 协议规定进行设置。

物理连通性不一致的检测是邻居发现过程中重要的特征，并将检测过程融入到 LMP 端口验证处理过程中。当双向端口的发送端口和远端的一个端口相连，接收端口和远端另外一个端口相连时引起物理错连。下面描述了纤内和纤外 IPCC 的错连检测处理过程：

a) 纤内信令

当出现下列情形之一时，节点检测到错连：

- 当发送配置消息后，没有收到配置确认(或配置出错)响应消息；
- 所收到的配置确认或配置出错消息中没有包含所期望的“远端节点 ID”或“远端 CCID”。

b) 纤外信令

当出现下列情形之一时，节点检测到错连：

- 收到测试失败消息；
- 所收到的测试成功消息中包含了不期望的接口 ID(远端接口 ID 或本地接口 ID)。

附录 A 中描述了错连检测的实例。

6.1.5 LMP 消息的可靠传递

6.1.5.1 消息编号的用法

LMP 消息使用消息 ID(MESSAGE_ID)和消息 ID 确认(MESSAGE_ID_ACK)消息来支持可靠的消息交互。本条描述了这些对象的用法。消息 ID 和消息 ID 确认对象包含一个消息编号(Message_Id)域。仅有一个消息 ID/消息 ID 确认对象会被包含在每个 LMP 消息里。对于控制通道配置消息而言，消息编号域在 CCID 范围内唯一。对于 TE 链路配置消息而言，消息编号在 LMP 邻接范围内唯一。

消息 ID 对象消息编号域包含一个发生器选择的值。该值只能单调地增长。如果同一个 CCID(对于控制通道配置消息)或 LMP 邻接(对 TE 链路配置消息)已经发送过一个值，则该值被认为是用过的。消息 ID 确认对象的消息编号域包含被确认消息的消息编号域。携带消息 ID 对象的消息如果未被确认，需要重传，直到消息被确认，或者达到重传限制。值得注意的是，32 位的消息编号值可以绕接。下面的表达式可以用于测试一个新接收的消息编号值是否小于之前接收到的值：

```
If ((int) old_id-(int) new_id > 0)
{
    新 ID 值 <  旧 ID 值;
}
```

对接收消息进行处理的节点应该检查新收到的消息是否失序，并且确定是否可以忽略。失序的消息能够通过检查消息编号域的值进行识别。如果消息被确定失序，消息应该被丢弃。如果是一个配置消息，消息编号值小于之前从发送者的 CCID 收到的最大的消息编号值，则该消息应被当作失序来处理。如果是一个通道状态消息(ChannelStatus)，并且消息编号小于之前发送者的配置流量工程链路最大的消息编号值，那么接收者应该检查之前接收的通道状态消息所包含的每条数据通道的消息编号值的状态。如果消息编号值大于最近接收到的与消息所包含的至少一条数据通道相关的消息编号值，该消息不应该被认为失序。然而，对于所有最近所接收到的消息编号值大于该消息编号值的数据通道，其状态不应该被更新。其他的消息 ID 不应该被认为失序。

6.1.5.2 LMP 消息重传机制

在节点发送一个需要确认的消息之后，他应该在 Ri 秒之后立即启动一次重传。如果在 Ri 秒之前

接收到一个对应的确认消息，应该取消消息重传。否则他应该在(1＋Delta) * Ri 秒后启动重传。重传一直持续到接收到合适的确认消息或者达到快速重试限制 Rl。

当传送一个需要确认的消息的时候，发送节点可以使用下面的算法：

在初始化发送之前，初始化 Rk = Ri 和 Rn = 0。

```
while (Rn++ < Rl) {
    发送消息;
    等待 Rk 毫秒;
    Rk = Rk * (1 + Delta);
}
```

不同时地，当发送节点接收到对应地确认消息，改变重传计数 Rn 为 Rl。

发送节点不会在配置消息或者链路属性关联消息里公告或者协商指数衰减参数。

6.1.6 LMP 重启恢复处理

本条描述控制平面重启之后 LMP 状态再同步的机制。在控制通信故障后，第一个控制通道的启动将导致控制平面的重启。控制通信故障可能是 LMP 邻接故障或者节点故障引起的，这时，LMP 控制状态丢失，但数据平面不受影响。后者可以通过设置 LMP 消息的公共头的“LMP 重启”位来检测。如果控制平面的故障是由于控制通道的丢失，LMP 链路信息应该保留。也有可能在节点故障的过程中，节点有能力保留 LMP 链路信息。然而，在这两种情形中，数据通道的状态均要同步。假设在控制平面重启的过程中，节点 ID(Node_Id)和本地接口 ID(Interface_Ids)不变。

在节点控制平面重启之后，控制通道应采用参数协商的过程来重新建立。在重建控制通道的过程中，配置消息应该使用单播 IP 源和目的地址的方式发送。

如果控制平面的故障是由于节点故障，这时 LMP 控制状态丢失，“LMP 重启”标志应在 LMP 消息里设置，直到接收到一个 RcvSeqNum 和本地 TxSeqNum 相等的 Hello 消息。这表示控制通道启动，并且 LMP 邻居检测到这个重启。

下面假设 LMP 组件重启只发生在一个 TE 链路的端点。如果 LMP 组件的重启发生在 TE 链路的两个端点，那么应该采用正常的链路属性关联过程。

一旦控制通道启动，LMP 邻居应对通过邻接的每条 TE 链路发送一个链路属性关联消息。所有链路属性关联消息的 N bit 位应设置为 0，指示参数是不协商的。消息提供了本地/远端 Link_Id 和 Interface_Id 映射，相关数据链路参数和目前那条数据链路分配了用户流量的指示。如果节点接收到了链路属性关联消息，就检测本地的配置。如果在重启过程中，节点能够保留 LMP 链路信息，就应处理链路属性关联消息，另外接收到的 DATA_LINK 对象的分配/去分配标志应优于其他任何本地值。然而，如果在重启的过程中，节点不能保留 LMP 链路信息，节点应接受接收到的链路属性关联消息的数据链路参数，并且响应一个链路属性关联确认消息。

在完成链路属性关联交换之后，重启了控制平面的节点应该对该 TE 链路发送一个通道状态验证请求(ChannelStatusRequest)消息。节点也应该对所有未分配的数据通道的连通性进行验证。

6.2 UNI 手动管理功能

6.2.1 邻居和数据链路手工配置

当 UNI 接口中任意一侧不支持邻居链路自动发现功能时，需要在 UNI-C 和 UNI-N 通过手动配置对端节点标识和数据链路端口标识信息映射表。

6.2.2 控制链路手工配置

当 UNI 接口中任意一侧不支持控制通道维护功能时，需要在 UNI-C 和 UNI-N 通过手动配置可用的控制通道信息。

7 UNI 编址

7.1 UNI 编址方式选择

UNI 编址存在 OIF 编址和 IETF 编址两种方式。实际部署过程中可以根据需要选择两种编址方式中的一种。编址的选择应和信令方式的选择保持一致。即，采用 OIF UNI 信令时，编址需要采用 OIF UNI 的编址；采用 IETF GMPLS UNI 信令方式时，需要采用 IETF 编址方式。

7.2 OIF UNI 编址方式

7.2.1 控制实体的标识

UNI 编址能够完成与光传送网络相关的不同实体之间的标识和可达性功能以及控制平面的连通功能。为描述 UNI 协议运行所涉及到的多个地址空间，需要介绍相关的需要标识的实体以及相关的术语。

如第 5 章中的描述，信令协议消息通过 IP 控制通道进行交互。概括如下：

a） IP 控制通道(IPCC)

在 UNI-C 和 UNI-N 节点之间传送 IP 包的数据通道。IPCC 的实现有多种方式，UNI-C 和 UNI-N 节点可以通过多个 IPCC 控制通道进行连接(多个控制通道之间的关系既可以使用对等配置，也可以使用主备配置)，见第 5 章中的描述。

b） 信令通道

UNI-C 和 UNI-N 之间交换信令消息地逻辑通道。信令通道的实现可以使用两个 UNI 对等实体之间的所有 IPCC 以及各个对等体的信令协议实体来实现。信令通道故障可能是因为所有的 IPCC 控制通道失效或信令实体失效。

图 13 描述了 UNI-C 和 UNI-N 之间的配置实例，UNI-C 和 UNI-N 分别驻留在对应的客户设备和 TNE 设备中，并且由 4 条数据链路连接。每个数据链路可以用来实现纤内 IPCC，如第 5 章中的描述。两个节点之间还有纤外 IPCC，他们通过外部以太网实现。IP 网络中，每个 IPCC 在两个节点之间代表一个点到点的链路，可以是有编号的也可以是无编号的。对于前者，每个 IPCC 在每一端都分配一个 IP 地址来标识。对于后者，每个单独的 IPCC 不分配 IP 地址，而是在每一侧分配一个接口标识，每一端可以通过分配给节点的 IP 地址和接口索引来对 IPCC 进行标识。

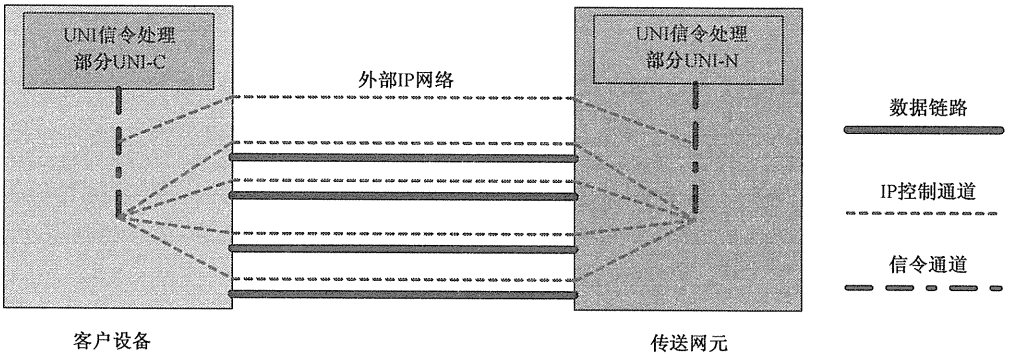


图 13 信令通道和 IP 控制通道配置

如图 13 所示，UNI 信令通道位于一条或多条 IPCC 的上一层。信令通道自身应唯一标识，当底层的 IPCC 状态发生改变时不会影响到上层的信令通道。可以通过使用节点标识(NODE ID)来实现对信令通道的标识。

7.2.2 UNI 地址空间

和 UNI 信令相关的地址空间描述如下：

a) 内部传送网络地址

这些地址为传送网络内部的节点地址，可以用来作内部路由，以及网络管理的目的。地址见图 14 中如 TNE A 和 TNE B。这些地址信息不会暴露给网络客户，也不是 OIF 标准化的内容，因此不在本部分的研究范围之内。

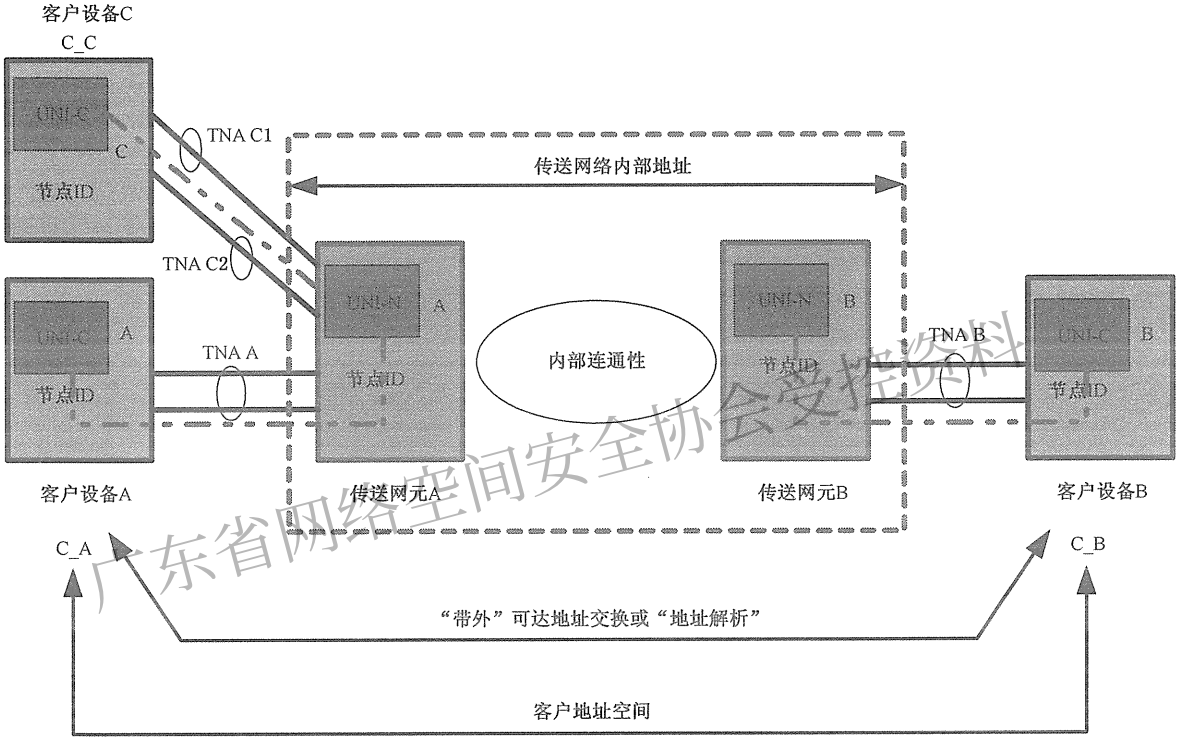


图 14 UNI 地址空间

- UNI-N 和 UNI-C 节点 ID

  这个 IP 地址用来标识终结在控制平面 UNI-C 或 UNI-N 的连接。节点 ID 不会随着底层 IPCC 的地址或状态变化而变化。UNI 中，节点 ID 为 IP v4 地址。并且在 UNI-C 和 UNI-N 之间的通信控制域内保持唯一。节点 ID 不需要全局唯一，但应在需要通信的对等实体中能够唯一标识通信实体。当然，如果节点 ID 为全局唯一的 IP 地址时很容易唯一标识通信实体。标识 UNI-C 或 UNI-N 中的连接控制平面终端。节点 ID 不随 IPCC 状态或地址变化而变化。一般情形下，节点 ID 从私有的 IPv4 地址空间中进行分配。节点 ID 是否过渡到 IPv6，要依赖于 UNI-C 和 UNI-N 节点是否增加 IPv6 能力。节点 ID 由管理该节点的网络操作员分配。

- IP 控制通道标识(CCID)

  如 6.1 描述。为了邻居发现、服务发现、以及 IPCC 维护，应对 IPCC 的端点进行标识。该标识名为 CCID。IPCC 可以是有编号的，这时在每一端可以使用 IPv4 地址对 IPCC 进行标识。如果无编号，可以通过每一端的节点 ID 和接口索引来唯一标识 IPCC。CCID 为 32 比特并在节点范围内唯一，编号独立于 IPCC 端节点的地址。

- 传送网络分配的地址(TNA)

  UNI 连接端点由传送网分配地址 TNA 来标识。每个 TNA 地址是一个由传送网分配给连接 UNI-N 和 UNI-C 的一条或多条数据承载链路的全局唯一地址。传送网操作员分配 TNA 地址是基于传送网网络操作员策略。TNA 地址结构见 7.2.4。TNA 地址为 ITU-T G.8080 中定义的关于传送资源地址的实例。

b) 客户地址

如图 14 所示,其中 C_A,C_B 和 C_C 为客户地址。客户地址应服从客户网络的编址方案(例如 IP,ATM 等),并且不通过信令消息直接交换。本部分实现了客户和 TNA 地址空间的分离,因此,客户地址不在本部分的讨论范围。

7.2.3 逻辑端口标识

如上面的描述,单个 TNA 地址可以指示一条或多条数据链路。数据链路组中的单个链路共享相同的 TNA 地址并在每一端通过逻辑端口再进行标识。逻辑端口标识根据每端节点本地的策略进行分配。因此,连接客户和 TNE 的每条数据链路会分配两个逻辑端口标识,一个在客户侧,另外一个在 TNE 侧。对于 UNI,逻辑端口标识为 32 位整数编码并且应在节点内部唯一。每个逻辑端口标识和一个物理端口一一对应。逻辑端口标识的编号规则和物理端口的编号规则可以不同。这样可以避免将网络运营商内部的端口编号规则暴露给客户。另外,运营商可以为数据链路维护而给定和客户设备相同的逻辑端口标识,而不考虑终结链路的 TNE 内部的物理端口标识。

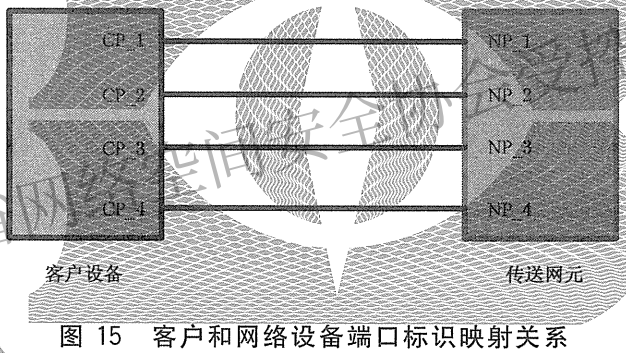


图 15 客户和网络设备端口标识映射关系

图 15 描述了 TNE 和客户设备之间有多条数据链路的情形。每一侧相关的逻辑端口标识分别表示为 CP_i 和 NP_i。由于逻辑端口信息由连接建立和拆除消息承载,UNI 信令的正确运行要依赖于客户和 TNE 之间逻辑端口标识的正确映射。映射关系的建立可以通过下面的方式之一来建立:

- 自动建立,利用 6.1 中所描述的 LMP 邻居发现和链路验证处理来建立映射关系;
- 手动建立,手动配置 UNI-C 和对应 UNI-N 实体之间的映射关系。

本部分中,UNI-C 应该允许手动配置端口映射信息。UNI-N 也应该能指定 UNI-C 使用每个端口的映射关系。如图 15 所示,可以配置 UNI-C 使用 NP_1(由 UNI-N 分配)而不是 CP_1(由 UNI-C 内部分配)来标识第一条数据链路。

7.2.4 TNA 地址结构及作用

TNA 地址用来标识 UNI 连接的端点。TNA 地址空间应该足够大以便能提供全球网络节点的地址需要,并且要能够支持层次级别以及地址汇聚。

UNI TNA 地址能够提供 IP 或 NSAP 协议族地址。如:UNI 信令协议要支持如下 3 种类型的 TNA 地址:

- IPv4 地址(32 位);
- IPv6 地址(128 位);

- NSAP 地址(160 位):NSAP 格式结构参考 ITU-T X.213 协议规定。

下面的 C-Type 类型允许 LMP 以 NSAP 格式对 TNA 地址进行编码:

LINK_ID 类型

Class = 3

C-Type = 7,NSAP LOCAL_LINK_ID

C-Type = 8,NSAP REMOTE_LINK_ID

TNA 地址由 UNI 信令消息承载,用来唯一标识连接的数据端点。当将相同的 TNA 地址分配到多条终结在客户设备的数据链路时,每个数据链路使用逻辑端口标识来进行区分。这种情形下,TNA 地址和逻辑端口标识的组合来唯一标识一条数据链路。而且,根据所选择接口的特性以及所请求的业务,应选择解复用信息,如通道。这些信息由信令消息承载并作为"标签"。"标签"指示出详细的复用信息,这些信息用来在给定的数据链路范围内标识出逻辑通道。信令详细的描述见 8.2.4。

UNI 中,允许不同种类的 TNA 地址之间存在连接。如,源端 TNA 地址可以使用 IPv4 格式的地址,目的端 TNA 地址可以使用 IPv6 格式的地址。这种情形下需要进行地址之间的转换,但这些内容不在 UNI 规范的研究范围。地址信息如何在信令消息中承载见 8.2.4 的描述。

传送网络维护 TNA 地址和传送网络地址之间的映射关系的方法不在 UNI 规范的研究范围。

7.3 IETF GMPLS UNI 编址方式

7.3.1 编址原则

IP 编址为 IETF 中的通用编址方式,GMPLS 中不考虑 IP 编址外的其他编址方式(如 NSAP)。在 GMPLS 中对于链路的编址方式有 Numbered(有编号)和 Unnumber(无编号)两种,具体可参见 IETF RFC4990。

7.3.2 UNI-C 和 UNI-N 编址

使用 GMPLS 支持 UNI,需用到 UNI 地址空间。用于标识 UNI-C 和 UNI-N 的地址属于相同的地址空间。这个地址空间可以采用和运营商网络中核心节点间通讯相同的地址空间,或者可以是由核心节点支持的 VPN 中用到的地址空间。也即在一个 Overlay 模型实例中的〈UNI-C,UNI-N〉形成的集合使用相同的地址空间,在该集合中的 UNI-C 通过 GMPLS 能建立 UNI-C 间的 LSP,而不需进行地址转换。这些 LSP 经过了运营商网络的核心节点。

当一个运营商核心网络支持多个 Overlay 模型的实例时,出于保密性的需求,可能使用一个或多个地址空间,由于一个 Overlay 模型实例中的 UNI 的地址空间由〈UNI-C,UNI-N〉对组成,这样对核心节点的地址不需进行改变。

在一个 Overlay 实例中,用于标识 UNI-C,UNI-N 的节点 ID(Node ID)为 Ipv4 或 Ipv6 地址。

7.3.3 控制通道编址

UNI-C 和 UNI-N 间可存在一条或多条控制通道(CC),用于传送 GMPLS 的控制消息,控制通道可以是带内或带外。在控制通道上传送 IP 包,控制通道的两端采用 IP 编址,在传送 IP 包前,本端需知晓控制通道远端的 IP 地址,该地址可采用自动发现或人工配置的方式实现。

为了控制通道维护,链路发现采用 CC-Id(Control Channel ID)对控制通道进行标识,一条控制通道的两端分别分配一个 CC-Id。CC-Id 在节点内部唯一,为 32 bit 的非 0 整数。

7.3.4 UNI 连接端点编址

UNI 连接端点编址是对连接 UNI-C 和 UNI-N 间的一条或多条数据链路的编址。该编址可用

UNI-C 和 UNI-N 间的有编号(Numbered)的流量工程链路(TE Link)的编址标识。当采用链路绑定(Link Bundling)时,一条 TE Link 中包含多条数据链路,此时需对 TE Link 中的每条数据链路的两端进行标识,该标识采用 32bit 的无编号接口 ID。

8 UNI 信令功能

8.1 UNI 信令方式选择

UNI 信令方式存在 OIF UNI 信令和 IETF GMPLS UNI 信令两种方式。实际部署过程中可以根据需要选择两种信令方式中的一种。UNI 信令方式的选择需要和 UNI 编址方式的选择保持一致。即:采用 OIF UNI 信令方式时,编址需要采用 OIF UNI 的编址方式;采用 IETF GMPLS UNI 信令方式时,需要采用 IETF 的编址方式。

8.2 OIF UNI 信令功能

8.2.1 UNI 抽象消息

8.2.1.1 UNI 抽象消息概述

本节描述了 UNI 信令消息,见表 5 所示。这些消息标明为"抽象"是因为具体的实现是根据所使用的信令协议不同而不同。8.2.4 描述了 RSVP-TE 协议中针对抽象消息的具体消息实现。下面的描述中,"发起 UNI-C"和"终结 UNI-C"用来表示业务两端的用来发起和终结给定信令行为的实体。对于一个给定的信令行为,发起 UNI-C 不一定是发起原始业务请求的 UNI-C。如,用来进行业务状态查询或删除的发起 UNI-C 可以和发起原始业务建立请求的 UNI-C 不同。对于信令代理,相同的代理 UNI-C 可以同时当作业务的两个端点来使用。

表 5 UNI 接口抽象消息

消息编号	抽象消息描述	消息流方向
1	业务创建请求	UNI-C→UNI-N & UNI-N→UNI-C
2	业务创建响应	UNI-N→UNI-C & UNI-C→UNI-N
3	业务创建确认	UNI-C→UNI-N & UNI-N→UNI-C
4	业务删除请求	UNI-C→UNI-N & UNI-N→UNI-C
5	业务删除响应	UNI-N→UNI-C & UNI-C→UNI-N
6	业务状态查询	UNI-C→UNI-N & UNI-N→UNI-C
7	业务状态查询响应	UNI-N→UNI-C & UNI-C→UNI-N
8	通告	UNI-N→UNI-C
9	业务修改请求	UNI-C→UNI-N & UNI-N→UNI-C
10	业务修改响应	UNI-N→UNI-C & UNI-C→UNI-N
11	业务修改确认	UNI-C→UNI-N & UNI-N→UNI-C

每个 UNI 消息都有一组相关的 UNI 属性。对于给定的信令消息有些属性是必选的,有些是可选的。下节中对各种消息的具体情形进行了描述。另外,网络可能不支持所有请求的属性。这种情形下,网络应该在响应消息中对客户发送适当的指示信息。由于与具体的使用协议相关,本节不对这些属性的编码进行定义。可以参考 8.2.4 中 RSVP-TE 信令中分别定义的属性编码。

8.2.1.2 业务创建消息

a) 业务创建请求

业务创建请求被客户设备用来在指定的源和目的客户设备之间请求业务。业务请求还包含描述该业务服务需求相关的一组属性信息。

业务请求发送方向如下：

1) 发起 UNI-C 向 UNI-N 发送业务创建请求；

2) UNI-N 向终结 UNI-C 指示输入的业务请求。

表 6 描述了业务创建请求消息中必选和可选的属性。不是所有的属性都应用在上述情形 1 和情形 2。“应用情形”栏表明相应的属性应用在哪种情形。携带的消息类型表明该属性值由何种消息来携带。

表 6 UNI 接口业务创建请求消息中包含的属性

属性	应用情形	携带的消息类型
源 TNA 地址(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
源逻辑端口标识(M)	发起 UNI-C→UNI-N	连接消息
源通用标签(O)	发起 UNI-C→UNI-N	连接消息
目的 TNA 地址(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
目的逻辑端口标识(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
目的通用标签(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
本地连接 ID(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
合同 ID(O)	发起 UNI-C→UNI-N	呼叫消息
编码类型(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
交换类型(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
流量参数(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
方向属性(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
通用负载标识(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
业务等级(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
分集(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
呼叫名称(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息

b) 业务创建响应

业务创建响应用来通知发起业务请求的 UNI-C 业务已经建立。相关的客户可以开始在建立的连接上进行数据传输。

业务创建响应的发送方向如下：

1) 终结 UNI-C 向 UNI-N 发送用来指示接受或拒绝业务创建请求；

2) UNI-N 向发起 UNI-C 发送用来指示所请求的业务建立成功或建立失败。

消息中所携带的属性如表 7，和上面每种应用相适应。

表 7　UNI 接口业务创建响应消息中包含的属性

属性	应用情形	携带的消息类型
源逻辑端口标识(O)	UNI-N→发起 UNI-C	连接消息
源通用标签(O)	UNI-N→发起 UNI-C	连接消息
目的逻辑端口标识(M-1,O-2)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
目的通用标签(M-1,O-2)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
本地连接 ID(M)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
连接状态(M)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
错误编码(O)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	呼叫/连接消息
呼叫名称(M)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	呼叫消息

注：(M-1,O-2)表示，对于上述情形 1(即，终结 UNI-C→UNI-N)是必选的，对于上述情形 2(即，UNI-N→发起 UNI-C)是可选的。

8.2.1.3　业务确认消息

业务创建确认用来通知业务的目的 UNI-C 业务已经建立。相关的客户可以开始在所建立的连接上进行数据传输。

业务创建确认消息的发送方向如下：

1）　发起 UNI-C 向 UNI-N 通知业务建立完成；

2）　UNI-N 向终结 UNI-C 指示业务已经成功建立，相关的客户可以在所建立的连接上开始传送数据。

消息中所携带的属性信息以及对应的适用范围如表 8。

表 8　UNI 接口业务创建确认消息中包含的属性

属性	应用情形	携带的消息类型
源逻辑端口标识(O)	发起 UNI-C→UNI-N	连接消息
源通用标签(O)	发起 UNI-C→UNI-N	连接消息
目的逻辑端口标识(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
目的通用标签(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
本地连接标识(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
连接状态(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
出错编码(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
呼叫名称(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息

8.2.1.4　业务删除消息

a）　业务删除请求

业务删除请求用来发起对业务的删除。如果业务删除由 UNI-C 发起，则 UNI-C 应该维护连接控制状态，并且在收到连接应答之前，相关的客户应该维持数据平面的连接。这样可以避免连接的另外一端的客户产生告警。终结连接删除请求的 UNI-C 当接收到连接删除请求后可以删除连接(相关的客户

可以删除数据平面连接状态)。

因为下列几种情形,网络内部也可能要删除一条连接(强制删除):

- 内部网络故障,强制网络来终结连接;
- 在一个预先定义的超时时间内 UNI-N 没有收到删除响应。

当网络发起业务删除请求时,不需要业务删除响应,当发起业务删除请求时 UNI-N 可以终结所有业务的控制状态和数据通道。

业务删除请求的发送方向如下:

1) 发起 UNI-C 向 UNI-N 发起业务删除;

2) UNI-N 向终结 UNI-C 指示业务已经被远端删除;

3) UNI-N 向 UNI-C 指示业务已经被网络删除。

消息相关的属性以及适用范围如表 9:

表 9 UNI 接口业务删除消息中包含的属性

属性	应用情形	携带的消息类型
本地连接标识(M)	发起 UNI-C→UNI-N &. UNI-N→终结 UNI-C &. UNI-C→UNI-N	连接消息
呼叫名称(M)	发起 UNI-C→UNI-N &. UNI-N→终结 UNI-C &. UNI-C→UNI-N	呼叫消息

注:业务删除可以由业务的任何一端发起,而不仅仅是由发起业务请求的原始端发起删除。

b) 业务删除响应

业务删除响应消息用来指示业务删除的处理已经完成。当接收到业务删除响应后,发起业务删除的 UNI-C 可以删除控制平面状态(相关的客户可以删除数据平面的业务状态)。

业务删除响应的发送方向如下:

1) 终结 UNI-C 向 UNI-N 发送响应消息作为输入业务删除请求的应答;

2) UNI-N 向发起 UNI-C 发送响应消息来指示业务删除请求已经成功完成。

消息中包含的属性信息以及适用范围如表 10:

表 10 UNI 接口业务删除响应消息中包含的属性

属性	应用情形	携带的消息类型
本地连接标识(M)	终结 UNI-C→UNI-N &. UNI-N→发起 UNI-C	连接消息
连接状态(M)	终结 UNI-C→UNI-N &. UNI-N→发起 UNI-C	连接消息
出错编码(O)	终结 UNI-C→UNI-N &. UNI-N→发起 UNI-C	连接消息
呼叫名称(M)	终结 UNI-C→UNI-N &. UNI-N→发起 UNI-C	呼叫消息

8.2.1.5 业务查询消息

a) 业务状态查询

业务状态查询用来查询给定业务的状态和相关属性。

业务状态查询消息的发送方向如下:

1) 发起 UNI-C 向 UNI-N 查询 UNI-C 拥有的一条或多条连接的状态和/或连接的属性;

2) UNI-N 向两端的 UNI-C 查询 UNI-C 拥有的一条或多条连接的状态和/或连接的属性。

消息属性以及适用情形如表 11：

表 11　UNI 接口业务查询消息中包含的属性

属性	应用情形	携带的消息类型
TNA 地址(O)	发起 UNI-C→UNI-N & UNI-N→UNI-C	呼叫消息
0 个或多个本地连接标识(M)	发起 UNI-C→UNI-N & UNI-N→UNI-C	连接消息
呼叫名称(M)	发起 UNI-C→UNI-N & UNI-N→UNI-C	呼叫消息

单个业务状态查询消息中可以包含多条本地连接 ID。如果只指定 TNA 地址而不指定本地连接 ID，则该 TNA 拥有的所有连接状态都要返回。否则，返回指定业务的状态信息。

注：业务状态查询可以由业务两端的任意一个 UNI-C 来发起，而不仅仅是由发起业务请求的 UNI-C 来发起。

b）　业务状态响应

业务状态响应返回指定业务的状态以及相关的属性信息。

业务状态响应消息的发送方向如下：

1）　UNI-N 向 UNI-C 发送响应消息来指示所请求的业务属性的状态；

2）　UNI-C 向 UNI-N 发送响应消息来指示所请求的业务属性的状态。

消息中所携带的属性以及适用情形列表如下。如果要返回多条连接的状态，表 12 所列出的内容应按每条连接来重复。“本地逻辑端口标识”和“本地通用标签”指示接收到业务状态响应消息的一侧（源或目的）的端口和通道信息。换句话说，该消息不包含远端端口和通道信息。

表 12　UNI 接口业务状态查询响应消息中包含的属性

属性	应用情形	携带的消息类型
本地连接标识(M)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
连接状态(M)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
源 TNA 地址(O)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫消息
目的 TNA 地址(O)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫消息
本地逻辑端口标识(O)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
本地通用标签(O)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
合同 ID(O)	UNI-N→UNI-C	呼叫消息
编码类型(O)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
交换类型(O)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
流量参数(O)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫/连接消息
连接方向(O)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
通用负载标识(O)	UNI-N→UNI-C & UNI-C→UNI-N	连接消息
业务等级(O)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫消息
分集(O)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫消息
出错编码(O)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫/连接消息
呼叫名称(M)	UNI-N→UNI-C & UNI-C→UNI-N	呼叫消息

8.2.1.6 通知消息

通知消息由 UNI-N 自发地向任意一端 UNI-C 上报，用来指示业务状态的改变(不可恢复的业务故障)。消息中所包含的属性信息如表 13。

表 13 UNI 接口通知消息中包含的属性

属性	应用情形	携带的消息类型
本地连接标识(M)	UNI-N→UNI-C	连接消息
连接状态(M)	UNI-N→UNI-C	连接消息
出错编码(M)	UNI-N→UNI-C	呼叫/连接消息
呼叫名称(M)	UNI-N→UNI-C	呼叫消息

8.2.1.7 业务修改消息

a) 业务修改请求

业务修改请求消息用来改变一条存在连接的属性值。

业务修改请求发送方向如下：

1) 发起 UNI-C 向 UNI-N 请求修改一条存在的连接；

2) UNI-N 向终结 UNI-C 指示一个输入的连接修改请求。

业务修改请求消息中所携带的属性以及适用情形如表 14：

表 14 UNI 接口业务修改请求消息中包含的属性

属性	应用情形	携带的消息类型
源 TNA 地址(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
源逻辑端口标识(M)	发起 UNI-C→UNI-N	连接消息
源通用标签(O)	发起 UNI-C→UNI-N	连接消息
目的 TNA 地址(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
目的逻辑端口标识(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
目的通用标签(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
本地连接 ID(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
合同 ID(O)	发起 UNI-C→UNI-N	呼叫消息
编码类型(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
交换类型(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
流量参数(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
方向属性(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
通用负载标识(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
业务等级(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息
分集(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
呼叫名称(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息

b） 业务修改响应

业务修改响应消息用来向 UNI-C 响应其所发起的连接修改请求。相关的客户设备可以用新的连接参数来传输数据。

业务修改响应消息的发送方向如下：

1） 终结 UNI-C 向 UNI-N 发送修改响应消息，以接受或拒绝输入的业务修改请求；

2） UNI-N 向发起 UNI-C 指示业务修改成功或者修改失败。

业务修改请求消息中所携带的属性以及适用情形如表 15：

表 15 UNI 接口业务修改响应消息中包含的属性

属性	应用情形	携带的消息类型
源逻辑端口标识(O)	终结 UNI-C→UNI-N	连接消息
源通用标签(O)	终结 UNI-C→UNI-N	连接消息
目的逻辑端口标识(M-1,O-2)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
目的通用标签(M-1,O-2)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
本地连接 ID(M)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
连接状态(M)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	连接消息
错误编码(O)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	呼叫/连接消息
呼叫名称(M)	终结 UNI-C→UNI-N & UNI-N→发起 UNI-C	呼叫消息

c） 业务修改确认

业务修改确认消息用来通知终结 UNI-C 对业务修改的确认。相关的客户设备可以在所建立的连接上使用新的连接参数来开始数据的传送。

业务修改确认消息的发送方向如下：

1） 发起 UNI-C 向 UNI-N 发送修改确认消息，确认业务修改完成；

2） UNI-N 向终结 UNI-C 指示连接修改成功完成，相关的客户设备可以使用新的连接参数来开始数据的传输。

业务修改确认消息中所携带的属性以及适用情形如表 16：

表 16 UNI 接口业务修改确认消息中包含的属性

属性	应用情形	携带的消息类型
源逻辑端口标识(O)	发起 UNI-C→UNI-N	连接消息
源通用标签(O)	发起 UNI-C→UNI-N	连接消息
目的逻辑端口标识(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
目的通用标签(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
本地连接标识(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
连接状态(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	连接消息
出错编码(O)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫/连接消息
呼叫名称(M)	发起 UNI-C→UNI-N & UNI-N→终结 UNI-C	呼叫消息

8.2.2 UNI 业务属性参数

8.2.2.1 标识相关的属性

a) 传送网络分配的地址(TNA)

TNA 地址是由服务提供者(传送网络)分配给一条或多条数据链路的地址。对于 UNI,TNA 地址可以是 IPv4、IPv6 或 NSAP 地址。

b) 逻辑端口标识

逻辑端口标识是用来指示客户或 TNE 端口的索引。

c) 通用标签

通用标签是用来指示特定数据链路上所携带信号中的复用通道的数据结构。如,该标识用来在 STM-16 链路上指示一个 VC-4 的通道。通常,该数据结构用来指示多个复用等级情形。该数据结构的编码见 8.2.5。

d) 本地连接 ID

本地连接 ID 标识一条 UNI 本地连接。本地连接 ID 应在给定的 UNI 上唯一,但不需要全网唯一。本地连接 ID 在发起端由发起连接请求的 UNI-C 分配,在终结端由 UNI-N 来分配。因此,本地连接 ID 由从发起 UNI-C 到 UNI-N 中的每个连接创建请求消息携带,以及由终结端 UNI-N 到终结端 UNI-C 中的每个连接创建指示消息携带。为同一条连接所产生的连接 ID 值在发起端和终结端可以不同。在发起端,UNI-N 应验证任何 UNI-C 所分配的本地 ID 的唯一性。

8.2.2.2 业务相关的属性

a) 编码类型

编码类型指定了经过 UNI 被编的信号的编码格式。具体编码值见 IETF RFC3471。

b) 交换类型

指示在一条特定链路上用来执行的交换类型。具体编码值见 IETF RFC3471。

c) 方向性

方向属性指示连接是单向还是双向连接,缺省情形下为双向。

d) 通用负载标识

通用负载标识(G-PID)指示所建立的连接用来承载的负载类型(如,标识客户层连接)。G-PID 值的定义见 IETF RFC3471。

e) 服务等级

服务等级属性指示服务的分类情形。运营商可以根据预先定义的特征(如,恢复计划)来指定不同服务等级的范围(如,金级、银级、铜级)。预先定义的服务类型和下列不同的类型相关:网络的恢复类型(如,无恢复,1+1 保护)、连接建立和保持优先级、故障恢复后的连接返回策略以及保持策略。不同服务等级的定义以及他们的缺省值由服务提供商来设置。

8.2.2.3 路由相关的属性

a) 分集

对于一条新创建的连接,该属性指示源于相同 UNI-C 的 n 条存在的连接在传送网络内部有分集路由需求。该属性包含 n 项〈分集类型、本地连接 ID〉,这里,分集类型可以从下列类型列表中选择:

1) 节点分离:新建连接不应该使用被本地连接 ID 所标识的连接路径所经过的任何网络节点;

2) 链路分离:新建连接不应该使用被本地连接 ID 所标识的连接路径所经过的任何网络链路;

3) SRLG 分离:新建连接不应使用被本连接 ID 所标识的连接路径所经过的有相同 SRLG 的任何

链路；

4） 共享路径：新建连接应该使用被本地连接 ID 所标识的连接所使用的相同的链路。

客户所请求的连接可能是 SRLG 分离的，但不是节点分离的。为请求节点和 SRLG 分离，客户应该在分集请求中有两个分离的实体，其中一个为节点分离另外一个为 SRLG 分离。

8.2.2.4 策略相关的属性

合同 ID：该标识由服务提供者来分配并配置给客户。客户不用解析该标识。

8.2.2.5 其他属性

a） 连接状态

用来指示连接的状态，相关值定义如下：

1） 连接活动；

2） 连接不存在；

3） 连接不可用；

4） 连接阻塞。

b） 错误编码

错误编码值用来描述连接活动的出错原因。在连接建立过程中产生的错误被作为 RSVP-TE 协议定义的一部分。另外，下列定义的错误编码和 UNI 相关。

1） 发送者没有授权（策略错误）；

2） 接收者没有授权（策略错误）；

3） 服务等级不可用；

4） 分集属性不可用；

5） G-PID 不支持；

6） 无效的/不明的连接 ID。

8.2.3 UNI RSVP-TE 信令过程

8.2.3.1 UNI 信令相关的机制

a） UNI 接口、信令通道、控制通道、逻辑端口标识、编址

RSVP-TE 消息通过 UNI 信令通道进行交换。信令通道可以通过一条或多条底层的 IP 控制通道来实现。对如何决定 UNI 控制接口以及如何维护 IP 控制通道在 6.1.2 中有描述。连接端点的标识的描述见第 7 章中的描述。

b） 发送 UNI RSVP-TE 消息

当 UNI-C（UNI-N）发送 RSVP-TE 消息时，它应将消息的目的地址设置成对等 UNI-N 或 UNI-C 的地址值。对等实体的节点 ID 用来作为该地址值（见第 7 章的描述）。节点应该使用简单的 IP 封装并且任何 RSVP-TE 消息一定不能包含路由器告警选项。见表 17 中的描述。

如果简单的 IP 封装会导致某些问题，可以使用 GRE 或 IP-in-IP 封装（通过配置）。

表 17 UNI RSVP-TE 消息的 IP 包头中的填充值

传递 UNI RSVP-TE 消息的 IP 包头中的填充值	
IP 协议版本号	4
IP 包头长度	5
TOS	见 IETF RFC 2205 中的定义

表 17（续）

传递 UNI RSVP-TE 消息的 IP 包头中的填充值	
消息包长度	消息长度
标致位	见 IETF RFC 791 中的定义
偏移量	见 IETF RFC 791 中的定义
TTL	≥1
协议号	46
包头校验和	见 IETF RFC 791 中的定义
源地址	UNI-C/UNI-N SCN 的 IP 地址
目的地址	UNI-C/UNI-N SCN 的 IP 地址

c） 接收到 UNI RSVP-TE 消息

成功通过所有的安全验证处理后，UNI-C 或 UNI-N 节点应按 IETF RFC2205 和 IETF RFC3209 的规范定义处理接收到的 RSVP-TE 消息。如果接收到的 RSVP-TE PDU 没有通过安全验证，节点必需丢弃该报文并且一定不能返回 ACK 消息，即使该消息请求 ACK 消息。

d） 可靠的消息交换

为在 UNI 接口上支持可靠的消息交换，UNI-C 和 UNI-N 实现应支持 RSVP-TE 简化刷新的扩展处理（IETF RFC2961）。并且，Path、PathTear、PathErr、Resv、ResvConf 以及 Srefresh 消息中应包含 MESSAGE_ID 对象。在所有的 RSVP-TE 触发消息、PathErr、ResvErr、ResvConf 以及 Srefresh 中的 MESSAGE_ID 对象的 Ack_Desired 标志位应置位。RSVP-TE 的刷新消息中的 MESSAGE_ID 对象的 Ack_Desired 标志位可以置位。

消息标识和应答是基于每一跳的。每个 MESSAGE_ID 对象的包含一个消息标识。该标识应在节点范围内（单个节点 ID 范围内）唯一标识一个消息。

当接收刷新消息的 MESSAGE_ID_ACK 失败后，一定不能删除相关的连接。

注：ACK 消息或 MESSAGE_ID_ACK 对象在本条所描述的时序图中可能没有准确的出现。

e） 连接状态维护

RSVP-TE 通过使用“软状态”来达到维护连接状态的目的。RSVP-TE 软状态有 Path 和 Resv 消息创建并定期刷新。在“清除超时”时间间隔到达后没有收到匹配的刷新消息时，软状态将会删除。状态也可以由显式的 PathTear、带 Path_State_Removed 标志位的 PathErr、ResvTear 消息来删除。

UNI 信令维持软状态，只有收到用户的显式拆除消息后才删除软状态。也就是说，正常情形下，连接删除应该作为显式拆除请求的响应，软状态超时时不应该进行连接删除操作。因此，UNI-C 或 UNI-N 发送状态超时表明系统出现故障。应该上报告警作为状态超时的响应。

控制平面故障一定不能导致对已经建立的连接进行删除。正在处理的连接建立请求可以被删除（既可以在故障发生时删除，也可以在故障恢复后删除）。已经建立的连接收到阻塞的连接释放请求后，应释放该连接（可以在故障发生期间释放，也可以在故障恢复后释放）。

即使用了通过 MESSAGE_ID 和 MESSAGE_ID_ACK 对象实现的可靠消息交换机制，也不应该省略连接状态的刷新过程。为减少刷新消息的数量，节点应该使用总结刷新（Srefresh）消息 IETF RFC2961 来刷新连接状态。

f） 预约风格

RSVP-TE 预约风格在 IETF RFC2205 中有定义。对于固定带宽业务，UNI 信令使用 FF（Fixed Filter）风格。对于可修改带宽的业务，使用 SE（Shared Explicit）风格。

g） 本地连接标识

本地连接标识用来唯一标识一条 UNI 连接。RSVP-TE 信令中是通过 Path 和 PathErr 消息中的 UNI_IPv4_SESSION、LSP_TUNNEL_IPv4_SENDER_TEMPLATE 对象组合或 Resv、ResvTear、ResvErr 和 ResvConf 消息中的 UNI_IPv4_SESSION、LSP_TUNNEL_IPv4_FILTER_SPEC 对象的组合来实现的。在连接的生命周期内本地连接标识维持不变。

h） 连接流量参数

连接的流量参数依赖具体的技术并在 SENDER_TSPEC 和 FLOWSPEC 对象中编码。UNI-N 和 UNI-C 节点应支持 GMPLS-SDH 中定义的标准的 SDH 流量参数，并覆盖 8.2 中的相关定义。当网络不支持连接所请求的流量参数时，网络应产生 PathErr 消息并指示出错原因为“流量控制出错/业务不支持”（Traffic Control Error/ Service Unsupported）。

8.2.3.2 UNI 呼叫处理过程

呼叫是网络边缘节点之间的一种关联关系，一个呼叫用来支持一个用户业务实例。呼叫本身不提供对用户业务的连通性，仅仅只用来构建后续一个或多个 LSP 之间的关联关系。呼叫反映网络边缘节点之间的一种合约关系。呼叫的建立需要包括：策略验证、鉴权、网络安全认证以及边缘节点之间的能力协商等处理。呼叫可以用来管理一组连接，这组连接共同为客户提供端到端的网络连接服务。

连接相关的状态信息需要在该连接所经过的每一个网络节点中维护。和连接相比，呼叫相关的状态信息只需要在网络边缘节点中维护，连接所经过的网络中间节点不需要处理呼叫相关的信息。

呼叫用来完成以下几个功能：

a） 在 LSP 连接建立之前用来标识和验证呼叫的发起者；

b） 支持具有差异化路由的多条 LSP，这些 LSP 可以共同构成一个虚级联业务；

c） 支持在单个呼叫中关联多条 LSP；

d） 通过允许相关 LSP 运行状态的改变来简化控制平面的操作。

OIF UNI 中的呼叫功能继承了 ITU-T G.8080 和 ITU-T G.7713 中的定义。呼叫功能由呼叫控制器完成，分为两种呼叫控制器：

a） 主叫呼叫控制器（Calling Call Controller）、被叫呼叫控制器（Called Call Controller）

b） 网络呼叫控制器（Network Call Controller）

主叫呼叫控制器通过一个或多个网络呼叫控制器与被叫呼叫控制器进行交互。

在 UNI 中，源 UNI-C 为主叫呼叫控制器，宿 UNI-C 为被叫呼叫控制器，UNI-N 为网络呼叫控制器。

呼叫建立的步骤如下：

——源 UNI-C 发起到目的 UNI-C 的呼叫建立请求。源 UNI-N 验证呼叫请求的有效性（包括授权和一致性验证以及验证是否满足策略相关的约束条件）。随后向目的 UNI-N 方向发送网络呼叫请求；

——目的 UNI-N 接收到网络呼叫请求后，根据策略配置信息来验证该呼叫请求是否能够接受，并根据源 UNI-N 和所请求业务的相关的能力信息来验证目的 UNI-N 支持业务连接的能力。验证成功后向目的 UNI-C 发送呼叫指示。目的 UNI-C 验证呼叫请求并向目的 UNI-N 发送呼叫应答；

——目的 UNI-N 收到呼叫应答后向源 UNI-N 发送呼叫应答消息，源 UNI-N 向源 UNI-C 转发呼叫应答；

——当 UNI-C 接收到肯定的呼叫应答后，发起呼叫相关连接的建立；

——当连接成功建立后，呼叫建立完成，可进行用户业务的传送；

——在上述呼叫和连接的处理过程中，任意一段（包括：源 UNI-C 到源 UNI-N、源 UNI-N 到目的

UNI-N、目的 UNI-N 到目的 UNI-C)的呼叫或连接请求失败,则整个呼叫过程失败,并要向源 UNI-C 用户发送呼叫被拒绝的通告消息,同时释放已经分配的网络资源。

UNI-C 和 UNI-N 都可以发起呼叫释放请求。UNI-N 验证呼叫释放请求,已通过验证的呼叫释放请求应总能够成功释放呼叫。在呼叫释放完成之前,应释放所有的和该呼叫相关的连接。在呼叫释放过程中的任何故障情况都要上报给管理系统,包括报告关于部分连接没有释放的信息,后续的处理主要防止再次使用没有被成功释放连接。呼叫释放的步骤如下:

——收到呼叫释放请求后,UNI-N 验证呼叫释放请求。其中包括授权和一致性验证以及验证是否满足策略相关的约束条件;

——当验证成功后,呼叫释放请求随后发起连接释放请求。连接释放请求的处理过程见 8.1.3.6,网络连接的释放在呼叫释放完成之前进行。如果呼叫对应多个连接,将会释放所有对应的连接;

——当源 UNI-C 收到连接释放成功响应消息后,呼叫释放成功完成。

8.2.3.3 UNI 呼叫修改过程(可选)

有两种方式支持 UNI 的呼叫修改:通过在呼叫中增加删除连接,通过改变呼叫中已有连接的参数。第一种方式用于 TDM 业务(SDH/SONET,OTN),第二种方式一般用于以太网业务。

a) 通过增加删除连接修改呼叫

在呼叫中增加删除连接可用于修改呼叫的带宽。由于每个连接都有独立的状态,增加或删除连接失败不会影响呼叫中已有的连接。

当进行新的呼叫和连接建立时,源 UNI-C 发往源 UNI-N 的路径建立(Path)消息中携带空的呼叫 ID,由源 UNI-N 分配呼叫 ID,源 UNI-C 收到 Resv 消息后,保存分配到的呼叫 ID。后续向呼叫中增加连接时,UNI-C 发送的路径建立(Path)消息中携带已有的呼叫 ID,从而在 UNI-C 和 UNI-N 处将连接和呼叫关联。如果路径建立(Path)消息中的非空呼叫 ID 无效,或不存在对应的状态,UNI-N 会向 UNI-C 返回 PathErr 消息,错误码为无效/未知呼叫 ID(Invalid/unknown CALL_ID)。

呼叫中的连接互相独立,从呼叫中删除一条连接对其他连接不产生影响。连接的删除可从源 UNI-C、UNI-N,宿 UNI-C、UNI-N 或网络中间发起。OIF 中不支持没有连接的呼叫,呼叫中的最后一条连接被删除将导致呼叫的删除。

b) 通过修改连接修改呼叫

可通过对呼叫中某条连接带宽的修改完成对呼叫带宽的修改。对于连接的修改请参考 8.2.3.5。

8.2.3.4 UNI 连接创建过程

为创建一条连接,UNI-C 节点向相邻的 UNI-N 节点发送路径建立(Path)消息。路径建立(Path)消息中应该包含一个 GENERALIZED_LABEL_REQUEST 对象,该对象用来指示连接需要绑定一个标签值。连接的流量特性参数被编码到路径建立(Path)消息中 SENDER_TSPEC 对象以及相关资源预留(Resv)消息的 FLOWSPEC 对象中。图 16 描述了连接成功建立的时序图。

路径建立(Path)消息中的 IPv4_IF_ID_RSVP-TE_HOP 对象指定了连接建立时的输出接口。Resv 消息中的 GENERALIZED_LABEL 对象指定了为连接使用而在该接口上分配的标签。

为请求一条双向连接,UNI-C 应在路径建立(Path)消息中插入 UPSTREAM_LABEL 标签并为连接选择上游方反向入口标签。

如果节点需要 GENERALIZED_LABEL 和 UPSTREAM_LABEL 对象具有相同的标签值,路径建立(Path)消息中应该包含 LABEL_SET 对象,以便将 GENERALIZED_LABEL 的对象标签值限定在和 UPSTREAM_LABEL 对象标签值相同。对于 SDH 连接这种情形是典型的配置。

可以假设的是,只有当源 UNI-N 向源 UNI-C 发送资源预留(Resv)消息时,才表明传送网络内部的

连接分段已经建立。因此，为避免数据丢失，在源 UNI-C 收到资源预留消息之前，源客户不应该启动数据的传输。目的客户可以选择在资源预留消息中插入 RESV_CONFIRM 对象。这种情形下，在目的 UNI-C 收到资源预留确认(ResvConf)消息之前，目的客户不应该启动数据的传送。当目的 UNI-C 发送资源预留消息之前，目的客户应该准备接收数据。当源的 UNI-C 发送路径建立消息之前，源客户应该准备接收数据。如果节点监控一条连接，在端到端的连接建立完成并验证通过后才能上报业务引发的告警信息。见图 16 中的虚线所示。

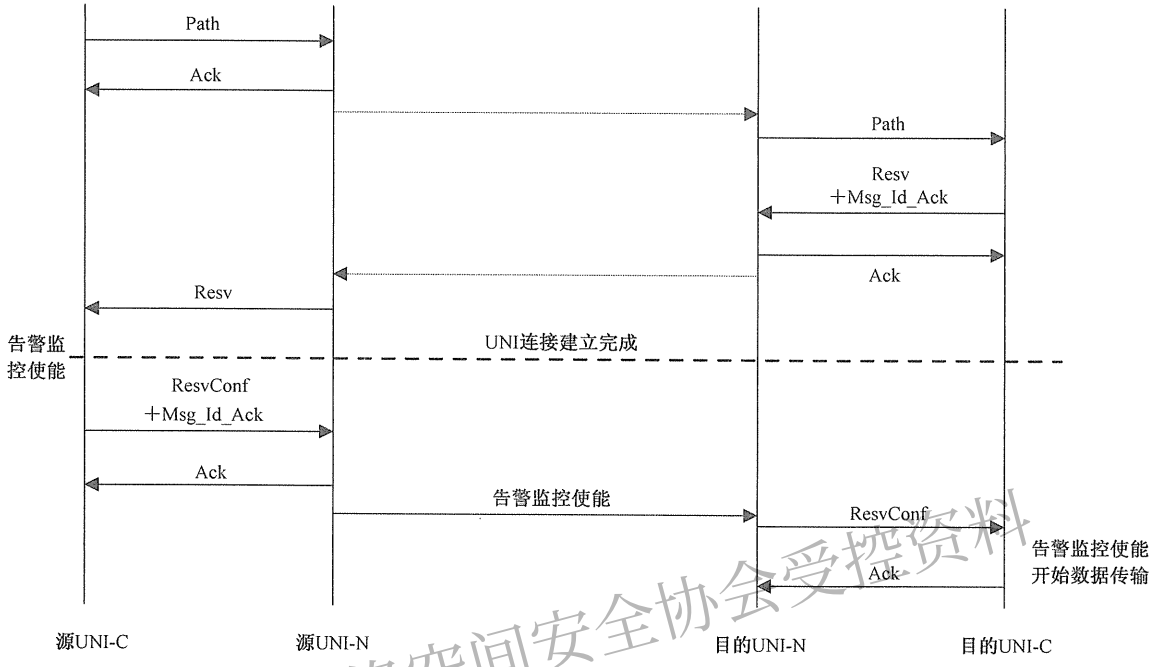


图 16　UNI 连接成功创建时序图

两条连接创建请求之间可能会发生标签的竞争，竞争的解决处理见 IETF RFC3473 中的描述。

由于资源不可用、策略或可达性约束等可能导致连接建立失败。图 17、图 18、图 19 描述了连接建立失败的时序图。

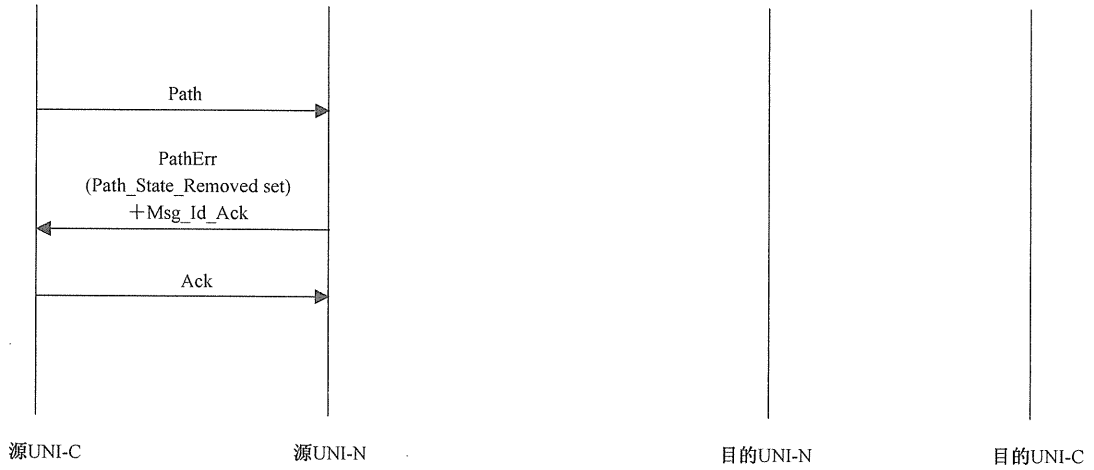


图 17　UNI 连接创建请求被源 UNI-N 拒绝而失败，并携带 Path_State_Removed 标志

拒绝连接请求的节点应该删除它自身的路径状态信息并在路径建立出错(PathErr)消息中设置 Path_State_Removed 标志位后返回给上游节点。当上游节点收到带有 Path_State_Removed 标志位的路径建立出错(PathErr)消息后，不应该向下游节点发送路径删除(PathTear)消息。见图 17 和图 19 中

的描述。

如果 Path_State_Removed 标志位在路径建立出错(PathErr)消息中没有被置位,源 UNI-C 应确定是否显式删除连接或允许它超时。为显式地删除一条连接,UNI-C 应向下游方向发送路径删除(PathTear)消息。接收到路径删除(PathTear)消息的节点如果没有找到匹配的路径状态,如果路径删除(PathTear)消息中携带的 MESSAGE_ID 对象中的 Ack_Desired 标志位被置位时,则应应答消息。然后忽略该路径删除(PathTear)消息。见图 18 中的描述。

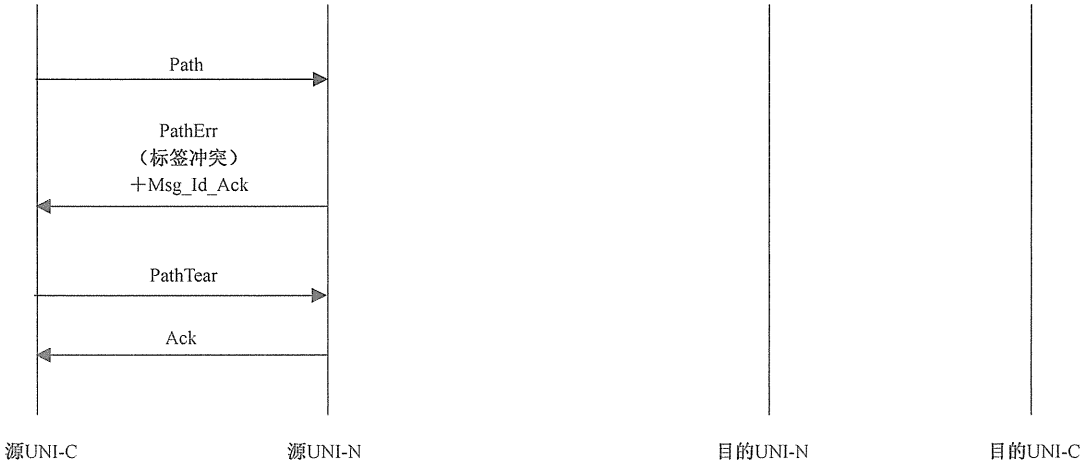


图 18 UNI 连接创建请求被源 UNI-N 拒绝而失败,不携带 Path_State_Removed 标志

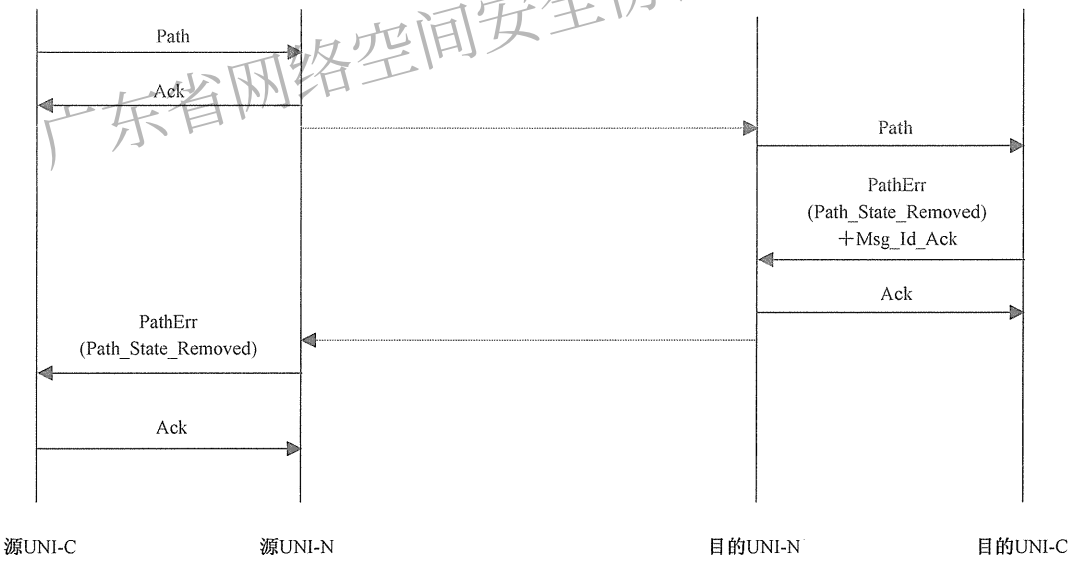


图 19 UNI 连接创建请求被目的 UNI-C 拒绝而失败,并携带 Path_State_Removed 标志

8.2.3.5 UNI 连接修改过程

UNI 不支持破坏性的连接修改。但应该支持在不改变流量参数的情形下对 RSVP-TE 对象进行修改,如 ADMIN_STATUS 对象和 MESSAGE_ID 对象等。

采用 IETF RFC3209 中 2.5 定义的“先建后拆”(Make-Before-Break)方式可实现以太网连接的无损带宽修改。

8.2.3.6 UNI 连接删除过程

RSVP-TE 当前的连接删除是使用单方向的路径删除(PathTear)消息,或 ResvTear 和 PathTear 消息的合并。当接收到路径删除(PathTear)消息后,节点删除连接状态信息并转发该消息。在光网络中,节点删除一条连接时(如,删除交叉连接),下游节点会认为是该连接发生故障(如,帧丢失、光信号丢失等),这可能随后会导致管理告警并可能会触发对该连接的保护恢复。

为处理这种情形,应实现正常的连接删除处理。这种处理中,应在所发送的 Path 或 Resv 消息中携带 ADMIN_STATUS,并沿着连接所经过的路径在实际删除连接之前通知所有节点准备删除该连接。该处理的描述见 IETF RFC3473,如图 20、图 21 所示。

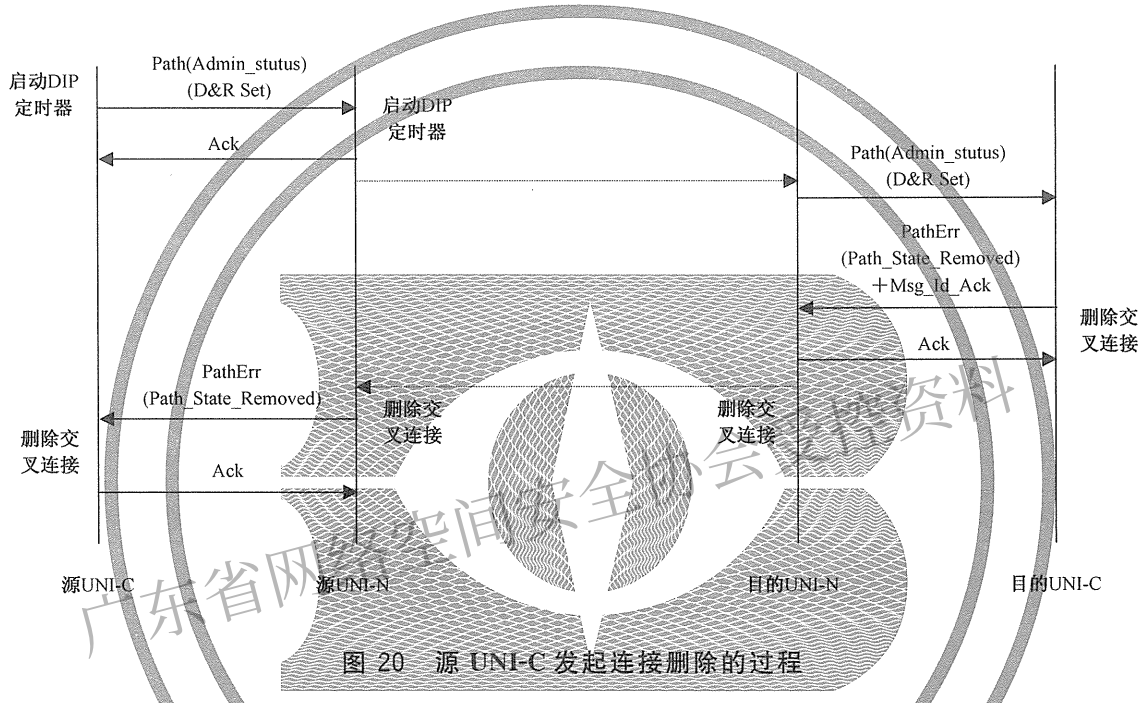


图 20 源 UNI-C 发起连接删除的过程

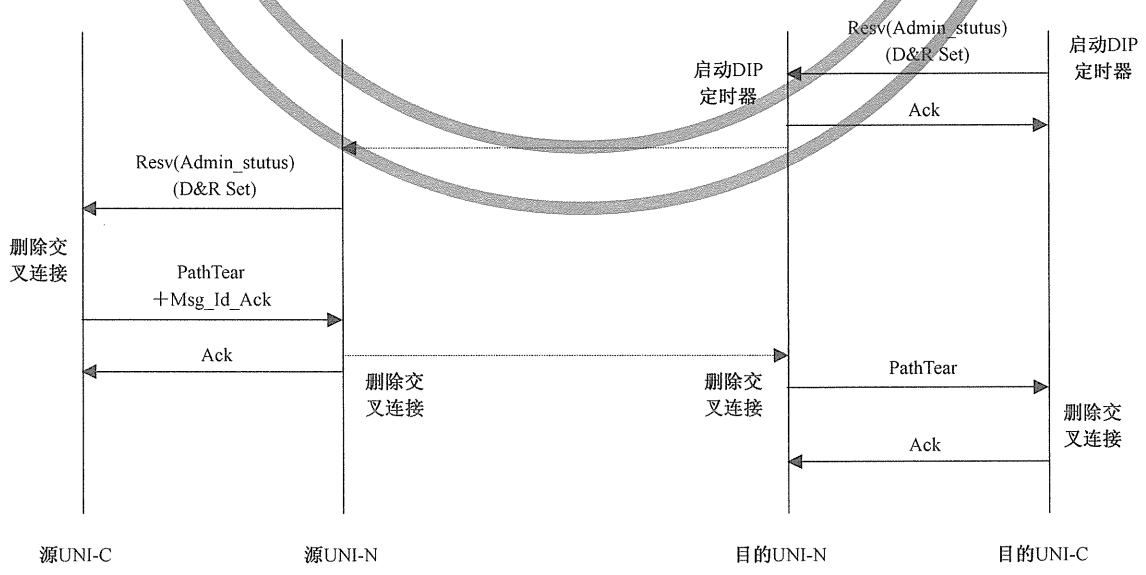


图 21 目的 UNI-C 发起连接删除的过程

UNI-N 可以通过发送 Resv 或 Path 消息到 UNI-C 来触发连接删除，其中消息中保护 ADMIN_STATUS 对象并且对象的 A&R 比特被置位。见图 22 和图 23 的描述。接收到该消息的 UNI-C 应该发起连接删除处理。如果 UNI-C 不通过发起正常删除来响应网络时，网络应该持续如图 22 和图 23 中虚线以下的消息流。

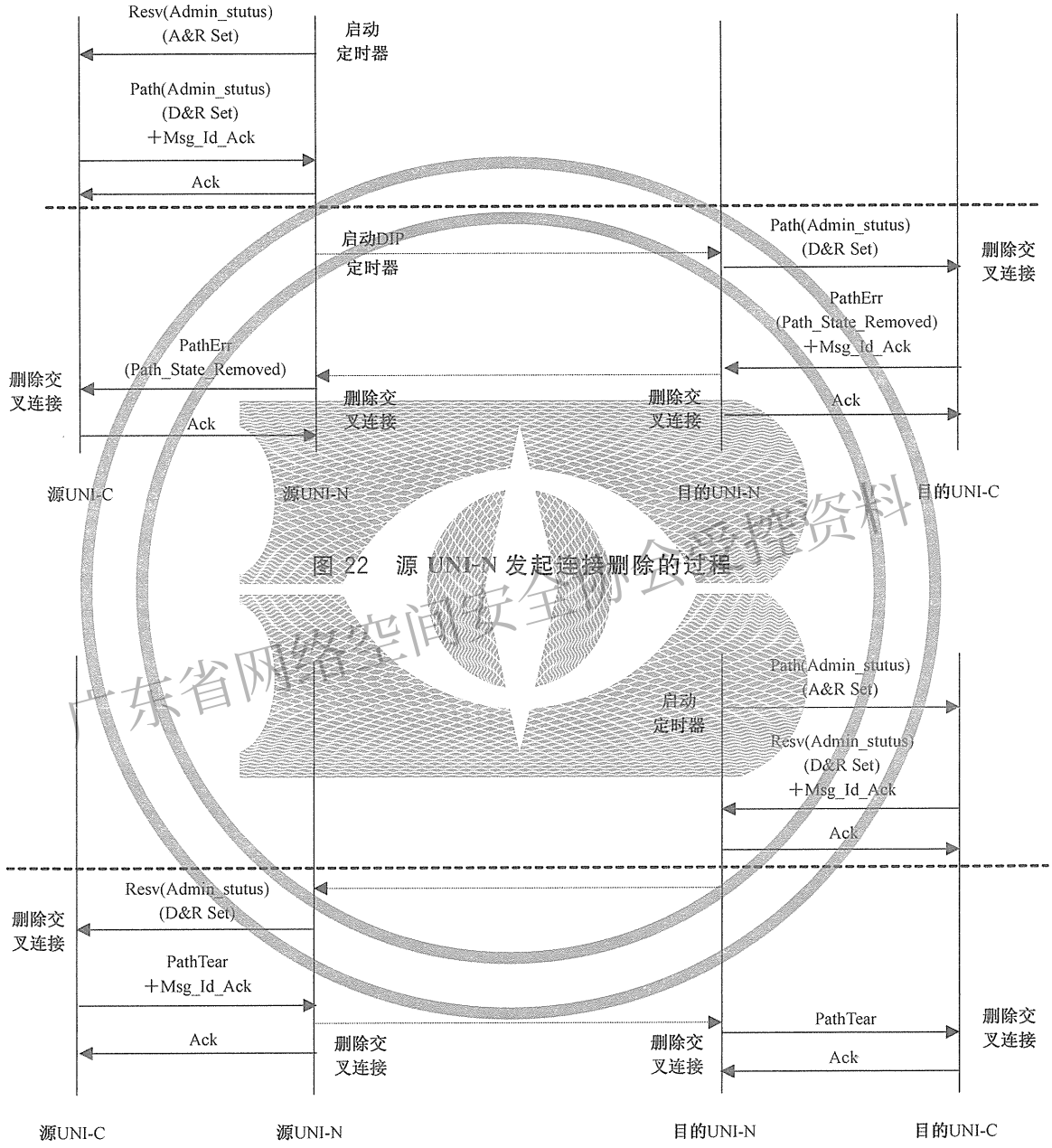


图 22　源 UNI-N 发起连接删除的过程

图 23　目的 UNI-N 发起连接删除的过程

强制删除用来处理需要单向删除的情形。通常是因为网络发生某种事件需要删除连接。如：

a）　网络内部故障，要强制网络终结连接；

b）　当 ADMIN_STATUS 对象中的“删除正在进行中”定时器超时。

如图 24 所示，UNI-N 通过向源 UNI-C 发送 PathErr，同时，UNI-N 向目的 UNI-C 发送 PathTear 消息来强制删除连接。其中发往源 UNI-C 的 PathErr 消息中的“Path_State_Removed”消息置位用来

指示路径状态已经被删除。由 PathErr 消息携带“Path_State_Removed”标志位来发起的强制删除可以通过使用 ResvTear 和 PathTear 消息的组合来替代。UNI-C 节点只有当“删除处理正在进行中”(Deletion In Progress)定时器超时后或软状态超时后才发起强制删除。

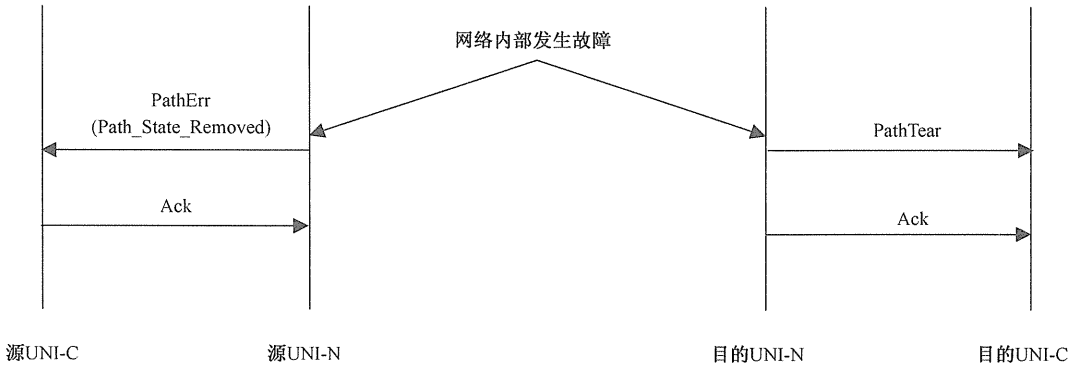


图 24　网络内部故障引发的强制删除过程

8.2.3.7　信令通道故障以及信令节点故障的恢复处理

RSVP-TE 协议中,和数据链路相关的信令消息在相同的链路上发送,链路或控制平面故障会导致对链路资源的预约进行删除。但是在 UNI 信令中,控制平面和数据平面分离,控制平面发生故障并不意味着传送平面也发生了故障。控制平面状态故障的处理(转发状态没有丢失)以及 UNI-C 到 UNI-N 控制通道故障的处理见 IETF RFC3473 中的描述,它是通过使用 Hello 消息并支持 RESTART_CAP 对象来实现的。这里要特别强调的是,信令通道或控制协议实体故障一定不能导致对已经建立的连接进行删除。

为满足该需求,节点应支持如 IETF RFC3473 中第 9 章所描述的故障处理,见图 25 和图 26 的描述。图 25 描述了节点从故障中进行恢复的消息流程(控制平面完全故障)。图 26 描述了相邻节点之间信令通道故障恢复的消息流程(信令协议实体没有发生故障),这种恢复处理流程是以 Srefresh 消息为基础的。

图 25 和图 26 的“自刷新处理”是指 UNI-C 或 UNI-N 节点在故障发生期间的行为和周期性从邻接点接收到 RSVP-TE 刷新消息时的行为一样。当 RSVP-TE 邻接重新建立或重启定时器超时之后,UNI-C 或 UNI-N 会停止自刷新处理。在实际运行的网络中,为防止出现重启超时而恢复还没有完成导致已有 LSP 被删除的异常情况,建议将重启定时器时间设置足够大,如 Hello 消息中重启能力(RESTART_CAP)对象的重启定时器(RESTART_TIME)域设置为 0xFFFFFFFF。

8.2.3.8　传送平面的故障和恢复处理

传送网络提供了丰富的故障检测手段,特别是可以利用传送网络中的开销字节来检测传送层故障。检测到故障的节点可以尝试进行故障恢复。

在网络内部为允许不同的保护和恢复机制,当检测到故障后节点不应该删除连接,而是应该继续接收和发送 RSVP-TE 刷新消息直到接收到显式的拆除消息或连接状态超时。

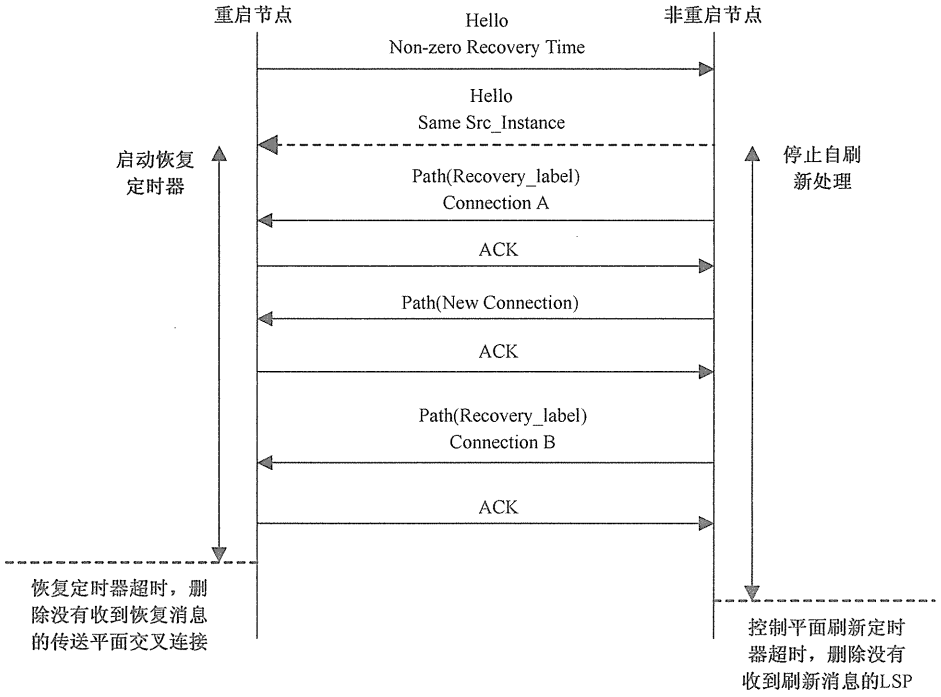


图 25　UNI 节点故障恢复流程图

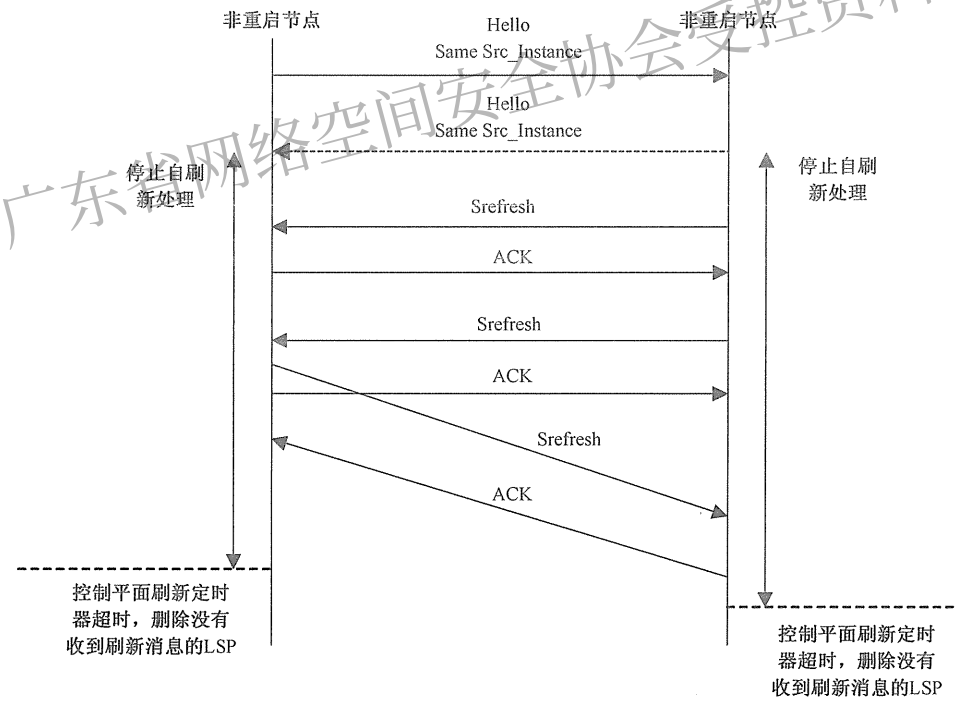


图 26　UNI 控制通道故障恢复流程图

8.2.4　UNI 信令的 RSVP-TE 消息和对象

8.2.4.1　UNI 抽象消息和 RSVP-TE 消息之间的映射

UNI 抽象消息大部分都能和现有的 IETF 扩展定义的 RSVP-TE 消息相对应。对应关系如表 18 所示：

表 18　UNI 抽象消息和 RSVP-TE 消息的映射

消息编号	UNI 抽象消息	RSVP 消息	消息类型
1	业务创建请求	Path	呼叫请求消息
		Path	连接创建请求消息
2	业务创建响应	Resv	呼叫响应消息
		Resv,PathErr	连接创建响应消息
3	业务创建确认	ResvConf	呼叫创建确认消息
		ResvConf	连接创建确认消息
4	业务释放请求	Path or Resv with ADMIN_STATUS "Deletion in Progress" bit	连接释放请求消息
5	业务释放响应	PathErr with Path_State_Removed flag,PathTear	连接释放响应消息
6	业务状态查询	隐含	
7	业务状态查询响应	隐含	
8	业务通告	Notify	呼叫故障通告
		PathErr,ResvErr	连接故障通告
9	业务修改请求	Path	呼叫修改请求消息
		Path	连接修改请求消息
10	业务修改响应	Resv	呼叫修改响应消息
		Resv,PathErr	连接修改响应消息
11	业务修改确认	ResvConf	呼叫修改确认消息
		ResvConf	连接修改确认消息

8.2.4.2　UNI RSVP-TE 信令消息定义

本节描述了和 UNI RSVP-TE 信令相关的消息以及 OIF UNI 中新增的对象定义。消息和对象的具体用法请参考 OIF-UNI-1.0-R2-RSVP 和 OIF-UNI-2.0-RSVP。在下列消息中，ResvErr 和 ResvTear 消息为可选消息，其他消息为必选消息。在对消息的描述中，“[ ]”中所包含的对象为该消息中的可选对象，其他对象为必选对象。IETF 的 UNI 信令过程和 OIF 有较大差异，例如 IETF 中的呼叫建立维护通过 Notify 消息完成，而 OIF 中的呼叫建立和维护与连接的建立和维护绑定在一起，通过 Path，Resv 等消息完成。且不支持一个呼叫下面没有连接。

a)　RSVP-TE 通用消息头

RSVP-TE 公用头中的标志位域应置 1，用来指示节点支持 Bundle 和 Srefresh 消息。

b)　Hello 消息(Msg Type = 20 [IETF RFC3209])

Hello 消息用作节点的故障检测。有如下的格式：

〈Hello message〉::= 〈Common Header〉 [〈INTEGRITY〉]
　　[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉] ...]
　　〈HELLO〉
　　〈RESTART_CAP〉

Hello 消息在相邻的 UNI 信令对等实体之间周期性重传。重传间隔应该通过管理进行配置。缺省值为 5 s。

c) Path 消息(Msg Type = 1 [IETF RFC2205])

Path 消息用于连接创建,呼叫和连接修改,呼叫优雅删除以及故障恢复。Path 消息的格式如下:

〈Path Message〉::= 〈Common Header〉
[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈UNI_IPv4_SESSION〉〈IPv4_IF_ID_RSVP-TE_HOP〉
〈TIME_VALUES〉
〈GENERALIZED_LABEL_REQUEST〉
〈CALL ID〉
[〈LABEL_SET〉...]
[〈ADMIN_STATUS〉]
〈Generalized UNI〉
[〈POLICY_DATA〉...]
〈sender descriptor〉

〈Generalized UNI〉::= 〈Common Object Header〉
〈DESTINATION_TNA〉
〈SOURCE_TNA〉
[〈DIVERSITY〉...]
[〈SERVICE_LEVEL〉]
[〈EGRESS_LABEL〉]

单向连接时发送者描述的格式如下:

〈sender descriptor〉::=
〈LSP_TUNNEL_IPv4_SENDER_TEMPLATE〉〈XXX_SENDER_TSPEC〉
[〈RECOVER_LABEL〉]

双向连接时发送者描述的格式如下:

〈sender descriptor〉::=
〈LSP_TUNNEL_IPv4_SENDER_TEMPLATE〉〈XXX_SENDER_TSPEC〉
〈UPSTREAM_LABEL〉[〈RECOVER_LABEL〉]

其中 XXX_SENDER_TSPEC 表示根据所请求的连接类型不同使用不同的流量参数格式。如,对于 SDH 类型的连接请求,表示 SDH SENDER_TSPEC(见,IETF RFC4606);对于 OTN 类型的连接请求,表示 G.709 SENDER_TSPEC(见 IETF RFC4328);对于 Ethernet 类型的连接请求,表示 Ethernet SENDER_TSPEC(见,draft-ietf-ccamp-ethernet-traffic-parameters)。

d) PathErr 消息(Msg Type = 3 [IETF RFC2205])

PathErr 消息用来上报错误并可以用做连接删除

UNI PathErr 消息格式定义如下:

〈PathErr message〉::= 〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉

〈UNI_IPv4_SESSION〉
〈CALL ID〉
〈IPv4_ERROR_SPEC〉
[〈ACCEPTABLE_LABEL_SET〉]
[〈POLICY_DATA〉…]
〈sender descriptor〉

e) PathTear 消息(Msg Type = 5 [IETF RFC2205])

当源 UNI-C 要删除一条连接时,使用 PathTear 消息。

UNI PathTear 消息格式如下

〈PathTear Message〉::= 〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉|〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈UNI_IPv4_SESSION〉
〈CALL ID〉
〈IPv4_IF_ID_RSVP-TE_HOP〉
〈sender descriptor〉

〈sender descriptor〉::=(见上面的定义)

f) Resv 消息(Msg Type = 2 [IETF RFC2205])

Resv 消息可用作连接创建,呼叫和连接的修改。当目的 UNI-C 接收到一个匹配的 ResvConf 消息后,应该停止在 Resv 消息中插入 ResvConfirm 对象。

UNI Resv 消息格式定义如下:

〈Resv Message〉::= 〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉|〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈UNI_IPv4_SESSION〉〈 IPv4_IF_ID_RSVP-TE_HOP 〉
〈TIME_VALUES〉
〈CALL_ID〉
[〈IPv4_RESV_CONFIRM〉]
[〈ADMIN_STATUS〉]
[〈POLICY_DATA〉...]
〈STYLE〉
〈FF flow descriptor〉

其中,

〈FF flow descriptor〉::=
〈 XXX_FLOWSPEC〉〈LSP_TUNNEL_IPv4_FILTER_SPEC〉
〈GENERALIZED_LABEL〉

上述 XXX_ FLOWSPEC 表示根据所请求的连接类型不同使用不同的流量参数格式。如,对于 SDH 类型的连接请求,表示 SDH FLOWSPEC(见,IETF RFC4606);对于 OTN 类型的连接请求,表示 G.709 FLOWSPEC(见 IETF RFC4328);对于 Ethernet 类型的连接请求,表示 Ethernet FLOWSPEC(见,draft-ietf-ccamp-ethernet-traffic-parameters)。

g) ResvConf 消息(Msg Type = 7 [IETF RFC2205])

ResvConf 消息应该从源 UNI-C 向下游方向发送,用来通知 RESV_CONFIRM 对象中所包含的请

求确认的节点来指示源节点已经接收到资源预留消息(Resv)。在 UNI 中,ResvConf 消息从源 UNI-C 发送到源 UNI-N,并从目的 UNI-N 发送到目的 UNI-C。此时 Resv 消息中可以不使用 RESV_CONFIRM 对象。但,节点如果接收到 RESV_CONFIRM 对象则应产生 ResvConf 消息作为应答。网络应将 ResvConf 消息从源 UNI-N 传递到目的 UNI-N。

〈ResvConf message〉::= 〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈UNI_IPv4_SESSION〉〈IPv4_ERROR_SPEC〉
〈IPv4_RESV_CONFIRM〉
〈STYLE〉〈FF flow descriptor 〉

h) ResvErr 消息(Msg Type = 4 [IETF RFC2205])(可选)

ResvErr 消息用来上报预约错误。UNI ResvErr 消息格式定义如下:

〈ResvErr message〉::=〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈UNI_IPv4_SESSION〉〈IPv4IF_ID_RSVP-TE_HOP〉
〈IPv4_ERROR_SPEC〉
[〈ACCEPTABLE_LABEL_SET〉]
[〈POLICY_DATA〉...]
〈STYLE〉〈FF flow description〉

i) ResvTear 消息(Msg Type = 6 [IETF RFC2205])(可选)

UNI 支持的 ResvTear 消息和 RSVP-TE 协议兼容。UNI ResvTear 消息定义如下:

〈ResvTear Message〉::=〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈UNI_IPv4_SESSION〉〈IPv4_IF_ID_RSVP-TE_HOP〉
〈STYLE〉
〈FF flow description 〉

〈FF flow description 〉::=(见上面的定义)

j) Srefresh 消息(Msg Type = 15 [IETF RFC2961])

UNI Srefresh 消息格式定义如下

〈Srefresh message〉::= 〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
〈MESSAGE_ID〉
〈srefresh list〉

〈srefresh list〉::= 〈MESSAGE_ID_LIST〉
[〈srefresh list〉]

k) Notify 消息(Msg Type = 21 [IETF RFC3473])

OIF 中的 Notify 消息用于从源 UNI-N 通告源 UNI-C,指示优雅删除因由源 UNI-C 发起。Notify 消息格式如下:

〈Notify message〉::= 〈Common Header〉[〈INTEGRITY〉]
[[〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]

〈MESSAGE_ID〉
〈ERROR_SPEC〉
〈notify session lists〉

〈notify session lists〉::= [〈notify session lists〉]
〈upstream notify session〉

〈upstream notify session〉::= 〈SESSION〉
〈CALL_ID〉
〈ADMIN_STATUS〉
〈sender descriptor〉

8.2.4.3 UNI RSVP-TE 对象定义

a) LSP_TUNNEL_IPv4_SENDER_TEMPLATE Object(Class-Num = 11 [IETF RFC3209])

LSP_TUNNEL_IPv4_SENDER_TEMPLATE 对象格式见 IETF RFC3209 中的定义。对于 UNI，IPv4 隧道发送者地址值在源 UNI 接口应设置成源 UNI-C 的节点 ID，在目的 UNI 接口应设置成目的 UNI-N 的节点 ID。LSP ID 由 Path 消息的发送者分配并在连接的生命周期内维持不变。

LSP_TUNNEL_IPv4_SENDER_TEMPLATE 对象和 UNI_IPv4_SESSION 对象组合应能唯一标识一条本地 UNI 连接。

b) UNI_IPv4_SESSION 对象(Class-Num = 1 [IETF RFC3209])

UNI_IPv4_SESSION 对象格式如图 27 所示。

UNI_ IPv4_SESSION 对象：Class = 1，C-Type = 11

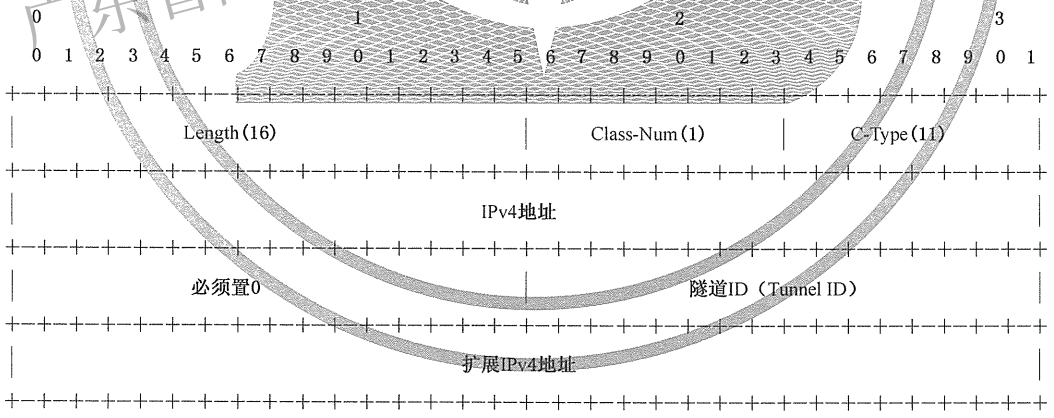


图 27 UNI IPv4 会话对象格式

IPv4 地址：在源 UNI 应设置源 UNI-N 的节点 ID。在目的 UNI 应设置目的 UNI-C 的节点 ID；

隧道 ID：16 位标识，由 Path 消息的发送者分配。在连接的生命周期内 ID 值保持不变。如果源/目的 TNA 地址不是描述的一部分时，在相同的 UNI-C 和 UNI-N 之间单跳的 LSP 只能通过隧道 ID 来进行区别。

扩展 IPv4 地址：源 UNI 应设置源 UNI-C 的节点 ID，在目的 UNI 应设置目的 UNI-N 的节点 ID。LSP_TUNNEL_IPv4_SENDER_TEMPLATE 和 UNI_IPv4_SESSION 对象的组合应能够唯一标识一条 UNI 本地连接并在连接的生命周期内保持不变。无法识

别的连接 ID 应该导致错误消息的产生，错误编码为“路由问题：无效的/未知的连接 ID”。

c) GENERALIZED_UNI 对象(Class-Num＝229)

通过 GENERALIZED_UNI 对象来指定 UNI 的具体属性。GENERALIZED_UNI 对象格式如图 28 所示：

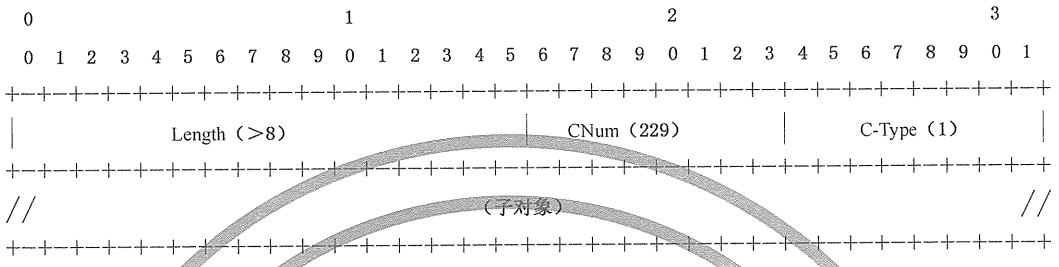


图 28 通用 UNI 对象格式

子对象：

GENERALIZED_UNI 对象的内容由一系列的变长数据项构成。子对象的通用格式如图 29 所示：

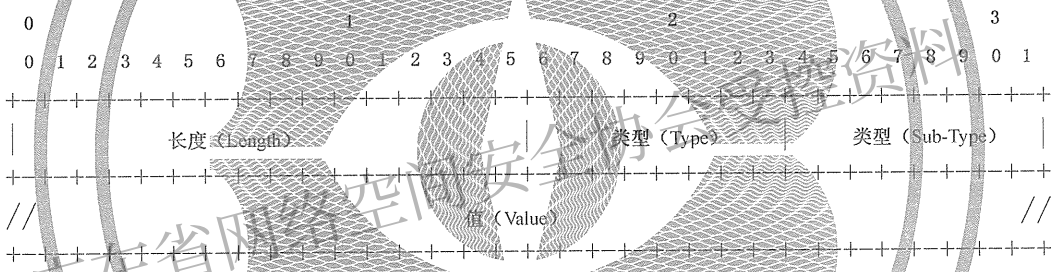


图 29 通用 UNI 对象中的子对象格式

为了将来的兼容，根据 RSVP-TE 对象的规则为子对象分配类型和子类型，但从自己的编号空间进行分配。以后的类型和子类型的处理方法和 RSVP-TE 处理类型和子类型相同。如果因为无法识别的类型或子类型，节点应该使用的出错编码为“类型未知”或“类型已知但子类型未知”，错误值应该设置成 GENERALIZED_UNI 对象的类型和子类型值。

- 源 TNA 地址子对象(Type＝1)

源 TNA 地址子对象包含源 UNI-C 的 TNA 地址。有以下三种格式：IPv4 或 IPv6。如果 GENERALIZED_UNI 对象包含多个源 TNA 地址子对象，只有第一个有意义其他的应忽略。

源 IPv4 TNA 地址子对象格式如图 30 所示：

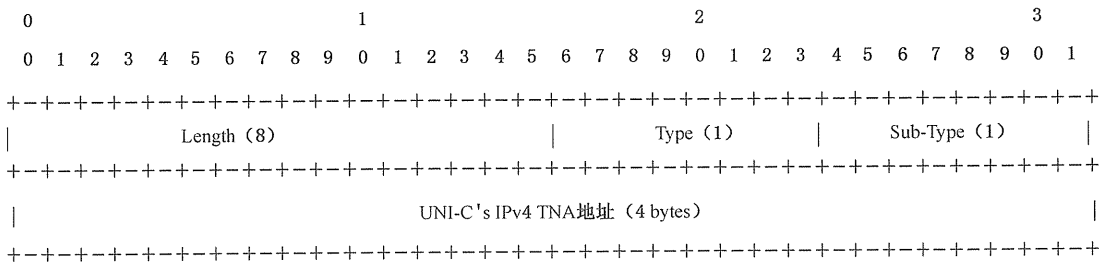


图 30 源 TNA IPv4 地址子对象

源 IPv6 TNA 地址子对象格式如图 31 所示：

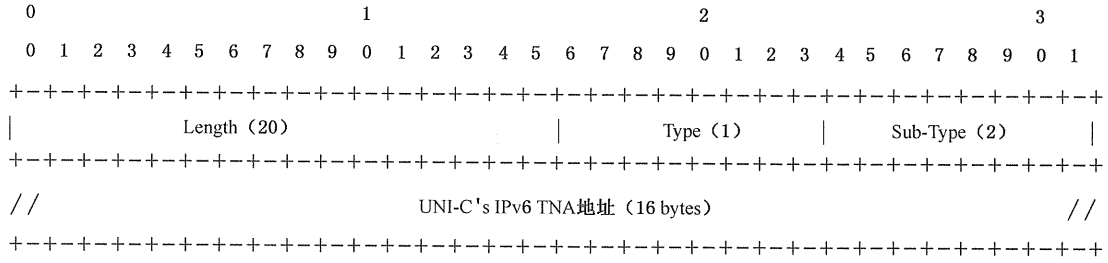


图 31　源 TNA　IPv6 地址子对象

源 NSAP TNA 地址子对象格式如图 32 所示：

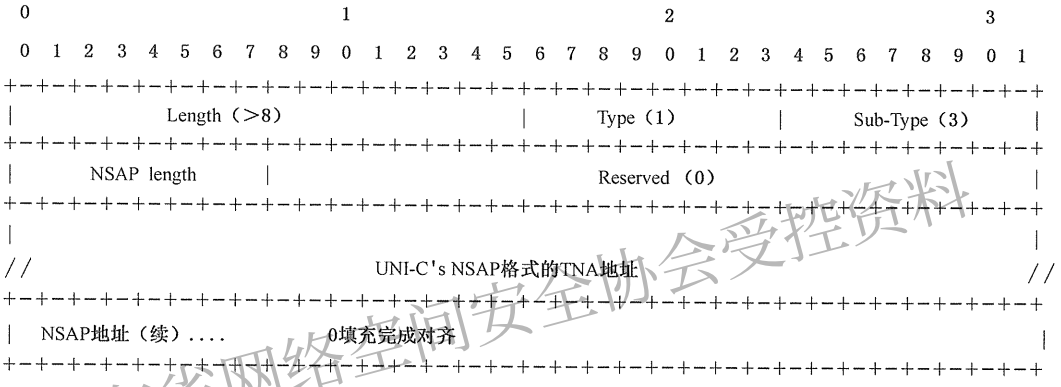


图 32　源 NSAP TNA 地址子对象

NSAP 长度(8 位)：源 NSAP 字节总长度。

源 NSAP 格式的 TNA 地址：变长域，结构根据 ISO/IEC8348，1993 并和 ITU X.213，1992 相同。

- 目的 TNA 地址子对象(Type=2)

目的 TNA 地址子对象包含目的 UNI-C 的 TNA 地址值。可以有以下三种格式：IPv4、IPv6 或 NSAP。如果 GENERALIZED_UNI 对象包含多个目的 TNA 地址子对象，只有第一个有意义其他的应忽略。

目的 IPv4 TNA 地址子对象格式同图 30，其中：Type =2，Sub-Type=1；

目的 IPv6 TNA 地址子对象格式同图 31 所示，其中：Type =2，Sub-Type=2；

目的 NSAP 格式的 TNA 地址子对象格式同图 32 所示，其中：Type =2，Sub-Tpye=3。

- 分集子对象(Type= 3，Sub-Type = 1)

分集子对象在 UNI 中定义，用来指定为一条新的连接所需要的物理路由分集。它携带本地一条存在连接的标识，可以在 Path 消息中携带。可以使用分集子对象的多个实例。子对象格式如图 33 所示：

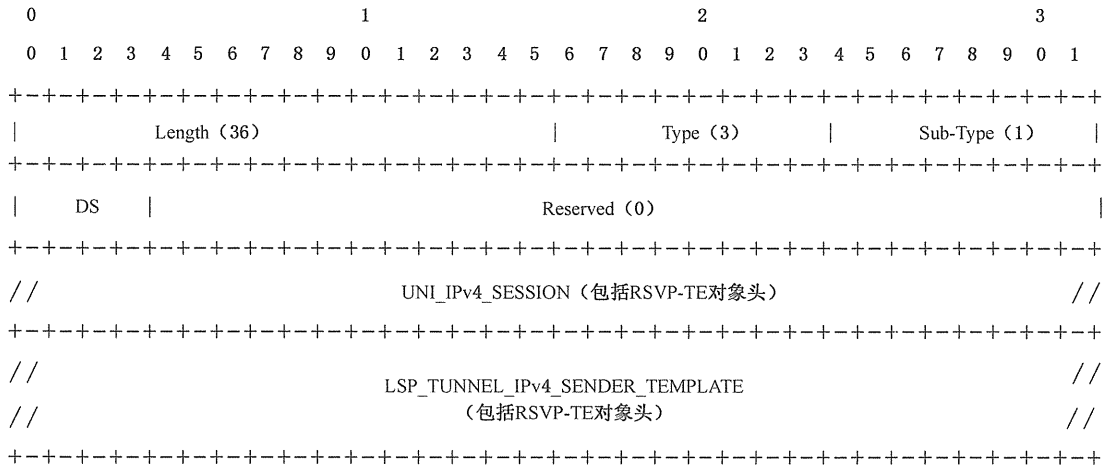


图 33　分集子对象格式

DS 指示所需要的分集需求，取值定义如下：

1：节点分离。新建连接不应该使用所列举的连接所经过路径上的任何节点；

2：链路分离。新建连接不应该使用所列举的连接所经过路径上的任何链路；

3：SRLG 分离。新建连接不应该使用和所列举连接有相同 SRLG 的任何链路；

4：共享路径。新建连接应该使用和所列举连接相同的链路；

0、5～15：保留。

分集子对象由源 UNI-N 进行处理。对于目的 UNI-N 或 UNI-C 该对象没有意义。因此，该对象不会发送到目的 UNI-C。

如果不能提供所请求的路由分集要求或接收到的分集对象中指定的值无法支持，源 UNI-N 应上报原因为“路由问题：分集要求不可用”的错误信息。

- 出口标签子对象（Type=4，Sub-Type=1）

UNI 定义了出口标签子对象用来指定目的 UNI 使用的出端口、标签。一共包括两个出口标签子对象。格式如图 34 所示：

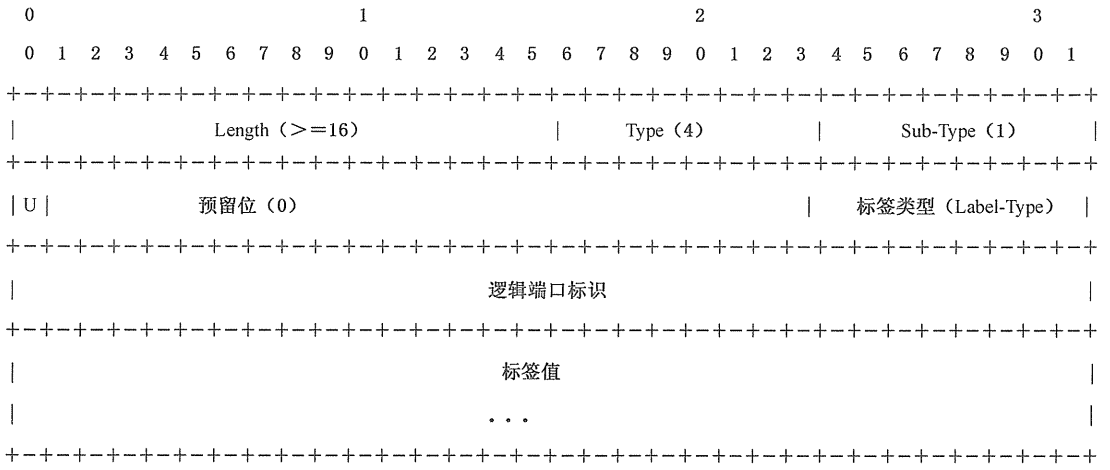


图 34　出口标签子对象格式

逻辑端口标识：

该域指示在目的 UNI-C　分配的逻辑端口标识。用来在目的 UNI 选择一条链路。链路标识

应用来为连接分配资源；

U 比特位：

该比特指示标签和逻辑端口 ID 的方向。0 表示下游方向标签和端口 ID，1 标识上游方向标签和端口 ID。仅仅用在双向连接的情形中。

标签类型：

该域用来指示标签子对象的 C-Type 类型。1 表示为出口标签、2 表示为软永久连接（SPC）标签。

标签值：

域用来使用的标签。该域的格式和 GMPLS 标签的标签域的使用相同。

下列原因会导致“错误的显式路由对象”的错误。

——如果逻辑端口标识不能在目的端有效标识一条链路；

——如果请求单向连接建立时将 U 比特置位；

——如果在有两个标签子对象有相同的 U 比特值。

目的 UNI-N 为单向连接检查一个子对象，对双向连接检查两个子对象。如果子对象的 U 比特检查出为 0，则该标签值拷贝到新的标签集对象中。该标签集对象应包含在目的 UNI-N 的相关输出 Path 消息中。如果目的 UNI-C 不能从标签集中提取标签或在解析标签集时出现错误，则终结连接请求并产生 PathErr 消息，其中指示“路由问题/标签集”。

如果 U 比特被检测出置为 1，则标签值用来为双向 LSP 上游方向分配的标签值。如果目的 UNI-N 无法接受该标签值，则应产生 PathErr 消息并指示“路由问题/标签分配失败”。如果目的 UNI-N 能够接受标签值，则该标签值拷贝到上游方向标签对象中去。该上游方向标签应包含在相关的输出 Path 消息中。如果目的 UNI-C 不能接受该标签值，则应产生 PathErr 消息并指示“路由问题/标签值不可接受”。

产生的 PathErr 消息可以包含可接受的标签集，见 IETF RFC3473。否则，Path 消息按通常的处理办法来处理。

处理完成后，标签子对象从 GENERALIZED_UNI 对象中删除。逻辑端口标识映射到（以本地标识为基础）IF_ID_RSVP-TE_HOP 对象的“接口 ID”域的 3、4、5 号类型的 TLV 中（见 IETF RFC3471 和 IETF RFC3473）。

- 服务等级子对象（Type=5，Sub-Type=1）

服务等级子对象指定 0～255 中的一个整数值，并可能包含在 GENERALIZED_UNI 对象中。每个服务等级和运营商预先定义的特征相关，如恢复类型（如，无保护、1+1 保护）、故障修复后连接返回策略、保持策略等。子对象格式如图 35 所示：

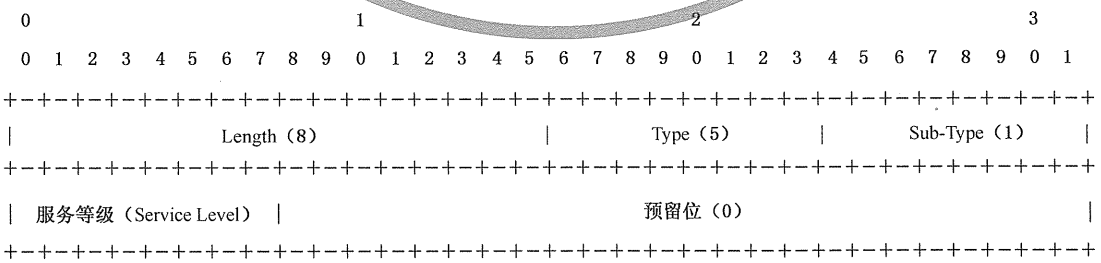


图 35 SLA 子对象格式

如果接收到的服务等级 TLV 指定的值不支持，节点应上报“路由问题：服务等级不可用”的错误信息。

d) GENERALIZED_LABEL_REQUEST 对象（Class-Num = 19 [IETF RFC 3473]）

GENERALIZED_LABEL_REQUEST 对象格式见 GMPLS RSVP-TE 中的定义。UNI 节点须支持 SDH LSP 编码类型和 TDM 交换类型。

e) IPv4_RESV_CONFIRM 对象(Class-Num = 15,[IETF RFC2205])

IPv4_RESV_CONFIRM 对象格式见 IETF RFC2205 中的定义。在目的 UNI,IPv4 接收者地址设置为目的 UNI-C 的节点 ID。在源 UNI,地址设置为源 UNI-N 的节点 ID。

f) IPv4_ERROR_SPEC 对象(Class-Num = 6 [IETF RFC2205])

IPv4_ERROR_SPEC 对象格式见 IETF RFC2205 中的定义。IPv4 出错节点地址设置为报告出错的 UNI-C 或 UNI-N 的节点 ID。UNI-N 和 UNI-C 实体应支持 InPlace、NotGuilty 和 Path_State_Removed 标志位。

注：IF_ID ERROR_SPEC 对象(Class-Num = 6 [IETF RFC3471])

指定和错误相关的具体的接口是非常有用的。为支持这种情形，定义了 IF_ID ERROR_SPEC 对象。IPv4/IPv6 IF_ID ERROR_SPEC 对象见 IETF RFC3473 中的定义。

希望指示与出错相关的具体接口的节点应该在相关的 PathErr 或 ResvErr 消息中使用相应的 IF_ID ERROR_SPEC 对象。IF_ID ERROR_SPEC 对象应该象其他 ERROR_SPEC 对象一样来产生和处理。

g) LSP_TUNNEL_IPv4_FILTER_SPEC 对象(Class-Num = 10 [IETF RFC3209])

LSP_TUNNEL_IPv4_FILTER_SPEC 对象和 UNI_IPv4_SESSION 对象组合来唯一标识一条连接。对象格式和 LSP_TUNNEL_IPv4_SENDER_TEMPLATE 对象相同。

h) MESSA MESSAGE_ID 对象(Class-Num = 23 [IETF RFC2961])

MESSA MESSAGE_ID 对象格式见 IETF RFC2961 中的定义。对于 UNI 中的触发消息、PathErr、ResvErr、ResvConf 消息以及 Srefresh 消息的 MESSAGE_ID 对象的 Ack_Desired 标志位应置 1 用来支持消息的可靠收发。刷新消息没有收到应答时一定不能导致节点删除对应的连接。

i) IPv4_IF_ID_RSVP_HOP 对象(Class-Num = 3,[IETF RFC3473])

IPv4_IF_ID_RSVP_HOP 对象格式见 GMPLS RSVP-TE 中的定义。对于 UNI，IPv4_IF_ID_RSVP_HOP 对象应用来选择位连接分配资源的数据链路。数据链路从 Path 消息的发送者的观点来指定。接收包含一个或多个 TLV 的 Path 消息的节点将 TLV 保存，并将这些值在随后的 Resv 消息中包含该值向产生该 TLV 的节点发送。这样看来，TLV 用来标识和一条 LSP 相关的数据通道：

- 对于单向 LSP，应指示下游方的数据链路；
- 对于双向 LSP，正常情形下要指示公用的上游和下游方向数据链路。在通过绑定链路的双向 LSP 的特殊情形下，可能要指定一条下游方向的数据链路，这条数据链路和上游方向的数据链路不同。

节点的节点标识用来填充 IPv4 IF_ID_RSVP_HOP 对象的 IP 地址域。

注：当不需要 TLV 时，不应该使用 IF_ID RSVP_HOP 对象。

j) 呼叫 ID(Call_ID)对象(Class-Num = 230,[IETF RFC3474])

呼叫 ID 对象定义于 IETF RFC3474，当进行呼叫/连接的初次创建时，呼叫 ID 中的 C-Type 及呼叫的值为全 0，但 IETF RFC3474 中没有明确空呼叫的定义。下面为空呼叫的定义：长度为 4，C-Type 为 0，没有其余字段。当呼叫 ID 得到后，呼叫 ID 的格式按照 IETF RFC3474 中的格式定义，如图 36 所示。

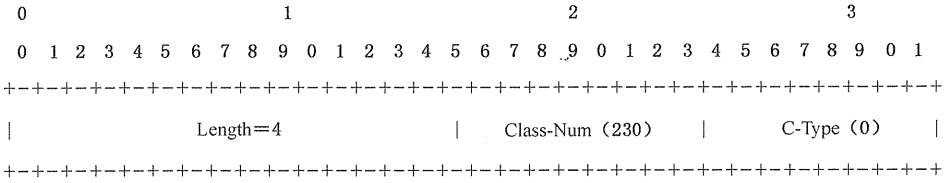


图 36 呼叫 ID 对象头格式

8.2.4.4 UNI RSVP-TE 错误编码定义

错误编码值用来描述连接活动的出错的原因。在连接建立过程中产生的错误被作为 RSVP-TE 协议定义的一部分。

8.2.5 UNI 支持 SDH 业务的 RSVP-TE 扩展

8.2.5.1 SDH 业务流量参数

SDH 业务流量参数由 RSVP-TE 协议的 SENDER_TSPEC 和 FLOWSPEC 对象来携带。具体 SENDER_TSPEC 和 FLOWSPEC 对象的说明见 IETF RFC2205 中的定义。其中：

SDH SENDER_TSPEC 对象：Class = 12，C-Type = 4；

SDH FLOWSPEC 对象：Class = 9，C-Type = 4

SDH 业务流量参数的格式如图 37 所示：

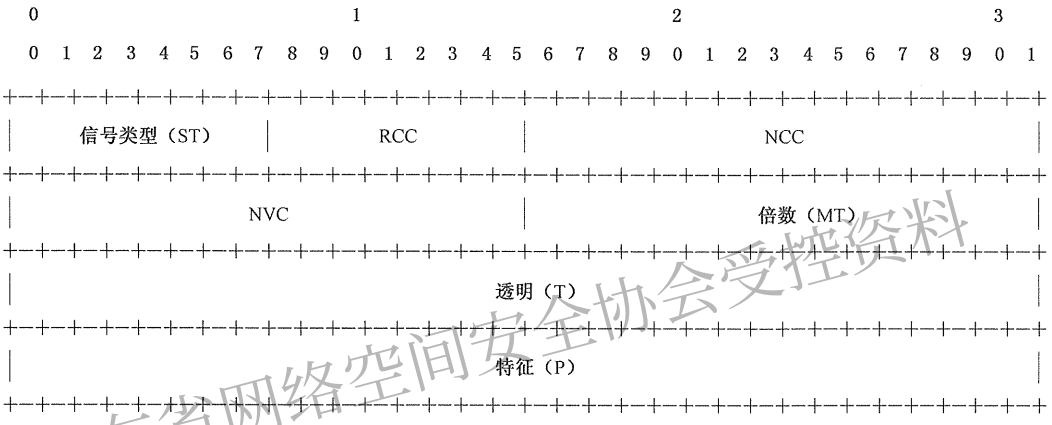


图 37 SDH 业务流量参数格式

其中：

- 信号类型(ST)/8 比特：指示所请求 LSP 的基本信号类型。类型值如图 38 所示：

Value	Type（基本信号）
2	VC-12
5	VC-3
6	VC-4

图 38 UNI 接口所支持的 SDH 业务连接请求的基本信号类型

- RCC/8 比特：所请求的基本信号的连续级联类型。该指示域为位域。每一个比特位指示一种特定的连续级联类型。多个标置位可以同时设置来表示可以有多种选择。当上游节点设置多个标置位时表示上游节点支持多种级联类型，下游节点可以根据本地所支持的级联类型来选择一种级联类型。如果下游节点多，所有的级联类型都不支持，则拒绝 LSP 建立。最低位的比特置 1 表示标准连续级联，其他为预留比特。
- NCC/16 比特：所请求的基本信号的 RCC 域中所指示的连续级联数。
- NVC/16 比特：所请求的基本信号的虚级联数。
- 倍数(MT)/16 比特：所请求的相同的信号（包括上述所指示的连续级联信号、虚级联信号或非

级联信号)的倍数,这些信号构成最终的一条 LSP。该值应大于或等于 1,表示只请求一个或多个信号实例。0 为非法值。

- 透明指示(T)/32 比特:该指示域为位域,每一个比特位指示一种透明类型。比特位置 1,表示该类型透明传输请求,其中:
  Flag 1(bit 1,最低位):再生段透明指示;
  Flag 2(bit 2,倒数第 2 位):复用段透明指示。
- 特征指示(P)/32 比特:支持所请求的 LSP 具有的特定的能力,如对 LSP 具有故障监视能力。不同的能力目前还没有标准化,未来可能需要通过扩展 TLV 来说明标准的能力。

附录 B 中描述了 SDH 业务的流量参数举例。

8.2.5.2 SDH 标签编码

UNI 接口中,SDH 标签使用 IETF IETF RFC3473 中定义的通用标签对象来承载。即通用标签对象格式如图 39 所示:

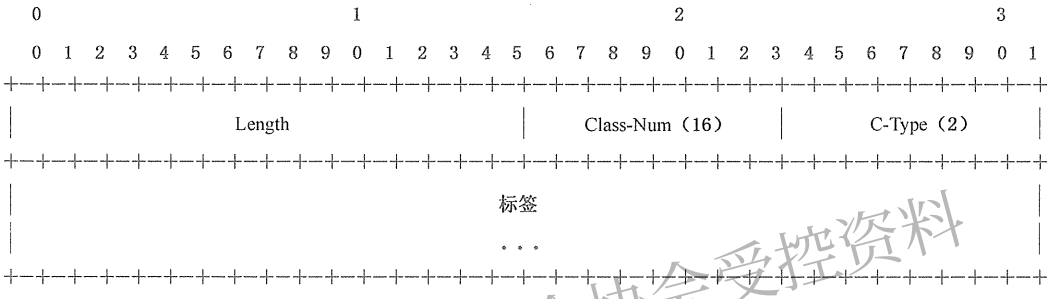


图 39 通用标签对象格式

SDH 标签值采用如图 40 所示的格式进行编码:

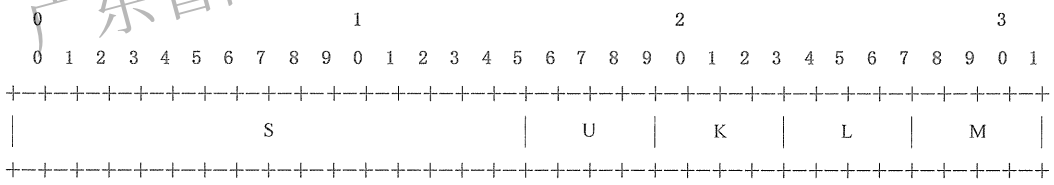


图 40 SDH 标签编码格式

该编码规则顺从 ITU-T G.707 中对 SDH 的时隙编号机制。其中:

S:表示 STM-N 信号中 AUG-1 的具体编号。取值范围在 1～N 之间;

U:表示 AUG-1 信号中 VC-3 的具体编号。取值范围在 1～3 之间。通常情况下我国 SDH 体制不采用此复用信号结构,因此该值应置 0;

K:表示 VC4 信号中 TUG-3 的具体编号。取值范围在 1～3 之间;

L:表示 TUG-3 或 VC3 信号中 TUG-2 的具体编号。取值范围在 1～7 之间;

M:表示 TUG-2 信号中对应 VC12 的时隙位置。取值范围在 3～5 之间。M 的取值大小和 VC12 在 TUG-2 中的时隙大小对应,即,M=3 对应 1 号 VC12 时隙,依此类推。

编码值和时隙对应关系如图 41 所示:

S	SDH
0	other
1	1st AUG-1
2	2nd AUG-1
3	3rd AUG-1
4	4rd AUG-1
:	:
N	Nth AUG-1

U	SDH AUG-1
0	other
1	1st VC-3
2	2nd VC-3
3	3rd VC-3

K	SDH VC-4
0	other
1	1st TUG-3
2	2nd TUG-3
3	3rd TUG-3

L	SDH TUG-3	SDH VC-3
0	other	other
1	1st TUG-2	1st TUG-2
2	2nd TUG-2	2nd TUG-2
3	3rd TUG-2	3rd TUG-2
4	4th TUG-2	4th TUG-2
5	5th TUG-2	5th TUG-2
6	6th TUG-2	6th TUG-2
7	7th TUG-2	7th TUG-2

M	SDH TUG-2
0	other
3	1st VC-12
4	2nd VC-12
5	3rd VC-12

图 41 SUKLM 标签编码和 SDH 时隙对应关系

在所请求的 LSP 连接业务为连续级联情况下，分配的标签值只指定连续级联信号的第一个起始时隙值；

在所请求的 LSP 连接业务为虚级联的情况下，在通用标签对象中应严格按顺序列举所分配的所有标签值。如果虚级联成员为同一个 LSP 上的成员，则所有这些虚级联成员应走相同的路由。

8.2.6 UNI 支持以太网业务的 RSVP-TE 扩展

8.2.6.1 以太网业务流量参数

以太网业务流量参数由 RSVP-TE 协议的 SENDER_TSPEC 和 FLOWSPEC 对象来携带。具体 SENDER_TSPEC 和 FLOWSPEC 对象的说明见 IETF RFC2205 中的定义。其中：

以太网 SENDER_TSPEC 对象：Class = 12，C-Type = 6(暂定)；

以太网 FLOWSPEC 对象：Class = 9，C-Type = 6(暂定)；

以太网业务流量参数格式如图 42 所示：

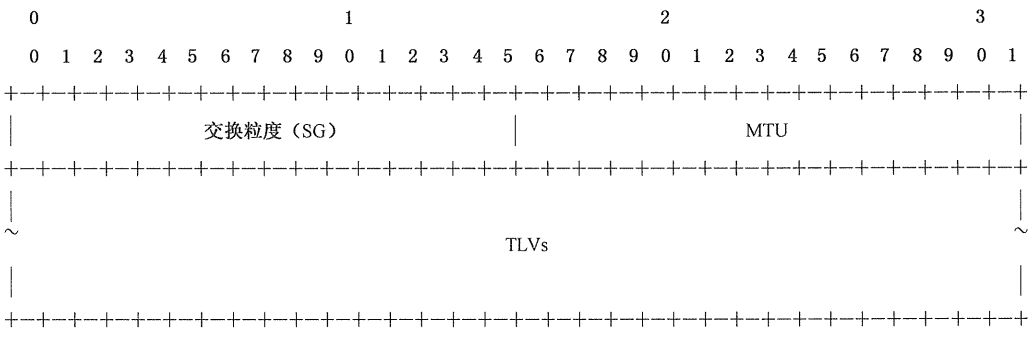


图 42 以太网流量参数格式

其中：

- 交换粒度(SG)/16 比特：指示承载所请求的以太网络 LSP 的以太网链路类型。1 表示以太端口交换(如，EPL)；2 表示以太帧交换(如，EVPL)；其他值为预留值。
- MTU/16 比特：最大传送单元。指示两个八位组的总数。
- TLV：标准的类型长度值表示，流量参数对象中至少包含一个 TLV。其中，类型值＝129、长度值＝24 时，表示该 TLV 指示以太网带宽值。该 TLV 的格式如图 43 所示：

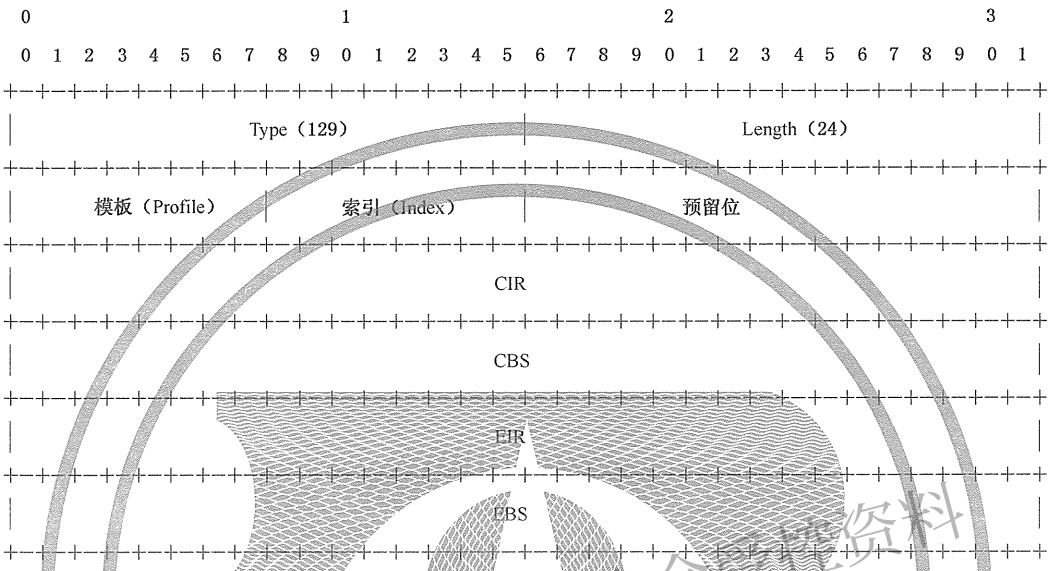


图 43 以太网带宽参数 TLV

- 模板(Profile)：该域为位域，其中低两位比特定义如下，其他比特位预留：
  标志位 1(bit 0)：耦合标置位(CF)；
  标志位 2(bit 1)：颜色模式标置位(CM)；
  其他位为预留位，发送侧置 0，接收侧忽略；
  标志位 1 可以用来在两个速率实现算法的操作模式上进行选择，置 1 表示要请求对应的测量；
  标志位 2 可以用来确定模板是颜色识别(color-aware)还是不区分颜色(color-blind)，置 1 表示颜色识别模式，置 0 表示颜色不区分模式。
- 索引(Index)：在请求多种类型的 LSP 时，该域用来对指定的业务确定带宽分配方式。缺省情况下应置 0。
- 承诺的信息速率(CIR)/32 位：单位为字节/秒，并按 32 位 IEEE 单精度浮点数编码。
- 承诺的突发尺寸(CBS)/32 位：单位为字节，并按 32 位 IEEE 单精度浮点数编码。当 CIR 严格大于 0 时，该值应大于或等于最大报文帧的尺寸。
- 超额信息速率(EIR)/32 位：单位为字节/秒，并按 32 位 IEEE 单精度浮点数编码。
- 超额的突发尺寸(EBS)/32 位：单位为字节，并按 32 位 IEEE 单精度浮点数编码。当 EIR 严格大于 0 时，该值应大于或等于最大报文帧的尺寸。

8.2.7 UNI 支持 OTN 业务的 RSVP-TE 扩展

8.2.7.1 OTN 业务流量参数

OTN 业务流量参数由 RSVP-TE 协议的 SENDER_TSPEC 和 FLOWSPEC 对象来携带。具体 SENDER_TSPEC 和 FLOWSPEC 对象的说明见 IETF RFC2205 中的定义。其中：

OTN SENDER_TSPEC 对象：Class = 12，C-Type = 5；

OTN FLOWSPEC 对象：Class = 9，C-Type = 5。

OTN 业务流量参数的格式如图 44 所示：

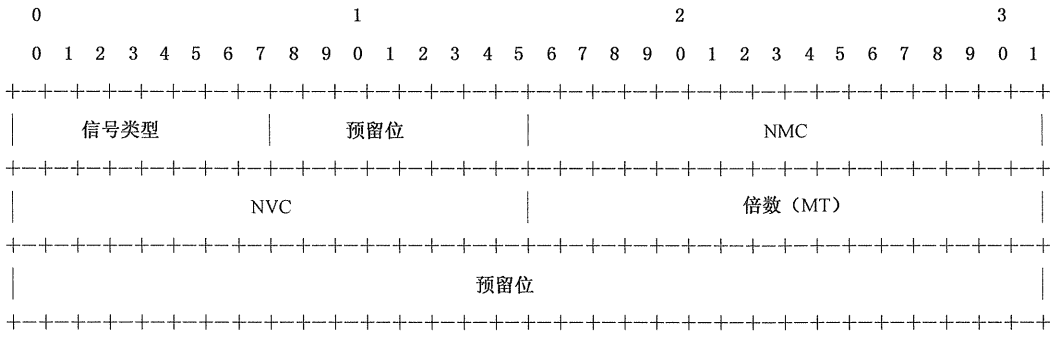


图 44 OTN 业务流量参数格式

其中：

- 信号类型(ST)/8 比特：指示所请求 LSP 的基本信号类型。类型值如图 45 所示：

Value	Type
1	ODU_1（i.e.，2.5 Gbps）
2	ODU_2（i.e.，10 Gbps）
3	ODU_3（i.e.，40 Gbps）
4	预留
5	预留
6	OCh at 2.5 Gbps
7	OCh at 10 Gbps
8	OCh at 40 Gbps
9-255	预留

图 45 UNI 接口所支持的 OTN 业务连接请求的基本信号类型

- 复用组件数(NMC)/16 比特：所请求的 LSP 中，用来指示 ODU_j 复用到 ODU_k 中的支路时隙个数。
- NVC/16 比特：所请求的基本信号为 ODU_k 的 LSP 中的 ODU_k 的虚级联数（只对 ODU_k 有效，其他类型的基本信号不支持虚级联，因此应填 0）。
- 倍数(MT)/16 比特：所请求的相同的信号的倍数，这些信号构成最终的一条 LSP。该值应大于或等于 1，表示只请求一个或多个信号实例，0 为非法值。

8.2.7.2 OTN ODU_k 标签编码

UNI 接口中，OTN ODU_k 标签使用 IETF RFC3473 中定义的通用标签对象来承载。通用标签对象格式如图 39 所示。

OTN 标签值采用图 46 所示的格式进行编码：

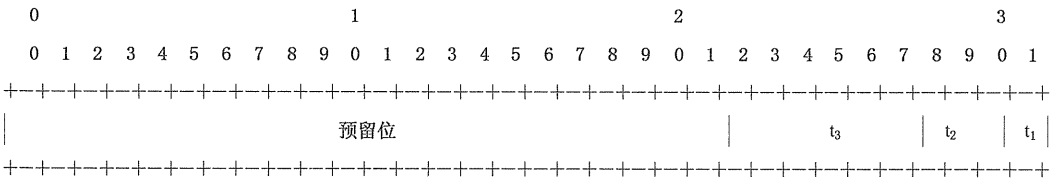


图 46 OTN ODU_k 标签编码格式

各个变量意义如下：

t_1(1 比特)：t_1=1 表示 ODU_1 信号，其他信号类型置 0；

t_2(3 比特)：t_2=1 表示不能进一步细分的 ODU_2 信号。

t_2=[2...5]和 ODU_1 复用到 ODU_2 中的时隙位置一一对应（ODU_1 的时隙编号=t_2−1）；

t_3(6 比特)：t_3=1 表示不能进一步细分的 ODU_3 信号。

t_3=[2...17]和 ODU_1 复用到 ODU_3 中的时隙位置一一对应（ODU_1 的时隙编号=t_3−1）；

t_3=[18...33]和 ODU_2 复用到 ODU_3 中的时隙位置一一对应（ODU_2 的时隙编号=t_3−17）。

OTN OCh 业务标签编码格式参考 IETF RFC3471 中对波长标签的编码。

8.3 IETF GMPLS UNI 信令功能

8.3.1 IETF GMPLS UNI 信令功能概述

OIF 的协议一个重要功能为提供异构网络的互联，因此 UNI 信令协议和运营商网络内部 NNI 信令协议可以不相同。IETF 对 GMPLS 最初的设计理念是同时考虑 UNI，NNI 并提供一致的模型，因此 IETF 中 UNI 信令协议和 NNI 信令协议是一致的，如采用 RSVP-TE。这也导致了 IETF GMPLS UNI 和 OIF UNI 的一些差异。

8.3.2 UNI 呼叫处理过程

8.3.2.1 UNI 呼叫支持的基本功能

IETF RFC4974 中描述了 IETF 中通过扩展 RSVP-TE 对呼叫的支持。根据 IETF 的定义，呼叫是端点间的一种协商，也可能需要网络中间节点间的配合。呼叫的建立包括能力交换、策略、鉴权及安全。呼叫可用来方便地管理一组连接，用于提供端到端的数据传送业务。

呼叫用来简化为支持端到端数据业务的一组连接的管理。当连接的状态在网络内部整个数据平面连接经过的节点上都需要维护的时候，呼叫消息不需要参与中间节点对连接的管理，除了需要处理呼叫的边界节点之外只需要在业务的两个端点进行管理。

呼叫的建立和维护可以独立于他所关联的连接来进行处理。

呼叫用来提供如下几个基本的功能：

a) 在连接建立之前，验证和标识呼叫的发起者；

b) 支持虚级联时走不同路由的多条组件 LSP；

c) 在单个呼叫下关联多条 LSP 连接；

d) 通过允许关联的 LSP 操作状态的改变来简化控制平面的维护。

8.3.2.2 接入链路能力的交换

在层叠模型的网络中，通过能力交换和协商，能使得源 UNI-C/UNI-N 知晓宿 UNI-C/UNI-N 的对应能力及其 UNI-C，UNI-N 间的接入链路能力，这样能使得后续 LSP 建立能更加高效，提高网络的资源利用率及连接建立的成功率。

能力交换和协商的过程如下：

a) 呼叫源端在呼叫消息中携带本端能力（例如适配能力）、接入链路能力（例如可用的波长），发往呼叫宿端。

b) 呼叫宿收到呼叫请求后，进行协商，配置宿端属性，在呼叫应答中返回协商后的宿端能力、宿端接入链路的能力。

c) 呼叫源端收到呼叫应答后，进行源端属性（例如适配能力）配置，并根据返回的宿端接入链路信息完成网络连接的创建。

以 SDH 网络或 OTN 网络提供以太网业务为例，在源 UNI-N，要将 UNI-C 发送过来的以太网帧经过封装、适配将到 TDM 容器中去，在宿 UNI-N，经过反向的过程将 TDM 容器中的数据去适配、解封装为以太网帧，发往宿 UNI-C。源、宿 UNI-N 的封装协议可有多种，如 GFP，LAPS；适配方式也有多种，如是否支持虚级联，相邻级联，是否支持 LCAS。要成功建立以太网业务，应源 UNI-N 和宿 UNI-N 采用相同的封装和适配方式，否则业务不能成功建立。因此在以太网业务连接建立前，首先通过上述的能力交换和协商过程完成两个端节点中以太网信号从用户侧到网络侧的协议适配栈和封装信息，然后再协商确定两个端节点之间用于支撑该以太网业务的网络连接类型和数量信息。通过建立所确定的网络连接类型和数量并利用这些信息对两个端点进行相应的配置后才能完成以太网业务的成功建立。

能力的交换和协商还包括 UNI-C 和 UNI-N 间的链路能力交换和协商。例如当通过波分网络承载 OCh 级别的 OTN 业务时，由于波分网络特有的限制（例如波长转换受限），要求波分路径中尽量使用相同的波长，因此源端知晓宿端接入链路的能力（如可用波长）能提高波长路径的成功建立率。

8.3.2.3 呼叫建立

GMPLS 中通过通告（Notify）消息进行呼叫的建立，呼叫和连接互相独立，呼叫的信令消息不必沿着连接的路由传送，而在 OIF 中，呼叫的建立由 Path 消息完成，因此呼叫和连接的建立同时进行。在 IETF 中，呼叫的建立在连接建立前完成。一个呼叫下面可没有关联连接，也可关联一条或多条连接。

呼叫的建立过程如图 47 所示：

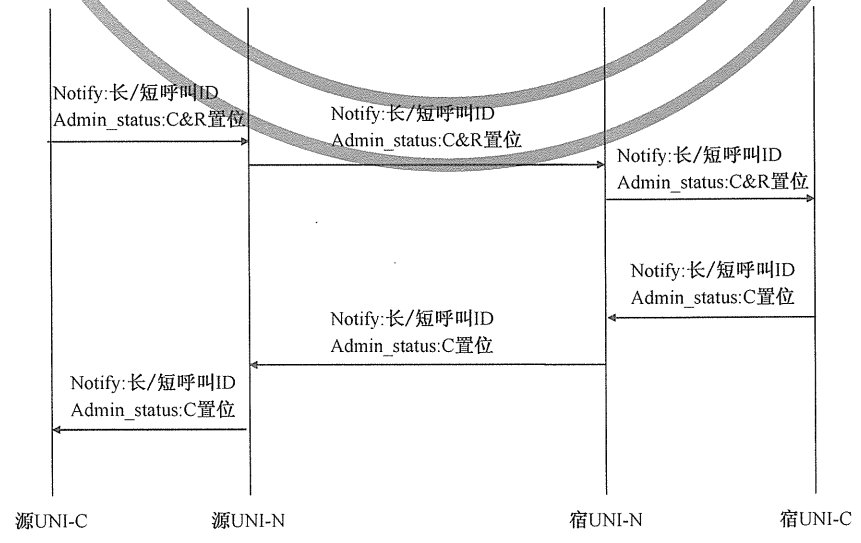


图 47 IETF GMPLS UNI 呼叫建立过程

源 UNI-C 发起呼叫请求，Notify 消息中 Admin_status 对象的 C，R 比特置位，源 UNI-C 可分配长呼叫 ID 和短呼叫 ID(长呼叫 ID 和短呼叫 ID 也由源 UNI-N 分配)。源 UNI-N，宿 UNI-N，宿 UNI-C 依次对呼叫进行校验，如果校验不成功，将返回呼叫建立失败，其中错误码携带在 Notify 消息中的 ERROR_SPEC 对象中(具体的错误码可参见 IETF RFC2205)。当宿 UNI-C 完成对呼叫请求的校验时，将通过 Notify 返回呼叫建立确认消息，该消息 Admin_status 对象的中 C 比特置位，其余比特不置位，当源 UNI-C 收到确认消息后，呼叫建立成功。在呼叫建立中，通过 Message ID 保证保障消息的可靠传送。如果源 UNI-C 在发送连续多次呼叫建立请求消息后(该次数可配置)，未收到 Message ID 回应，源 UNI-C 将发起呼叫删除。

8.3.2.4 呼叫中增加连接

当呼叫建立后，可向呼叫中增加连接。由于短呼叫 ID(Short Call ID)属于 SESSION 对象的一部分，因此在 SESSION 对象中具有相同短呼叫 ID(非 0)的连接属于同一个呼叫。没有和呼叫关联的连接中的短呼叫 ID 的值为 0，这样可以区分连接是否已和呼叫关联。当向呼叫中增加连接时，在建立连接的 Path 消息的 SESSION 对象中，将短呼叫 ID 设置为被关联呼叫的 ID，连接建立完成后即完成了向呼叫中增加连接。

8.3.2.5 删除呼叫中的连接

删除呼叫中的一条连接和 IETF RFC3473 中定义的标准删除过程一致。不支持将一条连接从呼叫中移出而不删除该连接。当呼叫中的最后一条连接被删除时，由于呼叫独立于连接，呼叫状态可以保留，需要显示的删除呼叫。

8.3.2.6 呼叫删除

只有当一个呼叫下面没有连接关联时，呼叫才能被删除。当试图删除存在连接关联的呼叫，该删除应被拒绝，同时返回呼叫管理(Call Management)错误码，错误值为连接依然存在(Connections Still Exist)，呼叫的状态保持不变。

呼叫删除过程如图 48 所示：

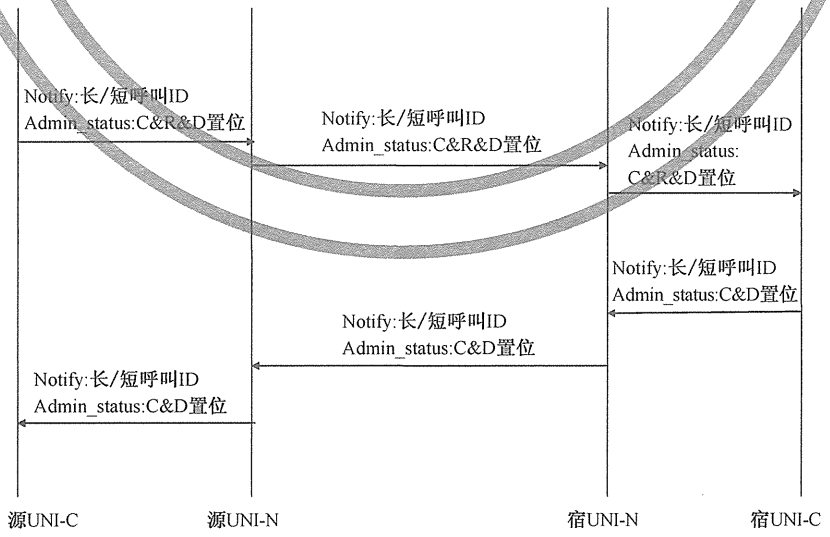


图 48 IETF GMPLS UNI 呼叫删除过程

呼叫删除消息由 Notify 消息携带，呼叫删除请求 Notify 消息中 Admin_status 对象 R，C，D 比特置位，其余比特不置位，呼叫删除响应消息中 R，D 比特置位，其余比特不置位。如果源 UNI-C 未收到呼

叫删除响应，如同呼叫建立中一样，源 UNI-C 可将呼叫删除消息重发一定的次数，如果还未收到删除响应，源 UNI-C 将认为呼叫已被删除，但这种情况下，该呼叫的长呼叫 ID 或短呼叫 ID 应该至少在 Notify 消息的 5 个刷新周期内不被重新使用。

由于呼叫是对称的，呼叫的删除也可由宿 UNI-C 发起，宿端发起呼叫删除的流程和源端发起类似。存在源 UNI-C 和宿 UNI-C 同时发起呼叫删除的可能，这种情况下，源和宿都会收到 Admin_status 对象中 R 比特被置位的 Notify 消息，此时该 Notify 消息都要被响应。

8.3.2.7 对象扩展

为了支持 IETF 中的呼叫功能，RSVP-TE 中的消息进行了对应的扩展：

扩展后的 Notify 消息格式如下：

```
〈Notify message〉  ::= 〈Common Header〉 [ 〈INTEGRITY〉 ]
                       [[ 〈MESSAGE_ID_ACK〉 | 〈MESSAGE_ID_NACK〉]...]
                       [ 〈MESSAGE_ID〉 ]
                       〈ERROR_SPEC〉
                       〈notify session list〉

〈notify session list〉::= [ 〈notify session list〉 ] 〈notify session〉

〈notify session〉  ::= 〈SESSION〉 [ 〈ADMIN_STATUS〉 ]
                       [ 〈POLICY_DATA〉...]
                       [ 〈LINK_CAPABILITY〉]
                       [ 〈SESSION_ATTRIBUTE〉 ]
                       [ 〈sender descriptor〉 | 〈flow descriptor〉 ]
```

notify session list 对象中可携带多个 notify session，在一个 notify 消息中携带多个并行的 Call 消息。

新增的 LINK_CAPABILITY 对象用于携带接入链路的能力信息。

LINK_CAPABILITY 对象包含一个或多个可变的子对象(SubObject)，如图 49 所示：

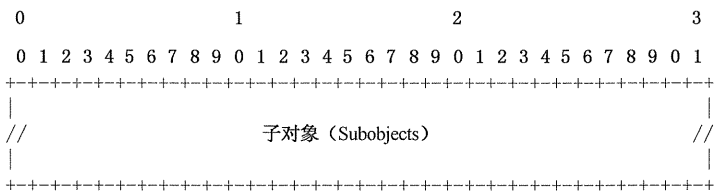


图 49 链路能力子对象格式

目前定义的子对象如下：

类型 1：链路的本端采用有编号的 Ipv4 地址

类型 2：链路的本端采用有编号的 Ipv6 地址

类型 64：该链路上的最大可预留带宽

类型 65：该链路接口交换能力

ADMIN_STATUS 对象增加的 C 比特位用于呼叫控制，更改后的对象格式如图 50 所示：

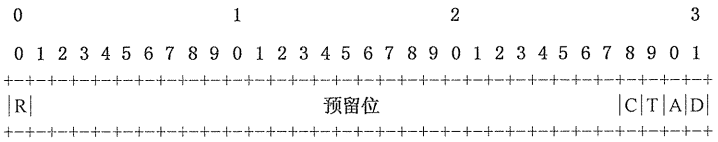


图 50 管理状态对象格式

C 比特用于指示该 Notify 消息是否为呼叫控制管理消息，当置位时标识该 Notify 消息用于呼叫控制和管理。

长呼叫 ID(Long Call ID)：

长呼叫 ID 用于标识一个全局唯一的呼叫，该 ID 只在对呼叫进行建立的 Notify 消息中需要。在 SESSION_ATTRIBUTE 对象的 SESSION_NAME 字段中放置长呼叫 ID。

短呼叫 ID(Short Call ID)：

为了将连接与呼叫进行关联，需要在连接中携带对呼叫的引用：短呼叫 ID。短呼叫 ID 携带在 SESSION 对象的保留字段中，长度为 16 bits，该 ID 在 SESSION 对象的 Tunnel_End_Point_Address 字段及 SENDER_TEMPLATE 对象的 Sender_Address 字段确定的空间中唯一。携带了短呼叫 ID 的 SESSION 对象格式如图 51 所示：

图 51 SESSION 对象格式

短呼叫 ID 的长度为 16 bits，在呼叫的生命周期中不变。

8.3.3 UNI 连接处理过程

8.3.3.1 连接建立

IETF 中 UNI 和 NNI 都基于 GMPLS 的 RSVP-TE，因此 IETF GMPLS UNI 中的连接建立可以允许一个 RSVP SESSION 触发 UNI 的信令及 NNI 信令，完成 UNI 连接的建立和删除。UNI 连接的建立方式可采用连续(Continuous)LSP 或嵌套(Nesting)/拼接(Stitching)LSP 的方式。对于连续 LSP 的方式创建的 UNI 连接，只存在源 UNI-C 到宿 UNI-C 间的一个会话，但，连续的信令方式要求 INNI 也支持同样的信令协议；对于嵌套/拼接方式创建的 UNI 连接，存在源宿 UNI-C 间的一个会话及源宿 UNI-N 间的一个会话。

连续 LSP 建立 UNI 连接的过程如图 52 所示：

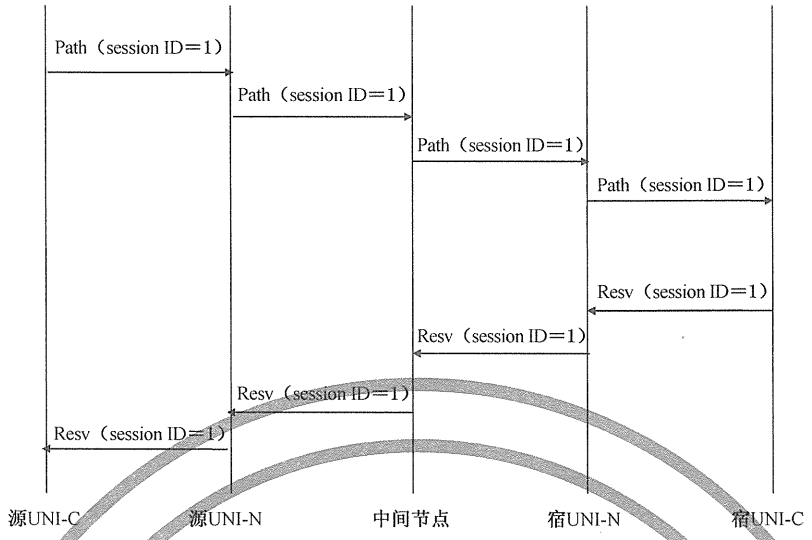


图 52　IETF GMPLS UNI 连续 LSP 建立过程

如上图通过源 UNI-C 到宿 UNI-C 的连续 RSVP 信令完成 UNI 连接的建立。在 UNI-N 处，由于地址隐藏，可能会进行地址映射。

嵌套/拼接 LSP 建立 UNI 连接的过程如图 53 所示：

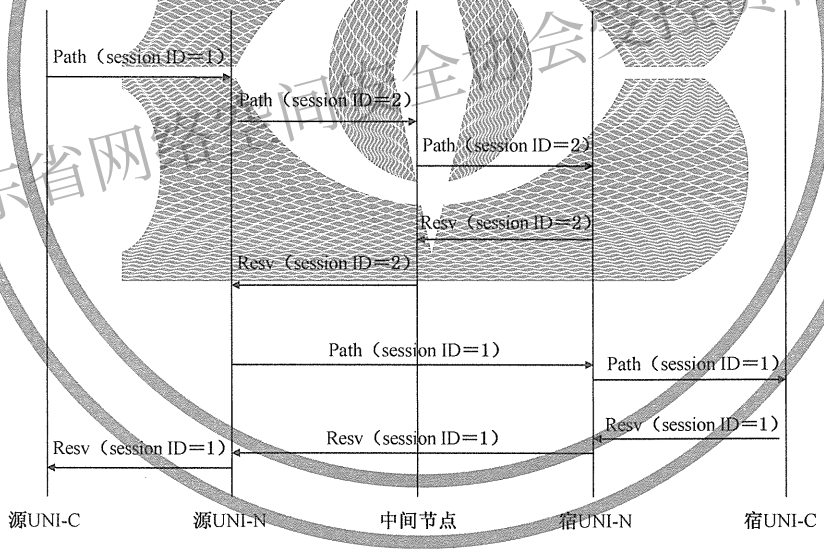


图 53　IETF GMPLS UNI 嵌套/拼接 LSP 建立过程

这种情况下，存在源 UNI-C 和宿 UNI-C 间的一个会话及源 UNI-N 到和宿 UNI-N 间的一个会话。当源 UNI-C 发送的 Path 消息到达源 UNI-N 时，源 UNI-N 查找核心网络中是否存在从源 UNI-N 到宿 UNI-N 的 LSP 可作为转发邻接(FA)提供给 UNI 连接，如果存在，该 FA 作为 UNI 连接中的一条，Path 消息将直接从源 UNI-N 发送到宿 UNI-N，接着由宿 UNI-N 发送到宿 UNI-C；Resv 消息沿着相同的路径返回，到达源 UNI-C，UNI 连接建立完成。如果 UNI-N 间无可用的 FA，在源 UNI-N 处，源 UNI-C 发送过来的 Path 消息将被暂时保持，由源 UNI-N 发起建立到宿 UNI-N 的 LSP(一个新的 SESSION)，该 LSP 建立完成后，作为 FA 提供给 UNI 连接，原来在 UNI-N 处保持的 Path 消息将新建的 FA 作为连接中的一跳，发往宿 UNI-N；Resv 消息沿着相同的路径返回，到达源 UNI-C，UNI 连接建立完成。核

心网络中采用嵌套还是拼接的方式提供 FA 取决于核心网络和用户网络的交换类型及交换粒度等因素。对于用户侧和网络侧使用不同交换类型的网络，应使用嵌套的 LSP，而且网络内部的一条 LSP 连接可以被多个 UNI 连接所使用。对于嵌套 LSP 更为详细的信息可参考 IETF RFC4206。对于用户侧和网络侧使用相同交换类型的网络，可以使用拼接 LSP 的方式，但网络内部的一条 LSP 只能被一条 UNI 连接所使用。对于拼接 LSP 更为详细的信息可参考 IETF RFC5150。

在 UNI-C 发起的 Path 消息中，SENDER_TEMPLATE 中的源地址为源 UNI-C 的节点 ID 或源 UNI-C 到源 UNI-N 间的一条有编号 TE 链路的 IP 地址，SESSION 对象中的目的地址为目的 UNI-C 的节点 ID 或目的 UNI-C 到目的 UNI-N 间的一条有编号 TE 链路的 IP 地址。源 UNI-C 发送的 Path 消息的 ERO 中可携带显示标签对象用于控制出口链路的标签选择。

8.3.3.2 连接删除

IETF 中的连接删除流程和“8.1.3.6 UNI 连接删除过程”流程一致。区别在于内部对会话的处理。OIF UNI 中要删除源 UNI-C 到源 UNI-N，源 UNI-N 到宿 UNI-N，宿 UNI-N 到宿 UNI-C 三段会话；IETF 中如果采用连续信令方式创建 UNI 连接，删除时只需删除源 UNI-C 到宿 UNI-C 的会话；如果采用嵌套/拼接方式创建的 UNI 连接，删除时要删除源 UNI-C 到宿 UNI-C 的会话，如果源 UNI-N 到宿 UNI-N 的 LSP 是由 UNI 连接建立信令触发创建的，则需删除该 LSP，否则该 LSP(FA)保留。

8.3.3.3 连接修改

对于以太网业务，支持无损的连接带宽修改，修改的方法可参考 IETF RFC3209 中 2.5 定义的“先建后拆”的方式进行连接的修改。

8.3.3.4 UNI 出口标签控制

本节采用 IETF RFC4003 标准来确定目的 UNI-N 如何进行出口标签的控制处理。

源 UNI-N 通过在路径建立消息中设置目的 UNI-N 的显式标签对象的端口子对象(见 IETF RFC3477 中 4)和标签子对象(见 IETF RFC3473 中 5.1)来控制目的 UNI-N 到 UNI-C 的端口和标签的选择。目的 UNI-N 根据 ERO 信息来确定 UNI 的出端口和标签信息。详细的处理参考 IETF RFC4003。

9 UNI 策略和安全(可选)

9.1 UNI 策略控制

传送网络应提供适当的机制来保证对传送网络的正确和授权使用以及客户的相关责任。总体来说，这些机制通常为策略控制。基于策略的标准也可以应用于资源的可用性考虑，即，传送网络内部决定是否为连接请求提供资源。策略控制规则可以根据参数条件来定义，如，源和目的地址、优先级、服务提供的双边协商、时间约束、费用约束等。策略控制也通常和相关的计费机制进行关联。对于初始的部署，策略控制使用简单的规则，如：对于一个给定的 UNI-C 代理，如果接收到的请求代表给定的用户组并且请求者的标识能够通过验证，则批准所有的请求。当从初始的部署中积累更多的运营经验后，策略规则可以变得更加成熟。

为支持策略控制，需要两个主要的结构实体：确定策略的策略决定点(PDP)、实际执行策略的策略执行点(PEP)。PEP 驻留在传送网络内部节点中。PDP 的驻留位置要依赖下列的多个因素：

a) 策略规则的复杂性，包括计算负载，支持的软件类型以及数据接入需求。如，接入物理上驻留在远端服务器中的信用数据库。这时，PDP 应该驻留在远端服务器中。

b) 需要策略决定的事件频繁程度。如，连接建立请求可能不太频繁，因而减少了 PDP 的计算复

杂度；

c) 传送网络设备的智能化和弹性化。成熟以及容易升级的传送网络节点可以作为 PDP 主机的更好的候选者。

上面这些因素的组合确定 PDP 是否应该驻留在 UNI-N 代理或在远端策略服务器上执行。如果使用外部策略服务器，PEP 和 PDP 通信时需要使用标准化的协议。这允许从单个 PDP 管理多个 PEP，并促进策略服务器和多个传送网元合并。COPS 协议 IETF RFC2748 已经被 IETF 作为 PEP-PDP 之间的通信标准。RSVP 信令中的 POLICY_DATA 对象用于携带策略信息。

UNI 协议的指定不依赖 PEP 和 PDP 模型在传送网络内部的位置。是否需要外部 PDP 依赖于：上述几个因素，即，策略决定的频度和和复杂度以及传送网络节点的处理能力。如果需要外部 PDP，推荐使用 COPS 协议。

如图 54 所示，PDP 将来可以接入外部服务器或数据库，如，获取策略规则，集中存储计费信息等。这些额外的机制不在 OIF 的研究范围。

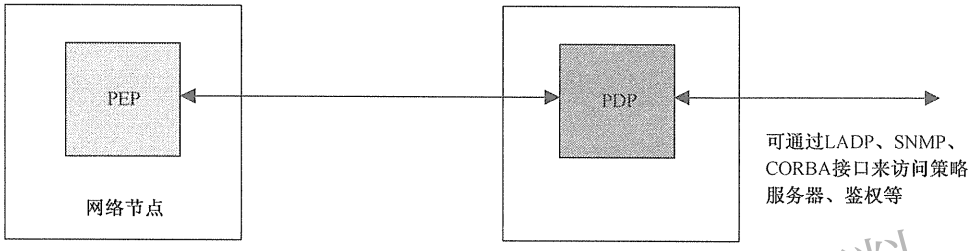


图 54 PEP 和 PDP

9.2 连接指配的策略应用样例

9.2.1 基于日期的指配

用户和运营商之间的合同允许在一定的时间周期内(一天中的小时范围和/或一周之内的几天)提供连接请求服务，如，在夜间为存储区域网络提供数据备份的连接服务。这种情形下，UNI-N 代理需要验证所接收到的连接请求是否在合同限定的时间之内。策略决定所需要的全部信息(收到请求的时间)隐式地包含在 UNI 信令信息中。

9.2.2 连接请求者的身份和信用验证

运营商运行的带宽交换可以允许运营商进行动态的光路连接服务贸易。带宽交换接收来自大量运营商的请求。代表不同运营商的多个 UNI-C 代理可能和运营商的运营商网络内部相同的 UNI-N 代理连接。对于运营商的运营商能够验证请求发起者的身份是非常必要的。另外，请求者支付服务费用的能力也可能需要验证。这需要如帐户编号或信用授权相关的信息。UNI-N 侧基于策略的许可控制涉及请求者身份的肯定验证以及信用的验证。这种情形下，需要用来做策略决定的信息可以对必选属性进行扩展，参考 IETF RFC2752 所描述的当使用 RSVP-TE 作为 UNI 信令协议时的身份表示。

9.2.3 基于使用的计费

基于使用的计费可以通过使用合同 ID 来支持，这里指的是连接拥有者的服务合同。合同 ID 可以由运营商指定，可以用来计费，帐单和 SLA 验证。由于包含敏感的信息，合同 ID 需要加密来保护发起者的隐私。

9.3 UNI 信令协议相关的策略控制机制

RSVP-TE 定义了策略对象，该对象用来映射经过 UNI 接口用于策略控制的可选对象。携带合同 ID 的策略数据对象的使用见 IETF RFC3473。该对象以后可以用来携带和策略相关的其他数据。

9.4 UNI 安全考虑

因为光路连接承载大量的数据并消耗主要的网络资源，需要使用安全机制来保证传送网络防止对控制平面的攻击或未受权时对网络资源的使用。

通信机制通常需要使用两种主要的安全机制：鉴权和加密。鉴权机制确保对数据的发起源进行鉴权并对 UNI 信令信息进行完整性验证，以便检测出并拒绝未授权的 UNI 操作。如，UNI 消息的鉴权服务可以阻止恶意的 UNI-C 代理通过插入大量的未授权连接创建请求来攻击网络服务提供者。另外，鉴权机制可以提供：1、重放保护，即，通过侦测和拒绝尝试重新排列、复制、截断或其他使用适当的消息序列进行篡改等；2、抗抵赖，即为计费和帐单目的。鉴权和加密可以使用对称或公钥加密算法来实现。简单的消息完整性验证和加密可以使用对称密码算法来实现。这些算法典型地需要一对共享地密钥。该算法不提供抗抵赖功能，但不要求复杂的计算。重放保护通常可以通过在消息中添加序列号或依靠其他协议（如 TCP）来保证消息流序列地完整性服务。公钥或非对称密码算法最初典型地用来提供两方对等实体的鉴权和密钥协商，它促进了上面所描述的完整性和加密的应用。非对称密码算法也用作数字签名，用来执行抗抵赖服务。可以通过公共密钥基础设施（PKI）或其他共同定义地密钥分配机制来支持非对称密码算法的使用。非对称密码算法相比对称密码算法有更大的计算复杂度。从 UNI 需求的观点来看，消息鉴权是最重要的安全特征。UNI 消息的加密也是希望的，尤其是当 UNI 消息属性中包含通信双方（客户和传送网络操作者）的私密信息时需要进行消息加密。这些属性信息包括如，帐户编号信息、合同标识编号信息等。

非本地设备的出现增加了安全和策略控制的需求。这种情形下，假定 UNI-C 和 UNI-N 节点通过网络设备相连，如 2 层交换和 IP 路由器。由于这些设备可以属于不同的网络运营商并可能超出服务提供商的控制范围，UNI-C 和 UNI-N 之间的控制通信安全威胁增加，如 IP 地址欺骗、窃听和未受权的试图闯入等。为考虑这些威胁，应该使用适当的安全机制来保护 UNI 信令和控制通道。

安全机制的使用增加了对资源的消耗，安全机制的设计要考虑成本和收益间的均衡。因此，协议的安全机制应该是：

a) 使用和实现是可选的：一些用户可能会使用其他方式提供足够的保护（例如：完善的接入控制和防火墙），这种情况下协议的安全性不太必要。设备供应商对于这种用户可提供无协议安全控制的产品。

b) 互操作性：在 UNI 中提供一组标准化的可互操作的安全服务，使得在一个运营商网络内的不同设备商间的设备可互通。因此，安全服务应具有尽可能少的方式、格式、可选特性和算法。

c) 和其他功能的协同：如果控制协议的安全特性（包括：信令，路由，自动发现）采用与承载业务、管理或 VPN 业务相同的方案，其安全特性的实现和部署会具有更低的成本及健壮性。

d) 高可靠性：如果安全协议已被标准化，进行过系统的分析和广泛的部署，则这种解决方案应被优先考虑。

e) 高质量：安全协议和算法的选择基于提供的安全级别，不能有缺陷和明显的弱点。安全机制的涉及能保障在较为广泛的主动和被动的攻击模式下得到验证。

f) 高效：标准的具有安全功能的处理器能处理多种验证和密钥协商协议。

总之，控制协议的安全机制应支持保密性，控制信息源端的鉴权，数据的完整性保证，基于每个消息的探测。

9.5 UNI 相关的安全机制

9.5.1 RSVP-TE 安全机制

RSVP-TE 加密验证协议 IETF RFC2747 中定义了完整性对象 INTEGRITY，该对象提供了对 RSVP-TE 逐跳消息的完整性和鉴权验证。INTEGRITY 对象包含消息的授权码 MAC，该授权码使用两个实体之间共享的密钥进行计算，并公用加密算法，如 HMAC-MD5 算法。作为明文密码机制，共享的密钥不会发送到网络之外。这可以防止截获攻击，这种攻击通过从物理媒介或数据包路径上截获信息后进行攻击。因而，该机制可以防止消息伪造或消息篡改。在每个 RSVP-TE 消息中，INTEGRITY 对象也使用一次性使用的编号来进行标识，这样可以使消息的接收者能够标识出重放并避免重放攻击。但不能检测消息删除。该机制也不能够提供任何加密，因为消息是明文格式。另外，由于它仅仅是基于对称的加密，它不能提供抗抵赖服务。标准的双方对等鉴权和密钥管理处理用来作为该机制的补充。不提供安全服务协商，因为协商是密钥管理架构中的一部分。可以对密钥进行手动管理。这种情形下特权用户手动配置鉴权密钥。IKE 或 Kerberos 也可以为该目的来使用。对后者的使用见 IETF RFC2747。

MD5 和 HMAC-MD5 加密算法被 IP 路由协议作为加密和鉴权算法，如 OSPF，IS-IS 和 BGP。使用该机制作为 RSVP-TE 的加密算法的一个优点是，相同或相似的机制以前被广泛使用，将会促进该机制的发展。

注：IETF RFC2747 中定义的 128 位 HMAC-MD5 和 IPsec[IETF RFC2403]中使用的 96 位版本有所不同。

9.5.2 邻居发现安全机制

邻居发现被认为是支持更高层协议的基本功能。同样，安全并没有作为邻居发现的一部分来考虑。任何安全相关的问题都是在邻居发现完成之后的才考虑。如果在执行自动发现相关的处理过程中要考虑安全问题，则建议禁止相关的处理而用手动配置邻居信息。

9.6 IP 安全协议(IPsec)

IPsec 为 IPv4 和 IPv6 在 IP 层提供不同的安全服务定义了一组协议。包括两个流量保护协议，IETF RFC2402 定义的鉴权头(AH)和 IETF RFC2406 定义的负载安全封装以及单个密钥管理协议，IETF RFC2409 和 IETF RFC2407 定义的互连网密钥交换协议。IPsec 提供的服务包括接入控制、无连接的完整性验证、数据源鉴权、重放保护和加密。一个重要的特征就是这些服务都在 IP 层提供，提供对 IP 以及 IP 上层协议的保护。同理，IPsec 可以应用于实现 UNI 信令的 RSVP-TE 协议，同样也保护本部分提到的其他协议如 LMP。

AH 提供面向无连接的完整性验证、数据源鉴权、可选的重放保护服务。ESP 提供包括 AH 所提供所有属性信息的加密。加密机制和提供的加密算法无关，IPsec 可以适应加密算法的扩展。IKE 是一种精密的对等实体之间的鉴权、密钥管理以及安全服务协商的协议，该协议包含多个加密算法和鉴权模式。

使用 IPsec 协议栈来保护 UNI 控制消息有许多优点。尤其需要指出的是，IPsec 可以为 RSVP-TE UNI 信令实现提供安全解决方案，为基于 LMP 的邻居发现协议提供加密。该机制并不是由两种信令传送机制来提供，还包含密钥管理协议。

同时也要看到，IPsec 是一个相对复杂和重量级的协议族，由于 AH 和 ESP 在 IP 层实现，IPsec 典型地需要对内核进行修改，潜在地使实现变得更加困难。而且，在可见的未来有多少 UNI 客户会支持 IPsec 还不清楚。

9.7 UNI 安全路标

UNI 使用底层信令传送机制(RSVP-TE)的加密鉴权选项。该规范的编撰是为了加速 UNI 的发展以及互连互通,还提供了广泛的部署以及相似加密机制的经验。这些机制提供对数据源的鉴权,对消息的完整性检验,因而防止服务攻击。

RSVP-TE 机制应该使用 128 位的 HMAC-MD5[IETF RFC2104]算法来替代 IETF RFC2385 中最初定义的 MD5 算法。目的是为了获得不同机制的共性并增强安全属性。应该注意的是,如,和最初的 MD5 加密算法相比,HMAC-MD5 明显提高了安全属性,但只略微增加了计算的复杂性。HMAC 架构也允许将来使用其他类似散列的功能(如,SHA-1,RIPEMD-160 以及 SHA-256)。

对更加灵活和包容的安全选项,如 IPsec,以及使实现更简单、更有效、更直截了当的机制的规格,可以在将来的 UNI 版本中考虑标准化。

9.8 UNI 安全扩展的使用

UNI 预先约定了 UNI 的安全扩展[SecExt]并且对 RSVP-TE 定义了上面描述的应用层安全机制。但是 UNI 并不排除使用任何底层网络层安全架构,如 IPsec。因此,SecExt 中描述的方法可以用于与 UNI 兼容的接口,和对 UNI 的修订或者扩展。

附 录 A
（资料性附录）
UNI 邻居发现实例

A.1 纤内 IPCC 配置情形下的邻居发现

考虑到客户和 TNE 之间通过多条数据链路连接情形，配置每条链路用来承载纤内控制通道业务，见 6.1 描述。本节中定义了每条通道的邻居发现处理。

a) 客户和 TNE 通过每条 IPCC 执行控制通道配置处理。每个处理过程中，交换接口 ID（在 CCID 域）信息，并验证配置（任何错连情形可以标识出来）。

b) 客户和 TNE 开始在每条 IPCC 上执行 Hello 协议。

A.2 纤外 IPCC 配置情形下的邻居发现

考虑客户和 TNE 之间通过多条组件链路连接情形，其中有一条或多条纤外 IPCC。见 5.4 的定义。本节中定义了每条通道的邻居发现处理。

a) 每一条 IPCC 和一个特定的邻居以及一组数据链路（等效于多个端口）关联。

b) 通过每条 IPCC，TNE 和客户之间执行控制通道配置处理。

c) 通过每条 IPCC，TNE 和客户之间开始执行 Hello 协议。

d) 客户选择其中一条连通到 TNE 的 IPCC，并通过这条 IPCC 发送 BeginVerify 消息。BeginVerify 消息中包含连续两次发送测试消息的时间间隔值（以毫秒为单位），名为 VerifyInterval。

e) 当 TNE 收到 BeginVerify 消息后，开始准备接收纤内测试消息：

   1) 选择一种测试消息的传送机制；

   2) 为测试处理分配唯一的标识 VerifyID；

   3) 确定一个名为 VerifyDeadInterval 的时间间隔（以毫秒为单位），该间隔用来等待接收测试消息；

   4) 将这些信息包含在 BeginVerifyAck 消息中并通过 IPCC 发回客户设备。

f) 当客户从网络设备接收到 BeginVerifyAck 消息后，开始向 TNE 周期性的发送测试消息，这些测试消息中包含接口 ID 和 VerifyID 并依次在每条被测链路的纤内进行发送。

g) 当收到测试消息后，TNE 记录所接收到的接口 ID 并通过 IPCC 通道返回 TestStatusSuccess 消息，其中包括客户接口 ID 和对应的 TNE 接口 ID。在 VerifyDeadInterval 时间间隔内，如果 TNE 没有检测到测试消息，TNE 通过 IPCC 通道向客户发送 TestStatusFailure 消息。

h) 通过 IPCC 接收到 TestStatusSuccess 消息后，客户设备能够检测出物理错连的情形。当没有发生错连时，客户设备将该链路标识为“Discovered”，并记录对应 TNE 的接口 ID。

当所有的链路测试完成后，客户通过 IPCC 发送 EndVerify 消息来指示测试完成。随后 TNE 通过 IPCC 发送 EndVerifyAck 消息。

当邻居发现处理完成后，用作发现处理的任何开销字节返回正常的操作。该方案本质上不影响原有的运行机制，并且允许对 SDH 信号的比特透明操作。

完成上述处理后，客户和 TNE 设备都维护一个完整的链路标识映射列表。如每个端口的远端接口 ID 和本地接口 ID。通过交换链路属性关联消息，可以对这些映射关系进行一致性检测。

A.3 实例 1

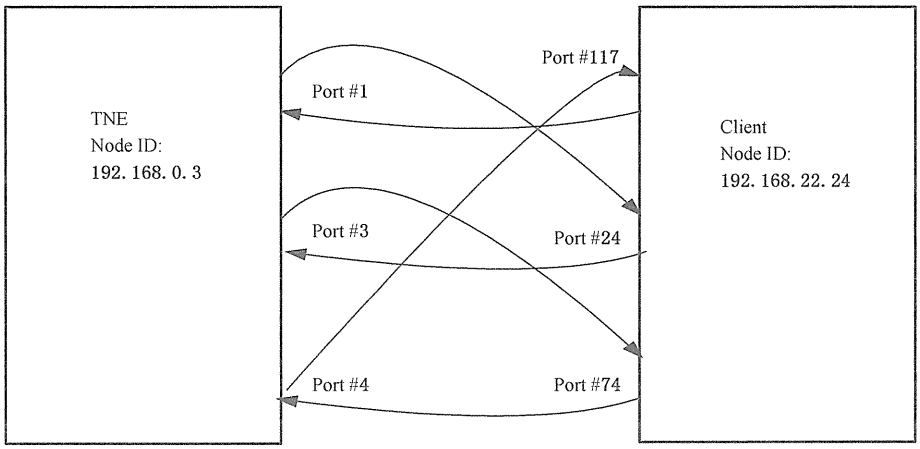


图 A.1 相同节点的端口错连检测

纤内配置：

图 A.1 实例中，当节点接收到的 ConfigAck 或 ConfigNack 消息中不包含期望的远端节点 I 或远端 CCID 时，可以检测出错连。描述如下：

第一步：TNE 从 #1 端口向客户发送 Config 消息（本地 CCID＝接口 ID＝ 端口 #1，本地节点 ID＝192.168.0.3）。该消息在客户端的 #24 端口被接收到。

第二步：客户从 #24 端口向 TNE 发送 ConfigAck 消息（本地 CCID＝接口 ID＝ 端口 #24，本地节点 ID＝192.168.22.24，远端 CCID＝端口 #1，远端节点 ID＝192.168.0.3）。该消息在 TNE 端的 #3 端口被接收到。

此时，TNE 检测出错连。

纤外配置：

如图 A.1 所示，当节点接收到的 TestStatusSuccess 消息中包含了不期望的接口 ID 时，检测出错连。注意，只有测试消息通过纤内传输，所有其他消息通过 IPCC 传送。这里，我们假定链路验证依次进行。下面描述的消息中所包含的标识信息包含在括号内。

TNE 发起处理流程：

第一步：TNE 向客户发送 BeginVerify 消息（包含所有为分配的数据链路，MessageID）；

第二步：客户向 TNE 发送 BeginVerifyAck 消息（验证 ID＝432，MessageIDAck）；

第三步：TNE 从 #1 端口向客户设备发送测试消息（验证 ID＝432，接口 ID＝端口 #1）。该消息被客户设备从 #24 端口收到；

第四步：客户向 TNE 发送 TestStatusSuccess 消息（验证 ID＝432，接口 ID＝端口 #24，远端接口 ID＝端口 #1，MessageID）；

第五步：TNE 向客户发送 TestStatusAck 消息（验证 ID＝423，MessageIDAck）；

第六步：#3 端口和 #4 端口重复 3～5 步；

第七步：TNE 向客户发送 EndVerify 消息（验证 ID＝423，MessageID）；

第八步：客户向 TNE 发送 EndVerifyAck 消息（验证 ID＝423，MessageIDAck）。

客户发起的测试处理类似于上面的步骤，定义如下：

第一步：客户向 TNE 发送 BeginVerify 消息（所有未分配的数据链路，MessageID）；

第二步：TNE 向客户发送 BeginVerifyAck 消息（验证 ID=727，MessageIDAck）；

第三步：客户从＃117 端口向 TNE 发送测试信号（验证 ID=727，接口 ID=端口＃117）。TNE 从＃1 端口接收到该消息；

第四步：TNE 向客户发送 TestStatusSuccess 消息（验证 ID=727，本地接口 ID=端口＃1，远端接口 ID=端口＃117，MessageID）；

第五步：客户向 TNE 发送 TestStatusAck 消息（验证 ID=727，MessageIDAck）；

第六步：＃24 端口和＃74 端口重复 3～5 步骤；

第七步：客户向 TNE 发送 EndVerify（验证 ID=727，MessageID）；

第八步：TNE 向客户发送 EndVerifyAck（验证 ID=727，MessageIDAck）。

至此，两端节点都检测到错连。

A.4 实例 2

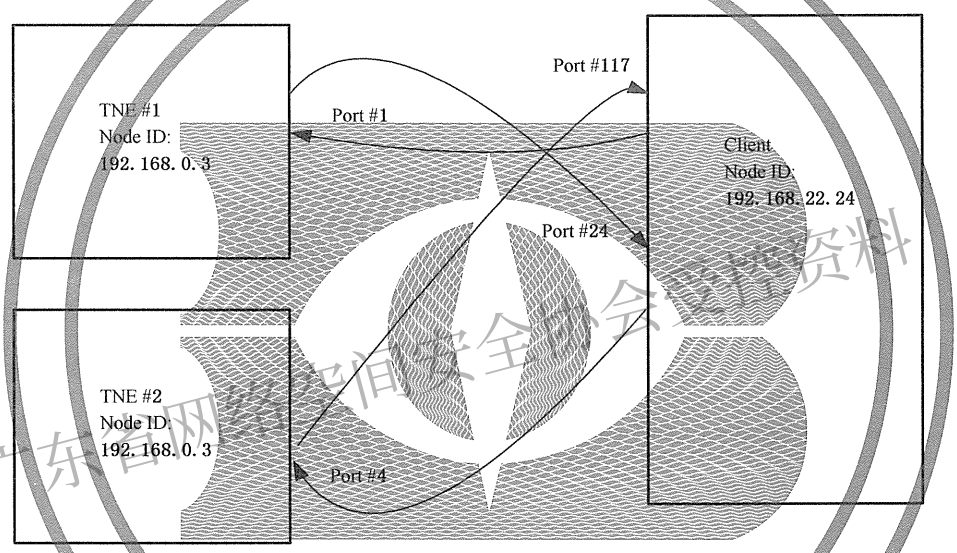


图 A.2 不同节点的端口错连检测

纤内配置：

图 A.2 实例中，TNE 节点发送 Config 后没有收到 ConfigAck（或 ConfigNack）响应消息，TNE 可以检测出错连。描述如下：

第一步：＃1TNE 从＃1 端口向客户发送 Config 消息（本地 CCID=接口 ID= 端口＃1，本地节点 ID=192. 168. 0. 3）。该消息在客户端的＃24 端口被接收到。

第二步：客户从＃24 端口向 TNE 发送 ConfigAck 消息（本地 CCID=接口 ID= 端口＃24，本地节点 ID=192. 168. 22. 24，远端 CCID=端口＃1，远端节点 ID=192. 168. 0. 3）。该消息在＃2TNE 的＃4 端口被接收到。

此时，＃1TNE 永远收不到 ConfigAck 消息，因此可以推断控制通道有问题。

纤外配置：

如图 A.2 所示，当节点接收到的 TestStatusSuccess 消息中包含了不期望的接口 ID 时，检测出错连。注意，只有测试消息通过纤内传输，所有其他消息通过 IPCC 传送。下面描述的消息中所包含的标识信息包含在括号内。

TNE 发起处理流程：

第一步：TNE 向客户发送 BeginVerify 消息（包含所有为分配的数据链路，MessageID）；

第二步：客户向 TNE 发送 BeginVerifyAck 消息（验证 ID＝200，MessageIDAck）；

第三步：TNE 从＃1 端口向客户设备发送测试消息（验证 ID＝200，接口 ID＝端口＃1）。该消息被客户设备从＃24 端口收到；

第四步：客户向 TNE 发送 TestStatusSuccess 消息（验证 ID＝200，接口 ID＝端口＃24，远端接口 ID＝端口＃1，MessageID）；

第五步：TNE 向客户发送 TestStatusAck 消息（验证 ID＝200，MessageIDAck）；

第六步：TNE 向客户发送 EndVerify 消息（验证 ID＝200，MessageID）；

第七步：客户向 TNE 发送 EndVerifyAck 消息（验证 ID＝200，MessageIDAck）。

客户发起的测试处理类似于上面的步骤，定义如下：

第一步：客户向 TNE 发送 BeginVerify 消息（所有未分配的数据链路，MessageID）；

第二步：TNE 向客户发送 BeginVerifyAck 消息（验证 ID＝350，MessageIDAck）；

第三步：客户从＃24 端口向 TNE 发送测试信号（验证 ID＝350，接口 ID＝端口＃24）。TNE 从＃1 端口接收到该消息；

第四步：等待 VerifyDeadInterval 毫秒后没有收到测试消息，TNE 向客户发送 TestStatusFailure 消息（验证 ID＝350，MessageID）；

第五步：客户向 TNE 发送 TestStatusAck 消息（验证 ID＝350，MessageIDAck）；

第六步：客户向 TNE 发送 EndVerify（验证 ID＝350，MessageID）；

第七步：TNE 向客户发送 EndVerifyAck（验证 ID＝350，MessageIDAck）。

至此，两端节点都检测到错连。

A.5　实例 3

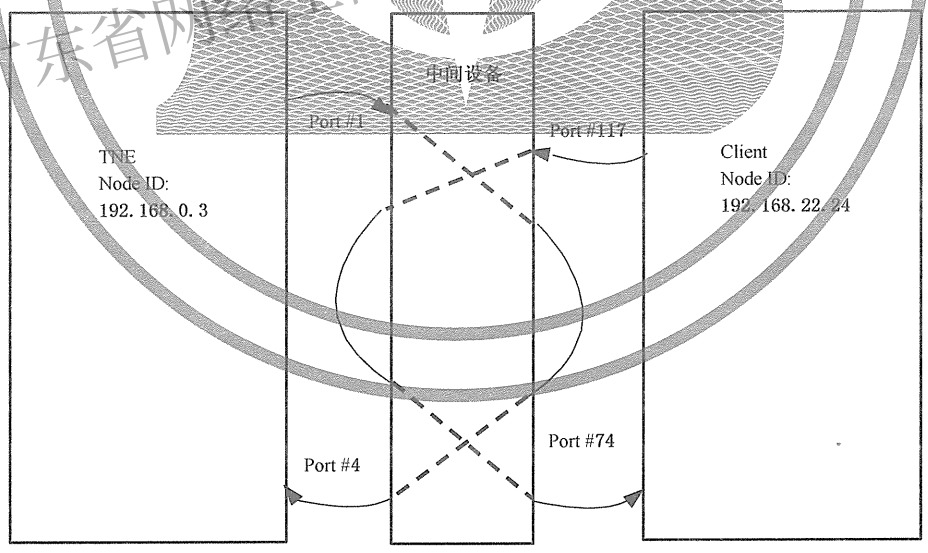


图 A.3　端口环回错连检测

纤内配置：

图 A.3 实例中，TNE 节点接收到的 Config 消息中的节点 ID 号和本地节点 ID 号相同。检测出连接出错。描述如下：

第一步：＃1TNE 从＃1 端口向客户发送 Config 消息（本地 CCID＝接口 ID＝ 端口＃1，本地节点 ID＝192. 168. 0. 3）。该消息由＃1TNE 本身从＃4 端口接收到。

类似地

第一步：客户从#117端口向TNE发送Config消息(本地CCID=接口ID= 端口#117，本地节点ID=192.168.22.24)。该消息由客户本身从#74端口接收到。

此时，TNE和客户都发现接收到的节点ID信息和本地相同，因此可以确定配置出错。

纤外配置：

如图A.3所示，当节点接收到TestStatusFailure消息后，检测出连接出错。注意，只有测试消息通过纤内传输，所有其他消息通过IPCC传送。下面描述的消息中所包含的标识信息包含在括号内。

TNE发起处理流程：

第一步：TNE向客户发送BeginVerify消息(包含所有为分配的数据链路，MessageID)；

第二步：客户向TNE发送BeginVerifyAck消息(TNA地址，验证ID=75，MessageIDAck)；

第三步：TNE从#1端口向客户设备发送测试消息(验证ID=75，接口ID=端口#1)。该消息被TNE从#4端口接收到；

第四步：TNE等待VerifyDeadInterval毫秒后还没有收到测试消息时，向客户发送TestStatusFailure消息(验证ID=75，MessageID)；

第五步：TNE向客户发送TestStatusAck消息(验证ID=75，MessageIDAck)；

第六步：TNE向客户发送EndVerify消息(验证ID=75，MessageID)；

第七步：客户向TNE发送EndVerifyAck消息(验证ID=75，MessageIDAck)。

客户发起的测试处理类似于上面的步骤，定义如下：

第一步：客户向TNE发送BeginVerify消息(所有未分配的数据链路，MessageID)；

第二步：TNE向客户发送BeginVerifyAck消息(验证ID=42，MessageIDAck)；

第三步：客户从#117端口向TNE发送测试信号(验证ID=42，本地接口ID=端口#117)。客户本身从#74端口接收到该消息；

第四步：等待VerifyDeadInterval毫秒后没有收到测试消息，TNE向客户发送TestStatusFailure消息(验证ID=42，MessageID)；

第五步：客户向TNE发送TestStatusAck消息(验证ID=42，MessageIDAck)；

第六步：客户向TNE发送EndVerify(验证ID=42，MessageID)；

第七步：TNE向客户发送EndVerifyAck(验证ID=42，MessageIDAck)。

至此，两端节点都检测到错连。

附 录 B
（资料性附录）
UNI 业务流量参数实例

B.1 SDH 业务连接请求的流量参数实例

各种典型 LSP 信号类型应用举例如下：

- UNI 请求的 LSP 为一条 VC-4 信号业务；
- UNI 请求的 LSP 为一条由 7 个 VC-4 信号构成的虚级联业务；
- UNI 请求的 LSP 为一条由 16 个 VC-4 信号构成的标准连续级联业务。

上述 3 种典型信号对应的 SDH 流量参数编码值如表 B.1：

表 B.1 SDH 流量参数编码值

编号	LSP 信号	ST	RCC	NCC	NVC	MT	T	P
1	VC-4	6	0	0	0	1	0	0
2	VC-4-7v	6	0	0	7	1	0	0
3	VC-4-16c	6	1	16	0	1	0	0

B.2 OTN 业务连接请求的流量参数实例

各种典型 LSP 流量参数应用举例如下：

- UNI 请求的 LSP 为一条 ODU1 信号业务；
- UNI 请求的 LSP 为一条由 3 个 ODU1 信号构成的虚级联业务；
- UNI 请求的 LSP 为一条速率为 10G 的 OCh 波长业务。

上述 3 种典型信号对应的 OTN 流量参数编码值如表 B.2：

表 B.2 OTN 流量参数编码值

编号	LSP 信号	ST	RCC	NCC	NVC	MT	T	P
1	ODU1	1	0	0	0	1	0	0
2	ODU1-3v	1	0	0	3	1	0	0
3	OCh	7	0	0	0	1	0	0

参 考 文 献

[1] IETF RFC2104 HMAC:建入-散列法用于信息身份验证
[2] IETF RFC2205 RSVP 功能规范 V1
[3] IETF RFC2385 BGP 会话的保护
[4] IETF RFC2402 IP 鉴权头部
[5] IETF RFC2403 在 ESP 和 AH 中使用 HMAC-MD5-96
[6] IETF RFC2406 IP 封装安全有效载荷(ESP)
[7] IETF RFC2407 Internet IP 用于解释 ISAKMP 的安全域
[8] IETF RFC2409 Internet 密钥交换(IKE)
[9] IETF RFC2615 SONET/SDH 承载 PPP
[10] IETF RFC2747 RSVP 加密鉴权
[11] IETF RFC2748 通用开放式策略服务协议
[12] IETF RFC2752 RSVP 的策略控制
[13] IETF RFC2961 RSVP 对减少刷新开销的扩展
[14] IETF RFC3209 RSVP 扩展支持 LSP 隧道
[15] IETF RFC3471 GMPLS 信令功能描述
[16] IETF RFC3473 GMPLS RSVP-TE 信令扩展
[17] IETF RFC3474 RSVP-TE 扩展支持 ASON 及应用
[18] IETF RFC3477 RSVP-TE 对无编号链路的支持
[19] IETF RFC4003 GMPLS 信令支持出口标签控制
[20] IETF RFC4204 链路管理协议
[21] IETF RFC4206 GMPLS 对层次 LSP 的支持
[22] IETF RFC4328 GMPLS 支持对 G.709 OTN 网络控制的扩展
[23] IETF RFC4606 GMPLS 支持对 SDH 网络控制的扩展
[24] IETF RFC4974 GMPLS RSVP-TE 信令支持呼叫的扩展
[25] IETF RFC4990 GMPLS 网络地址的使用
[26] IETF RFC5150 GMPLS 对拼接 LSP 的支持
[27] ITU-T G.707 基于同步数字系列(SDH)的网络节点接口
[28] ITU-T G.805 传送网通用体系结构
[29] ITU-T G.7713 分布式呼叫和连接管理
[30] ITU-T G.8080 自动交换传送网体系结构
[31] ITU-T X.213 信息技术-开放系统互联-网络业务定义
[32] OIF-UNI-1.0-R2-RSVP 用户-网络接口 1.0 RSVP 信令规范(版本 2)
[33] OIF-UNI-2.0-RSVP 用户-网络接口 2.0 RSVP 信令规范
[34] IEEE 802.3ah 第一公里以太网

GB/T 21645.5-2012

*
书号:155066·1-45708