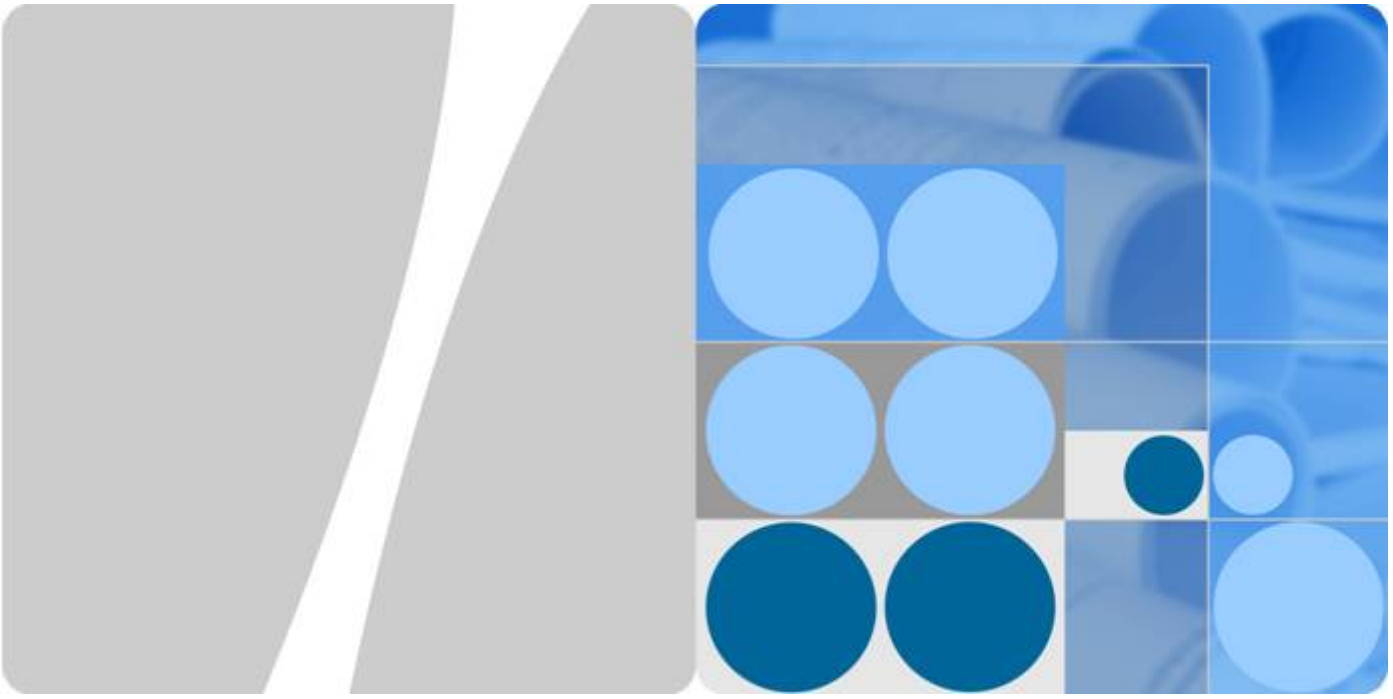


Huawei AR3200 系列企业路由器

V200R002C00

特性描述-网络管理

文档版本　01

发布日期　2011-12-30

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

前 言

读者对象

本文档针对网络管理特性，从简介、原理描述和应用三个方面介绍了网络管理特性。

本文档与其它类型手册相结合，便于读者深入掌握特性的实现原理。

本文档主要适用于以下工程师：

- 网络规划工程师
- 调测工程师
- 数据配置工程师
- 系统维护工程师

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
危险	以本标志开始的文本表示有高度潜在危险，如果不能避免，会导致人员死亡或严重伤害。
警告	以本标志开始的文本表示有中度或低度潜在危险，如果不能避免，可能导致人员轻微或中等伤害。
注意	以本标志开始的文本表示有潜在风险，如果忽视这些文本，可能导致设备损坏、数据丢失、设备性能降低或不可预知的结果。
窍门	以本标志开始的文本能帮助您解决某个问题或节省您的时间。
说明	以本标志开始的文本是正文的附加信息，是对正文的强调和补充。

命令行格式约定

格式	意义
粗体	命令行关键字（命令中保持不变、必须照输的部分）采用**加粗**字体表示。
斜体	命令行参数（命令中必须由实际值进行替代的部分）采用*斜体*表示。
[ ]	表示用“[ ]”括起来的部分在命令配置时是可选的。
{ x | y | ... }	表示从两个或多个选项中选取一个。
[x | y | ...]	表示从两个或多个选项中选取一个或者不选。
{ x | y | ... } *	表示从两个或多个选项中选取多个，最少选取一个，最多选取所有选项。
[x | y | ...] *	表示从两个或多个选项中选取多个或者不选。
&<1-n>	表示符号&前面的参数可以重复 1 ～ n 次。
#	由“#”开始的行表示为注释行。

修订记录

修改记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

文档版本 01 (2011-12-30)

第一次正式发布。

目 录

1 SNMP

关于本章

1.1 介绍

定义

SNMP（Simple Network Management Protocol，简单网络管理协议）是广泛用于 TCP/IP 网络的网络管理标准协议。SNMP 提供了一种通过运行网络管理软件的中心计算机（即网络管理工作站）来管理网元的方法。

SNMP 有以下几种操作方式：

- 管理工作站通过 GET，GET-NEXT 操作来获取网络资源信息；
- 管理工作站通过 SET 来设置网络资源；
- 管理代理主动上报 TRAP 给管理工作站，可使管理工作站及时获取网络状态，从而使网络管理员能够及时采取相应措施。

SNMP 的版本演进：

1990 年 5 月，RFC 1157 定义了 SNMP 的第一个版本 SNMPv1。RFC 1157 提供了一种监控和管理计算机网络的系统方法。SNMPv1 基于团体名认证，安全性较差，且返回报文的错误码也较少。

后来，IETF 颁布了 SNMPv2p。SNMPv2p 为了解决安全问题，引入参与者的概念。但由于实际应用中出现的问题，没有得到推广。之后颁布的 SNMPv2c 取代了 SNMPv2p，去掉了参与者的概念，但仍然沿用 SNMPv1 中的团体名进行安全认证。SNMPv2c 中引入了 getbulk 操作，并且提供更多的错误码信息。

鉴于 SNMPv2c 在安全性方面没有得到改善，IETF 颁布了 SNMPv3 的版本，提供了基于 USM（User Security Module）的认证加密和基于 VACM（View-based Access Control Model）的访问控制。

华为技术有限公司的产品支持 SNMPv1、SNMPv2c 和 SNMPv3。

目的

SNMP 的主要目的是网络管理。

网络管理分为两类：

- 第一类是对网络应用程序、用户帐号（例如文件的使用）和存取权限（许可）的管理。它们都是与软件有关的网络管理问题。这里不作深入解释。
- 第二类是对构成网络的硬件即网元的管理，包括工作站、服务器、网卡、路由器和集线器等等。通常情况下，这些设备与网络管理员所在的中心机房所在地距离很远。当这些设备有问题发生时，如果网络管理员可以自动地被通知，那么无疑是最佳的。但是路由器不会像用户那样，当有一个应用程序发生问题时会打电话通知。

为了解决这个问题，设备制造商已经在一些设备中提供了网络管理的功能，这样网络管理工作站就可以远程询问设备的状态，同样设备能够在特定类型的事件发生时向网络管理工作站发出警告。

网络管理通常被分为四个部分：

- 被管理节点：即被监视的设备，简称网元。
- 代理：用来跟踪被管理设备状态的特殊软件或固件(firmware)。

- 网络管理工作站：与不同的被管理节点中的代理通信，并且显示这些代理状态的中心设备。
- 网络管理协议：被网络管理工作站和代理用来交换信息的协议。

受益

企业网用户可以通过 SNMP 协议对网络中的设备进行管理和监控，简化了管理网络的部署，降低了用户的操作运行费用。

1.2 参考标准和协议

本特性的参考资料清单如下：

文档	描述	备注
RFC 1157	Simple Network Management Protocol (SNMP)	-
RFC 1905	Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)	-
RFC 2271	An Architecture for Describing SNMP Management Frameworks	-
RFC 3410	Introduction and Applicability Statements for Internet Standard Management Framework	-
RFC 3411	An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks	-
RFC 3412	Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)	-
RFC 3413	Simple Network Management Protocol (SNMP) Applications	-
RFC 3414	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)	-
RFC 3415	View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)	-
RFC 3418	Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)	-
RFC 2578	Structure of Management Information Version 2 (SMIv2)	-
RFC 2579	Textual Conventions for SMIv2	-
RFC 2580	Conformance Statements for SMIv2	-

1.3 可获得性

涉及网元

需要网管配合。

License 支持

无需获得 License 许可，均可获得该特性的服务。

版本支持

产品	最低支持版本
AR3200	V200R001C00

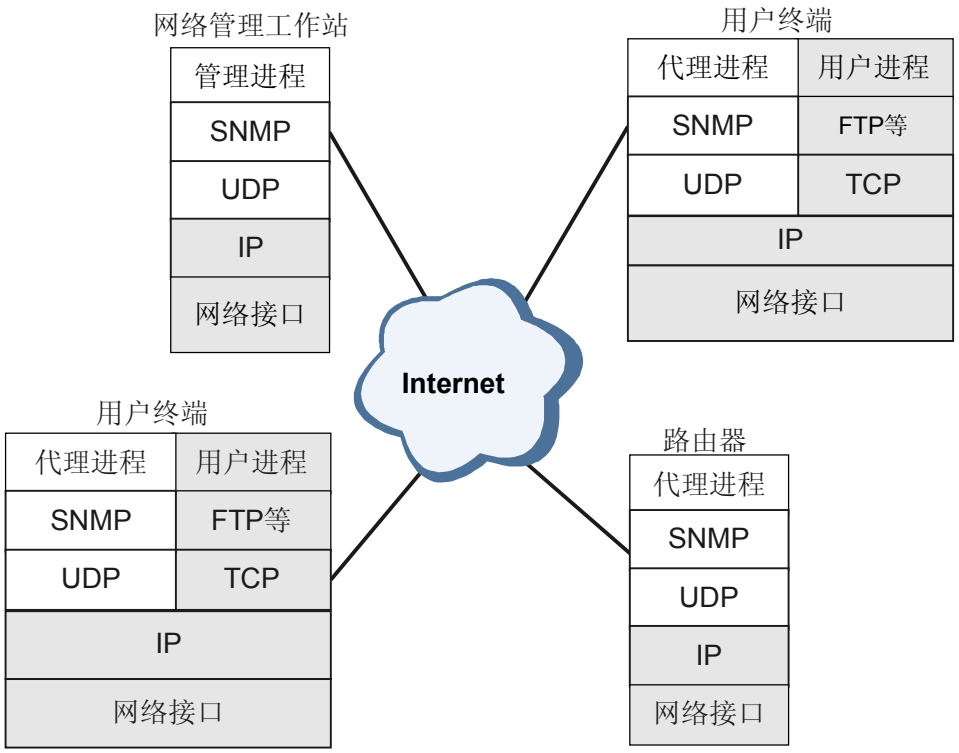
1.4 原理描述

1.4.1 SNMP 基本原理

如图 1-1 所示是使用 SNMP 的典型配置。整套系统必须有一个网管工作站（management station），它是整个网络的网管中心，在它之上运行管理进程。

每个被管对象中一定要有代理进程。管理进程和代理进程利用 SNMP 报文进行通信，SNMP 报文使用 UDP 作为运载协议。

图 1-1 SNMP 典型配置



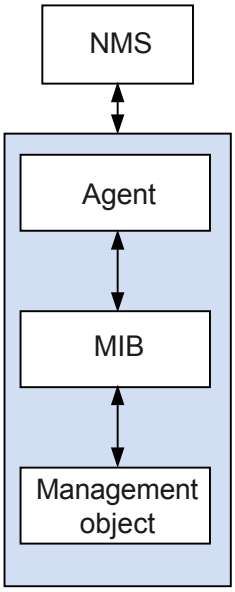
1.4.2 SNMP 管理模型

SNMP 的管理体系，在 NMS 和 Agent 两侧进行信令交互。

- 网管端工作站上的 NMS 作为管理者，向 Agent 发送 SNMP 请求报文。
- Agent 通过查询设备端的 MIB 得到所要查询的信息，向 NMS 发送 SNMP 响应报文。
- 设备端的模块由于达到模块定义的告警触发条件，通过 Agent 向网管端工作站的 NMS 发送 Trap 消息，告知设备侧的出现的情况，这样便于网络管理人员及时的对网络中出现的情况进行处理。

网络管理模型如图 1-2 所示。

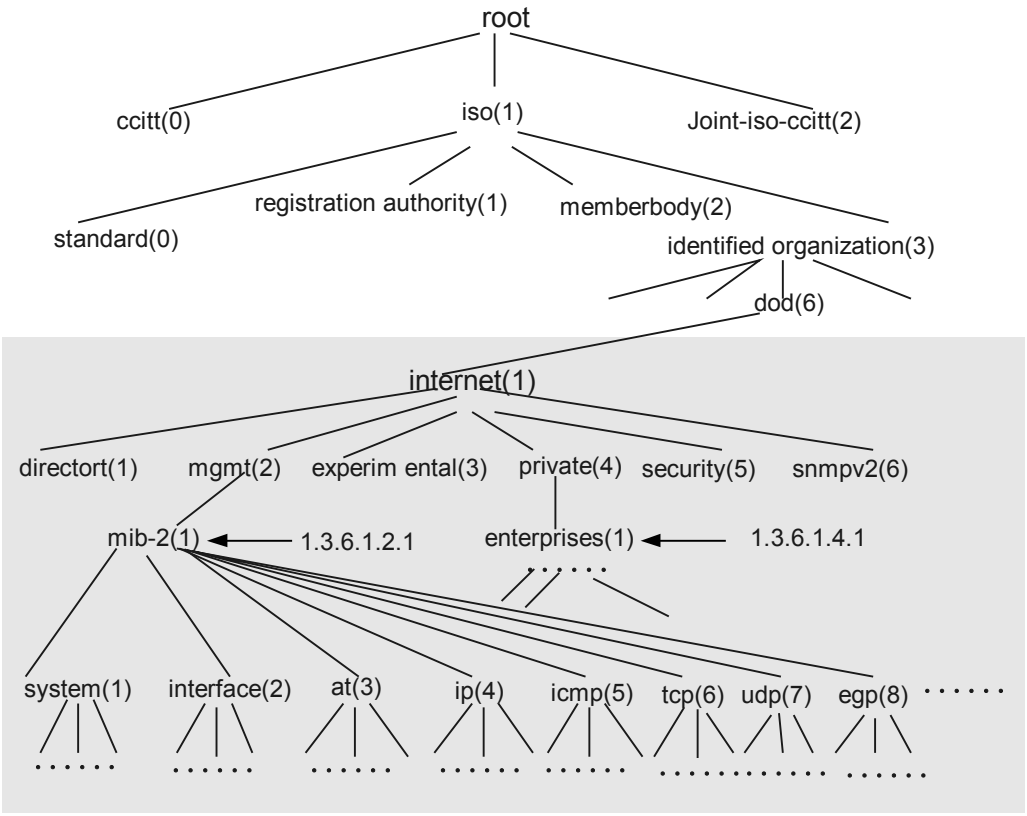
图 1-2 SNMP 管理模型



1.4.3 MIB

MIB（Management Information Base，管理信息库）指明了网络元素所维护的变量（即能够被管理进程查询和设置的信息）。MIB 给出了一个数据结构，包含了网络中所有可能的被管理对象的集合。SNMP 的管理信息库采用和域名系统 DNS 相似的树型结构，它的根在最上面，根没有名字。如图 1-3 所示的是管理信息库的一部分，它又称为对象命名（object naming）树。

图 1-3 MIB 树结构



对象命名树的顶级对象有三个，即 ISO、ITU-T（即 ccitt）和这两个组织的联合体。在 ISO 的下面有 4 个结点，其中的一个（标号 3）是标识组织 organization。在其下面有一个美国国防部 DoD（Department of Defense）的子树（标号是 6），再下面就是 Internet（标号是 1）。在只讨论 Internet 中的对象时，可只画出 Internet 以下的子树（图中带阴影的虚线方框），并在 Internet 结点旁边标注上{1.3.6.1}即可。

在 Internet 结点下面的第二个结点是 mgmt（管理），标号是 2。再下面是管理信息库，原先的结点名是 mib。1991 年定义了新的版本 MIB-II，故结点名现改为 MIB-II，其标识为{1.3.6.1.2.1}，或{Internet(1) .2.1}。这种标识为对象标识符。

最初的结点 MIB 将其所管理的信息分为 8 个类别，见**表 1-1**。现在的 MIB-II 所包含的信息类别已超过 40 个。

表 1-1 MIB 节点的管理信息类别

类别	标号	所包含的信息
system	1	主机或路由器的操作系统
interface	2	各种网络接口及它们的测定通信量
address translation	3	地址转换（例如 ARP 映射）
ip	4	Internet 软件（IP 分组统计）

类别	标号	所包含的信息
icmp	5	ICMP 软件（已收到 ICMP 消息的统计）
Tcp	6	TCP 软件（算法、参数和统计）
udp	7	UDP 软件（UDP 通信量统计）
egp	8	EGP 软件（外部网关协议通信量统计）

MIB 的定义与具体的网络管理协议无关。设备制造商可以在产品中包含 SNMP 代理软件，并保证在定义新的 MIB 项目后该软件仍遵守标准。用户可以使用同一网络管理客户软件来管理具有不同版本的 MIB 的多个路由器。若一台路由器上不支持此 MIB，那么就无法提供相应的功能。

1.4.4 SMI

管理信息结构 SMI（Structure of Management Information）为命名和定义管理对象指定了一套规则，可定义被管对象的对象标识、对象类型、访问级别和状态。

目前 SMI 有两个版本，SMIv1 和 SMIv2。

以下是 SMI 中定义的标准数据类型：

- INTEGER

  整型变量取值范围不定。例如，IP 的转发标志只有允许转发或者不允许转发两种取值，而 UDP 和 TCP 的端口号取值范围是 0 到 65535。

- OCTER STRING

  包含 0 或多个字节，每个字节值在 0 ～ 255 之间。

- DisplayString

  包含 0 或多个字节，每个字节必须是 ASCII 码。在 MIB-II 中，DisplayString 类型的变量不能超过 255 个字符（可以为 0 个字符）。

- OBJECT IDENTIFIER

  对象的唯一标识符，由整数序列组成，称为子标识符。该序列从左至右表示对象在 MIB 树形结构中的位置。

- NULL

  代表相关的变量没有值。例如，在 get 或 get-next 操作中，变量的值就是 NULL。

- IpAddress

  4 字节长度的 OCTER STRING，以网络序表示的 IP 地址。每个字节代表 IP 地址的一个字段。

- PhysAddress

  OCTER STRING 类型，代表物理地址（例如以太网物理地址为 6 个字节长度）。

- Counter

  非负的整数，可从 0 递增到 4294976295。达到最大值后归 0。

- Gauge

非负的整数，取值范围是从 0 ～ 4294976295(或增或减)。达到最大值后锁定直到复位。例如，MIB 中的 tcpCurrEstab 就是这种类型的变量，它代表目前在 ESTABLISHED 或 CLOSE_WAIT 状态的 TCP 连接数。

- TimeTicks

  时间计数器，以 0.01 秒为单位递增，但是不同的变量可以有不同的递增幅度。所以在定义这种类型的变量的时候，必须指定递增幅度。例如，MIB 中的 sysUpTime 变量就是这种类型的变量，代表代理进程从启动开始的时间长度，以多少个百分之一秒的数目来表示。

- SEQUENCE

  这一数据类型与 C 程序设计语言中的“structure”类似。一个 SEQUENCE 包括 0 个或多个元素，每一个元素又是另一个 ASN.1 数据类型。例如，MIB 中的 UdpEntry 就是这种类型的变量。它代表在代理进程侧目前“激活”的 UDP 数量（“激活”表示目前被应用程序所用）。在这个变量中 包含两个元素：

  - IpAddress 类型中的 udpLocalAddress，表示 IP 地址。
  - INTEGER 类型中的 udpLocalPort，从 0 到 65535，表示端口号。

- SEQUENDEOF

  这是一个向量的定义，其所有元素具有相同的类型。如果每一个元素都具有简单的数据类型，例如是整数类型，那么我们就得到一个简单的向量（一个一维向量）。但是我们将 看到，SNMP 在使用这个数据类型时，其向量中的每一个元素是一个 SEQUENCE（结构）。因而可以将它看成为一个二维数组或表。

SMIv1 和 SMIv2 具体的对比如下：

SMIv1	SMIv2	SMIv1	SMIv2	SMIv1	SMIv2
SYNTAX		ACESS	MAX-ACESS	STATUS	

SMIv1	SMIv2	SMIv1	SMIv2	SMIv1	SMIv2
● SimpleSyntax – INTEGER – OCTET STRING – OBJECT IDENTIFIER – NULL ● ApplicationSyntax – NetworkAddress – Counter (0..4294967295) – Gauge (0..4294967295) – TimeTicks (0..4294967295) – Opaque	● SimpleSyntax – INTEGER (-2147483648..2147483647) – OCTET STRING (SIZE (0..65535)), – OBJECT IDENTIFIER – NULL ● ApplicationSyntax – IpAddress – Counter32 (0..4294967295) – Gauge (0..4294967295) – TimeTicks (0..4294967295) – Opaque	● read-only ● read-write ● write-only ● not-accessible	● read-only ● read-write ● write-only ● not-accessible ● accessible-for-notify	● mandatory ● optional ● obsolete	● current ● deprecated ● obsolete

SMIv1	SMIv2	SMIv1	SMIv2	SMIv1	SMIv2
	(0..4294967295) - Counter64 (0..18446744073709551615) - Unsigned32 (0..4294967295)				

1.4.5 SNMPv1 工作原理

SNMP 规定了 5 种协议数据单元 PDU（也就是 SNMP 报文），用来在管理进程和代理之间的交换。

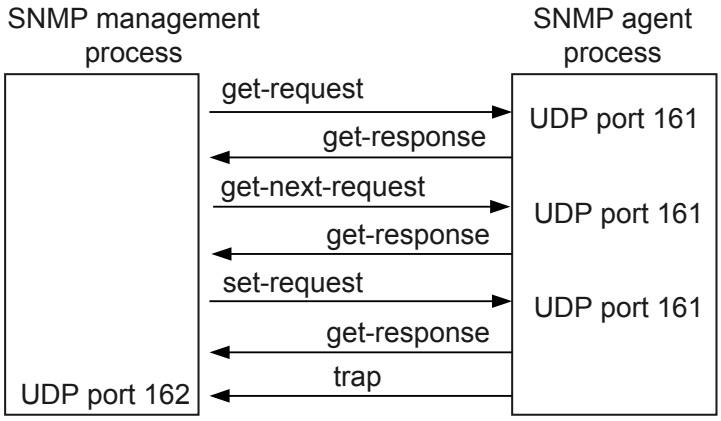
- get-request 操作：从代理进程中提取一个或多个参数值。
- get-next-request 操作：从代理进程中按照字典序提取下一个参数值。
- set-request 操作：设置代理进程的一个或多个参数值。
- get-response 操作：返回的一个或多个参数值。这个操作是由代理进程发出的，它是前面三种操作的响应操作。
- trap 操作：代理进程主动发出的报文，通知管理进程有某些事情发生。

前面的 3 种操作是由管理进程向代理进程发出的，后面的 2 个操作是代理进程发给管理进程的，为了简化起见，前面 3 个操作今后叫做 get、get-next 和 set 操作。如图 1-4 所示，描述了 SNMP 的这 5 种报文操作。

说明

在代理进程端是用知名端口 161 来接收 get 或 set 报文，而在管理进程端是用知名端口 162 来接收 trap 报文。

图 1-4 SNMP 的报文操作



一个 SNMP 报文共有三个部分组成，即公共 SNMP 首部、get/set 首部或 trap 首部、变量绑定。

公共 SNMP 首部

共三个字段：

- 版本

  版本字段的值是报文版本号减 1，如果是 SNMPv1 报文则对应字段值为 0。
- 共同体（community）

  共同体就是管理进程和代理进程之间的明文口令，字符串形式，常用的是 6 个字符“public”。
- PDU 类型

  共有 5 种 PDU 的类型，其对应关系如表 1-2 所示意图。

表 1-2 PDU 类型

PDU 类型	名称
0	get-request
1	get-next-request
2	get-response
3	set-request
4	trap

get/set 首部

- 请求标识符(request ID)

  这是由管理进程设置的一个整数值。代理进程在发送 get-response 报文时也要返回此请求标识符。管理进程可同时向多个代理发出 get 报文，这些报文都使用 UDP 进行传送，先发送的有可能后到达。设置了请求标识符可使管理进程能够识别返回的响应报文对应相应的请求报文。
- 差错状态（error status）

  由代理进程响应时填入，如表 1-3 的描述。

表 1-3 差错状态描述

PDU 类型	名称	说明
0	noError	一切正常
1	tooBig	代理无法将回答装入到一个 SNMP 报文之中
2	noSuchName	操作指明了一个不存在的变量

PDU 类型	名称	说明
3	badValue	一个 set 操作指明了一个无效值或无效语法
4	readOnly	管理进程试图修改一个只读变量
5	genErr	某些其他的差错

- 差错索引(error index)

  当出现 noSuchName、badValue 或 readOnly 的差错时，由代理进程在响应时设置的一个整数，它指明有差错的变量在变量列表中的偏移。

trap 首部

- 企业（enterprise）

  填入 trap 报文的网络设备的对象标识符。此对象标识符在对象命名树 enterprise 结点{1.3.6.1.4.1}的子树上。

- trap 类型

  此字段正式的名称是 generic-trap，分为表 1-4 中的 7 种。

表 1-4 trap 类型描述

Trap 类型	名称	说明
0	coldStart	代理进行了初始化
1	warmStart	代理进行了重新初始化
2	linkDown	一个接口从工作状态变为故障状态
3	linkUp	一个接口从故障状态变为工作状态
4	authenticationFailure	从 SNMP 管理进程接收到具有一个无效共同体报文
5	egpNeighborLoss	一个 EGP 相邻路由器变为故障状态
6	enterpriseSpecific	代理自定义的事件，需要用后面的“特定代码”来指明

当使用上述类型 2、3、5 时，在报文后面变量部分的第一个变量应标识响应的接口。

- 特定代码(specific-code)

  指明代理自定义的事件（若 trap 类型为 6），否则为 0。

- 时间戳(timestamp)

  指明自代理进程初始化到 trap 报告的事件发生所经历的时间，单位为 10ms。例如时间戳为 1908 表明在代理初始化后 19080ms 发生了该事件。

变量绑定(variable-bindings)

指明一个或多个变量的名及其对应的值。在 get 或 get-next 报文中，此值为空。

1.4.6 SNMPv2c 工作原理

SNMPv2 已被作为 Internet 的推荐标准予以公布。

简单性是 SNMP 标准取得成功的主要原因。因为在大型的、多厂商产品构成的复杂网络中，管理协议的明晰是至关重要的，但同时这又是 SNMP 的缺陷所在—为了使协议简单易行，SNMP 对部分的功能进行了裁减，如：

1. 没有提供成批存取机制，对大块数据进行存取效率很低；
2. 只在 TCP/IP 协议上运行，不支持别的网络协议；
3. 没有提供 manager 与 manager 之间通信的机制，只适合集中式管理，而不利于进行分布式管理；
4. 只适于监测网络设备，不适于监测网络本身。

针对这些问题，对 SNMP 改进工作一直在进行。如 1991 年 11 月，推出了 RMON（Remote Network Monitoring）MIB，加强 SNMP 对网络本身的管理能力。它使得 SNMP 不仅可管理网络设备，还能收集局域网和互联网上的数据流量等信息。1992 年 7 月，针对 SNMP 缺乏安全性的弱点，又公布了 SNMPv2。1996 年，IETF 的工作组发布了一系列 RFC 文档。在这系列新文档中，舍弃了原有 SNMPv2 的安全方面特性，最终形成 SNMPv2c。SNMPv2c 仍然使用 SNMPv1 的消息机制和共同体概念。

SNMPv2c 对 SNMPv1 的主要增强可以分为以下几类：

- SMI（管理信息结构）

  SNMPv2c 支持 SMIv2，对比只支持 SMIv1 的 SNMPv1，管理站与管理站之间通信能力得到加强。
- 报文类型

  对比 SNMPv1，SNMPv2c 新增两种报文类型。
  - get-bulk 报文：基于 get-next 实现。通过给 get-bulk 操作，实现了网管端对被管理设备的信息群查询，相当于连续执行多次 get-next 操作。在网管侧可以设置 get-bulk 的报文循环次数（相当于主机侧在一次报文交互时，执行 get-next 操作的次数）。
  - informRequset 报文：通过 informRequest 操作，为告警提供保证机制。设备端在发送 informRequest 消息后，网管端只有发送确认报文给设备端，设备端才能够确认网管端已经接收到告警信息，否则会重新传送此告警消息，直到得到网管端的确认消息或发送的次数到达最大重传次数。

1.4.7 SNMPv3 工作原理

RFC 2271 定义的 SNMPv3 体系结构，体现了模块化的设计思想，可以简单地实现协议功能的增加和修改。其特点：

- 适应性强：适用于多种操作环境，既可以管理最简单的网络，又能够满足复杂网络的管理需求。
- 扩充性好：可以根据需要增加模块。
- 安全性好：具有多种安全处理模块。

SNMPv3 主要有四个模块：信息处理和控制模块、本地处理模块、用户安全模块以及基于视图的访问控制模块。

与 SNMPv1 和 SNMPv2 相比，SNMPv3 通过本地处理模块和用户安全模块实现访问控制、身份验证和加密。

信息处理和控制模块

信息处理和控制模块（Message Processing And Control Model）在 RFC 2272 中定义，它负责 SNMP 报文的产生和分析，并判断信息在传输过程中是否要经过代理服务器等。在信息产生过程中，该模块接收来自调度器（Dispatcher）的 PDU，然后由用户安全模块在信息头中加入安全参数。在分析接收的信息时，先由用户安全模块处理信息头中的安全参数，然后再将解包后的 PDU 送给调度器处理。

本地处理模块

本地处理模块（Local Processing Model）的功能主要是进行访问控制、数据组包和中断。访问控制是指通过设置代理的有关信息使不同的管理站的管理进程在访问代理时具有不同的权限，通过 PDU 完成。常用的控制策略有两种：限定管理站可以向代理发出的命令或确定管理站可以访问代理的 MIB 中的具体内容。访问控制的策略必须预先设定。SNMPv3 通过使用带有不同参数的原语来灵活地确定访问控制方式。

用户安全模块

用户安全模块（User Security Model）则提供身份验证和数据加密服务。实现这个功能要求管理站和代理必须共享同一密钥。

- 身份验证：身份验证是指代理（管理站）接到信息时首先必须确认信息是否来自有权的管理站（代理）并且信息在传输过程中未被改变。RFC2104 中定义了 HMAC，这是一种使用安全哈希函数和密钥来产生信息验证码的有效工具，在互联网中得到了广泛的应用。SNMP 使用的 HMAC 可以分为两种：HMAC-MD5-96 和 HMAC-SHA-96。前者的哈希函数是 MD5，使用 128 位 authKey 作为输入。后者的哈希函数是 SHA-1，使用 160 位 authKey 作为输入。
- 加密：采用数据加密标准（DES）的密码组链接（CBC）码，使用 128 位的 privKey 作为输入。管理站使用密钥计算验证码，然后将其加入信息中，而代理则使用同一密钥从接收的信息中提取出验证码，从而得到信息。加密的过程与身份验证类似，也需要管理站和代理共享同一密钥来实现信息的加密和解密。

基于视图的访问控制模块

基于视图的访问控制模块（View-based Access Control Model）在 RFC 2515 中定义，它主要用于对用户组或者团体名实现基于视图的访问控制。用户必须首先配置一个视图，并指明权限。用户可以在配置用户或者用户组或者团体名的时候，加载这个视图达到限制读操作或写操作或 trap（v3）的目的。

1.4.8 SNMP 协议在安全性方面的比较

表 1-5 SNMP 协议安全性比较

协议版本	用户校验	加密	鉴权
v1	NO，采用团体字	NO	NO
v2c	NO，采用团体字	NO	NO

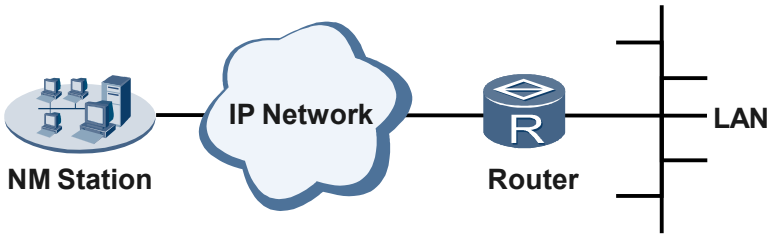
协议版本	用户校验	加密	鉴权
v3	YES，基于用户名的加解密	YES	YES

1.5 应用

网管软件通过 SNMP 协议对设备进行管理，查询设备的运行信息，接收设备发出的告警，采集性能统计数据。

SNMP 应用如图 1-5 所示。

图 1-5 网管通过 SNMP 协议管理设备示意图



1.6 术语与缩略语

术语

术语	解释
SNMP	简单网管协议
代理	被管设备端和管理相关的软件叫做代理进程（agent）。
ASN.1	抽象语法表示，使用独立于物理传输的方法定义协议标准中的数据类型。ASN.1 描述传输过程的中的语法，但不涉及具体数据含义的表示。
BER	基本编码规则，按照 ASN.1 的语法结构，描述了在传送过程中数据内容是如何表示的。
实体	可以被管理的软件或硬件。

缩略语

缩略语	英文全称	中文全称
SNMP	Simple Network Management Protocol	简单网管协议

缩略语	英文全称	中文全称
SMI	Structure of Management Information	管理信息结构
MIB	Management Information Base	管理信息库
PDU	Protocol Data Unit	协议数据单元
NMS	Network Management System	网络管理系统

2 RMON

关于本章

介绍了 RMON 的引入、基本概念、原理及华为的实现和应用。

2.1 介绍
介绍 RMON 的基本知识。

2.2 参考标准和协议
介绍了引用的相关资料。

2.3 可获得性

2.4 原理描述

2.5 应用

2.6 术语与缩略语

2.1 介绍

介绍 RMON 的基本知识。

RMON 的基本概念

- NMS

  网管站 NMS 是运行客户端程序的工作站。网络管理者通过向设备端发送请求，以及接收被管理设备发送的告警，来对设备进行监控和配置。

- MIB

  管理信息库，定义了被管对象的集合，把被管对象组织起来。

- Agent

  Agent 是驻留在被管理设备上的一个进程。

- 监视器

  用于监视网络通信情况的设备叫做监视器或称为网络探测器。

- 轮询

  网管站通过向设备发送 SNMP 报文，对设备的运行情况进行查询或对设置的参数进行配置。

SNMP 在网管方面的不足

SNMP 是互联网络中使用最广泛的网管协议，通过嵌入到设备中的代理软件来实现对网络通信信息的收集和统计。管理软件通过轮询方式向代理的 MIB 发出查询信号得到这些信息，通过得到的信息实现对网络的管理。虽然 MIB 计数器把统计数据的总和记录下来了，但它无法对日常的通信情况进行历史分析，为了能全面的查看一天中的流量和流量的变化情况。网管软件需要不断的轮询，才能通过得到的信息，分析出网络的状况。

采用 SNMP 进行轮询有两个明显的缺点：

- 占用了大量的网络资源，在大型的网络中，通过轮询的方式会产生大量的网络通信报文，这样将会导致网络的拥塞甚至于会引起网络的阻塞，故 SNMP 不适合管理大型的网络，不适合回收大量的数据，如路由表信息。
- 加重了管理者的负担，在 SNMP 轮询中收集数据的任务是由网络管理者通过网管软件完成的，如果网络管理者监控了 3 个以上的网段，就有可能出现由于负载过重，网络管理者无法完成任务的情况。

另外，MIB-II 中的公有 MIB 和各厂家定义的私有 MIB，主要提供有关设备的数据信息，如设备端口状态、流量信息等，没有提供一个子网网管的信息，这就不能满足管理大型互联网络的需要。

RMON 的引入

为了提高管理信息的可用性、减少管理站的负担、满足网络管理员监控多个网段性能的需求，IETF 开发了 RMON（Remote Network Monitoring）用以解决 SNMP 在日益扩大的分布式互联中的局限性。

RMON 基于 SNMP 体系结构实现，与现存 SNMP 框架兼容，仍然是网管工作站 NMS 和运行在各网络设备上的代理 Agent 两部分组成。没有另外一套底层机制，因此实现比

较简单。RMON 标准可以对数据网进行防范管理，它使 SNMP 更有效、更积极主动地监视远程网络设备，为有效的监控子网提供了一种高效的手段。减少了网管站与代理之间的通讯流量，从而实现更简单有效地管理大型网络的目的。

RMON 的目标

RMON 提供了一种高效的方法来监控子网范围内的行为，以下列出了 RMON 的实现目标：

- 离线操作：即使网络管理者不查询的情况，监视器也能不间断地采集错误、性能和配置信息。
- 前设监视：在网络出现故障时，监视器应该通知终端出现了此故障，并能为诊断故障提供有用的信息。
- 问题探测和报告：监视器可以预先监视包括网络中出现的问题和网络资源的消耗情况，用来检查错误和异常。
- 数据分析：监视器可以对子网采集到数据进行分析，从而减轻网管站的压力。
- 多管理站：网络管理站可能有多个。通过增加管理站来提高可靠性、执行不同的功能、为不同的内部单元提供不同的管理性能。

受益

企业网用户可以使用网管系统通过 RMON 框架更高效的采集 AR 设备的运行数据，从而更有效的监控设备的运行，降低操作维护费用。

2.2 参考标准和协议

介绍了引用的相关资料。

如果您想了解更多关于 RMON 的信息，请参考以下文档。

文档	描述	备注
RFC2021	Remote Network Monitoring Management Information Base Version 2	-
RFC2819	Remote Network Monitoring Management Information Base	-

2.3 可获得性

涉及网元

如果使用专用的 RMON Probe（探测仪）进行数据收集，则需要准备探测仪。

License 支持

不涉及。

版本支持

产品	最低支持版本
AR3200	V200R002C00

硬件要求

无

2.4 原理描述

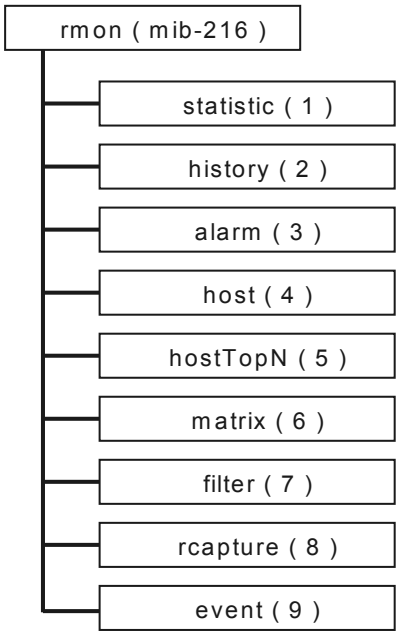
RMON 规范首先是一套 MIB 的定义，其作用是定义标准的网络监视功能和接口，使基于 SNMP 的管理终端和远程监控之间能够通信。

2.4.1 RMON 的基本框架

RMON 规范首先是一套 MIB 的定义，其作用是定义标准的网络监视功能和接口，使基于 SNMP 的管理终端和远程监控之间能够通信。MIB 被分成一定数量的功能表，每一组内部，可以有一个或多个控制表和一个或多个数据表。控制表对于管理者来说是可读可写的，而数据表是只读的。控制表定义了数据的采集功能，数据表按照控制表的定义，进行数据的采集。RMON MIB 结合在 MIB-II 中，子树的标识符为 16，按照功能分成九个组。

RMON 结构定义的组如图 2-1 所示。

图 2-1 RMON MIB 组



- 统计组：统计组统计被监控的每个子网的基本统计信息。它能统计某一网段的流量和各种类型包的分布，还能统计各种类型的错误帧数、碰撞次数等。
- 历史组：历史组定期收集网络状态统计信息并存储，以便后续的处理。
- 告警组：告警组允许针对告警变量（可以是本地 MIB 的任意对象）预先定义一组阈值，如果采样数据在相应的方向上越过阈值，监视器会记录日志或者把 Trap 发往网管站。
- 主机组：包含依附于该子网的各种类型主机的进出流量。
- 最高 N 台主机组：包含存储的主机统计信息，这些主机在主机表中某些参数最高。
- 矩阵组：以矩阵的形式表示错误和利用信息，以便于操作人员可以用任何一对地址来索引信息。
- 过滤组：过滤组提供一种手段，使得监视器可以观察接口上的分组，通过过滤选择出某种指定的特殊分组。
- 捕获组：捕获组可以用于设置一种缓冲机制，以捕获流经 filter 组的某个通道的数据分组。
- 事件组：事件组提供关于 RMON 代理所产生的所有事件的表。当某事件发生时，可以记录日志或发送 TRAP 到网管站。

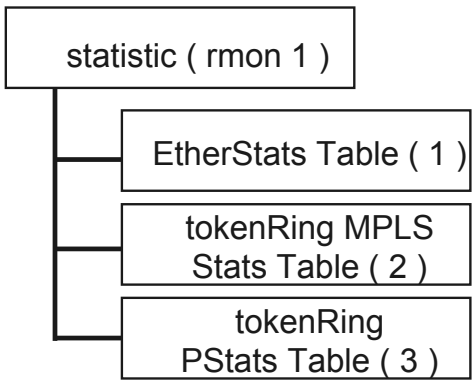
说明

告警组要求先实现事件组，最高主机组要求先实现 host 组，捕获组要求先实现过滤组。

统计组

统计组包含了每个被监控子网的统计信息，包含 3 个表如图 2-2 所示。

图 2-2 统计组



- EtherStatsTable 表

  该表包含了 21 个对象，每个被检测子网在该表中都有一条记录，表示每个被监控子网在该表中都有一条记录。该表中的大部分对象都是计数器，记录监视器从子网上收集到的各种不同的状态分组数。

  统计表提供了关于子网的相关信息，同时由于它还包括了很多错误信息，如 CRC 校验信息、正确包和错误包的信息，因此该组还能提供关于网络整体运行的情况。尽管统计组提供的信息和 MIBII 提供的信息有很多的相似的地方，但统计组提供的信息更加的详细，对于以太网网络更有针对性。

通过统计组中提供的信息，可以为告警组中的设置上限和下限阈值提供了依据，告警中可以通过设置统计组的信息，设置的相应的阈值，显示对网络的有效的监控。

- tokenRing MPLS Stats Table 与 tokenRing PStats Table

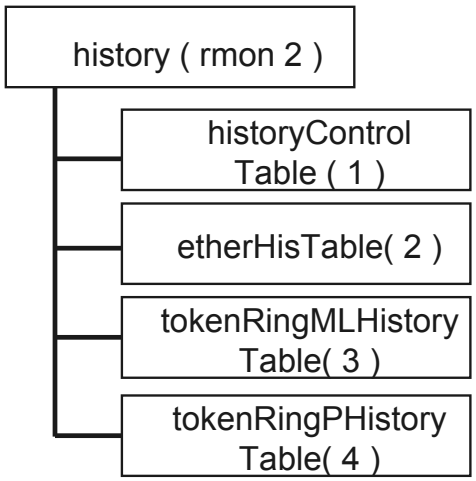
  这两个表中提供了令牌环网的统计信息，其中大部分是计数器。

历史组

历史组是用来定义监视器的一个或多个监控的取样功能的。

RFC 定义了两类表，一个历史控制表和 3 个历史数据表如图 2-3 所示。

图 2-3 历史组



- 控制表

  控制表用于指定端口采样以及采用功能的细节，如采样周期和接口信息。控制表中的每一条记录都定义了在一个特定端口上按照指定时间间隔采样数据的规范，它采集的数据都放入数据表中。RMON 规范建立在每一个被监控的接口上，至少要有两个控制行，一个定义 30s 的采样周期，另一个定义 30min 的采样周期。短周期用于检测突发的通信事件，长周期用于查看监控的稳定状态。

- 数据表

  数据表用于记录数据，其中 etherHistoryTable 是以太网介质特定的表。tokenRingMLIHistoryTable 和 tokenRing Table 是令牌介质的表。数据表提供了与统计组中类似的记数器。

告警组

告警组定义了一组网络性能的门限值。如果某个变量超过门限值时，将会产生一个告警事件并向网管发送 Trap 报文，该组必须和事件组同时实现，告警组依赖于事件组。

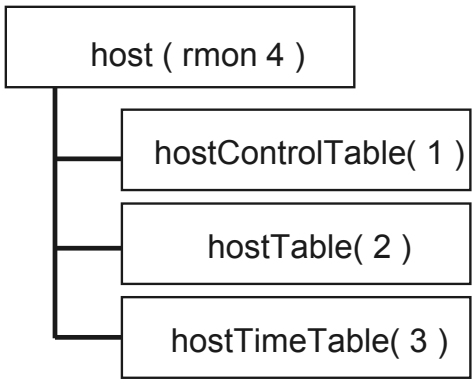
告警组只包含一个表 alarmTable。表中每一个记录都定义了一个特定的监视器变量、采样区间和门限值。

主机组

主机组用于收集 LAN 上特定主机的统计信息，通过监视器观察正常分组中源和目的 MAC 地址来发现 LAN 中新出现的主机。对于发现的每个主机，都维护着一组统计数据。

host 组由一个控制表和两个数据表组成如图 2-4 所示。

图 2-4 host 组



- 控制表 hostControl Table

  控制表中的每一行都对应于监视器中的唯一的网络接口。控制表中的选项可以定义数据，控制表中记录了数据表中最近被删除的表项的时间。控制表和数据表之间是直接对应的关系，对于控制表中每行所指定的网络接口，host Table 表都记录该网络接口上发现的 MAC 地址。

- 数据表 host Table

  host Table 的每行保存着相应主机的统计信息。该表既可以用主机的 MAC 地址也可以用网络接口进行索引。在 host Table 中增加一行后，监视器开始检查相应网络的接口的 MAC 地址。如该网络接口上每发现一台新的主机，就在 host Table 中加入一行。

- 数据表 host Time Table

  host Time Table 表的每一行都包含和 host Table 表的相应行一样的信息，只是该表是按照创建顺序而不是主机 MAC 地址来索引。

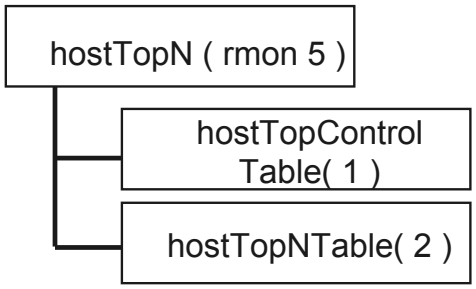
  hostTime Table 的组织还支持管理工作站在不需要下载整个表的情况下高效地找到某个特定网络接口的新表项。

最高 N 台主机组

最高 N 台主机组用于维护一个子网上一系列主机的统计信息，这些主机在基于某一参数的列表中处于最高位置，例如，该表可以表示某一天传输数据量最大的前 10 台主机信息。

最高 N 台主机组由一个控制表和一个数据表组成如图 2-5 所示。

图 2-5 最高 N 台主机组



- 控制表 hostTopControl Table

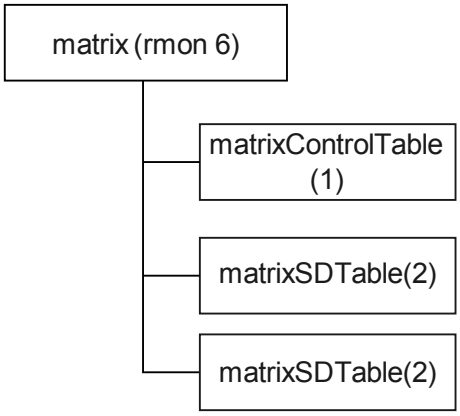
  在控制表 hostTopControl Table 中每一行定义了一个网络接口的一份 Top-N 报告。并定义了该 Top-N 报告上一次启动时候的距离系统启动时刻的时间。
- 数据表 hostTopTable

  数据表 hostTopTable 中包括了该表中表项所属的报告。每一行代表了唯一的一台主机，在表中有该主机对应的 MAC 地址，还定义了在采样周期内所选定变量的变化率。

矩阵组

矩阵组用于记录一个子网内各对主机间的流量信息，信息是以矩阵的形式存储。该组中共包含 3 张表，其中一张控制表和 2 张数据表，如图 2-6 所示。

图 2-6 矩阵组



- 控制表

  控制表中每一行标识一个单独的子网，并定义了一种用于发现特定网络接口上是否有会话，并且将其统计信息计入两张数据表中。
- 数据表 matrixSDTable

  此表用于存储从特定源主机到多台目的主机之间的流量统计信息。此表为最近在子网内交换过信息的每对主机保存两行：一行报告两主机间某一方向的流量；另一行报告另一方向的流量。

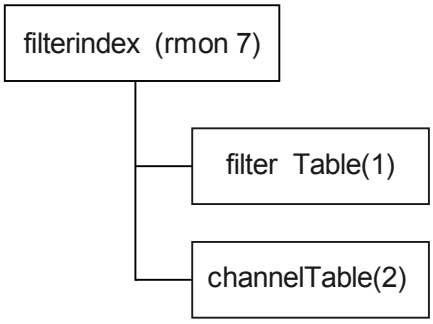
- 数据表 matrixSDTable

  该表与上表同，就是索引的顺序不同。

过滤组

过滤组提供了一种手段，使管理站可以指示监视器观测特定接口上选定的数据包。定义在该组中的基本组成块是两种类型的过滤器：数据过滤器和状态过滤器。数据过滤器允许监视器以位方式对观测的数据包进行屏蔽从而只有一部分匹配，状态过滤器允许监视器以数据包的状态为基础来进行匹配。各种过滤器可以用逻辑运算与、或来进行组合，形成复杂的测试模式。过滤组由两张控制表组成。如图 2-7 所示。

图 2-7 过滤组

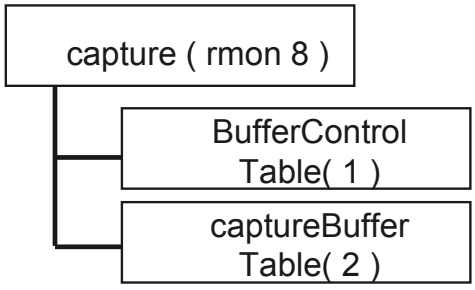


filterTable 定义了相关的过滤器，channelTable 的每一行定义了唯一的一个通道。channelTable 的每一行和 filterTable 的一行或几行相关联。

捕获组

捕获组可以用于设置一种缓冲机制，以捕获流经 filter 组的某个通道的数据分组。Capture 组的实现依赖于 filter 组的实现。它由两张表组成如图 2-8 所示。

图 2-8 捕获组



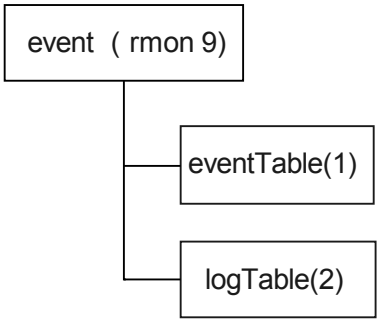
bufferControlTable 中的每一行定义了一个缓冲区，它用于捕获和存储一个通道中的分组。

数据表 captureBufferTable 的一行对应一个捕获的分组。

事件组

事件组支持事件的定义。一个事件可以被 MIB 中的某个条件所触发，而一个事件也可以触发 MIB 中定义的某个操作，并且事件还可以产生记入该组日志的信息或产生 SNMP trap 报文。event 组由一个控制表和一个数据表组成，如图 2-9 所示。

图 2-9 事件组



控制表 eventTable 包含事件的定义。表中的每行记录包含描述满足某种条件时产生的事件参数。

如果需要对一个事件进行记录，将在相应的 logTable 表中创建一些表项。

2.4.2 RMON 的特性

RMON 特性概述

RMON MIB 由一组统计数据、分析数据和诊断数据构成，利用许多供应商生产的标准工具都可以读取这些数据，因此它具有独立于网络站的远程网络分析功能。一般情况下每个子网都需要一个监视器。通常情况下把监视器称为探测器或 RMON 代理。这个监视器可以是一个独立的设备，该设备专门用来进行流量的获取和分析。为了进行有效的网络管理，监视器需要和中央网络管理站进行通信。

监视器观察网络中出现的每一个信息分组，并进行统计和总结，给网络管理人员提供重要的管理数据。监视器还能存储部分分组供以后分析使用，另外，监视器还能根据分组的类型进行过滤并捕获特殊的分组。

RMON 探测器和 RMON 客户端软件结合在一起在网络环境中实施 RMON。无需对设备进行不停的轮询 RMON 探测器就能自动地工作，利用 RMON 存储历史数据的能力，就能生成一个有关网络运行情况趋势的视图，无论何时出现意外的网络事件，探测器都能够上报网管中心，并描述不正常情况的捕获信息。客户端软件通过分析 RMON 上报的数据分析网络问题，并采取措施解决。

RMON 允许有多个监控者，它可用两种方法收集数据。

- 通过专用的 RMON 探测仪。NMS 直接从 RMON 探测器获取管理信息并控制网络资源，这种方式可以获取 RMON MIB 的全部信息。但使用此方式对网络进行监控，需要在所有的局域网段都部署 RMON 探测器，成本很高。
- 将 RMON Agent 直接嵌入网络设备（路由器、HUB）中，使它们成为带 RMON probe 功能的网络设备。NMS 是用 SNMP 基本命令与其交换数据信息，收集网络管理信息。这种方式受设备资源限制，一般无法获取 RMON MIB 的所有数据，大多

数只收集四个组（告警、事件、历史和统计）的信息。使用此种方式进行网络的监控 RMON 的效率更高、更经济。

说明

目前 AR3200 只能对网络设备的以太网接口进行监控和统计。

2.4.3 RMON 的远程监控

远程监控可以由专用的设备来实现，也可以由一个系统的功能模块来实现，为了有效地管理远程监视器，RMON MIB 支持管理终端的配置。

配置

远程监控需要为数据采集进行配置。配置指定要采集的数据类型和形式。RMOM MIB 通过下面的方式实现。MIB 被分成一些功能组，在每一个组内，可以有一个或多个控制表，并对应这一个或多个数据表。控制表对于管理者来说是可以读可写的，而数据表是只读的。终端管理站通过修改控制表的参数，实现对需要数据的采集。设置参数是通过控制表中增加一条新记录或更改一条旧记录来实现的。当根据控制表中的数据采集出数据后，这些数据将被存放在相应的数据表中。

监视器的功能定义和实施都是以表的形式实现的。操作的过程类似于数据库的操作，通过对控制表中的每个参数赋值，控制表中的每一条记录都定义了一项特定的数据采集功能，与该控制记录相对应的是若干条数据记录。每一条控制记录和数据记录通过指针绑定在一起。控制记录中包括一个索引值，可以通过索引值找多个数据表中对应的数据记录。而每一条数据记录也包含相似的索引，可以通过索引找到相应的控制表记录。

为了修改控制表中的参数，可以使用先删除后插入的方法，即首先删除与该索引相关的控制记录，在删除控制表中记录的时候，也要把数据表中的相关记录一起删除，然后管理站会生成一个控制记录，并插入到控制表。在控制参数和数据记录存在一一对应的关系时，控制表和数据表可以成为一个表。

监视器的操作控制

RMON 继承了 SNMP 的操作方式，同样是通过读取管理信息库中特定的值，然后再设置该对象的值来达到发布命令和控制的目的，一个对象可以用来表示一个命令。该对象如果被设定为一个特殊的值，那么则表明将采取一个特殊的操作。

多管理站管理

一个 RMON 代理可能会收到多个管理站的管理。只要允许对同一资源同时进行访问，就有可能出现冲突和不可预料的后果，对于共享 RMON 的代理过程来说，有可能存在下面的困难：

- 对资源的要求可能超出监视支持的能力范围。
- 某个管理站有可能长时间占用监视器资源，这样就阻碍了其他的管理站对监视的使用。
- 占用监视器资源的管理站有可能崩溃，则其占用的资源将无法进行释放。

为了解决上述问题，就需要一种回避和裁决的功能。这就是嵌入 RMON 的管理信息库控制表中一个相对简单的控制特性。在每一个控制表中引入一个列对象 Owner，该列对象可以表示表中特定的记录及相关功能的所属关系。所属关系可以解决多个管理站并发的访问的问题，其具体的用法如下：

- 管理站能识别自己所属的资源，也知道自己不再需要的资源。

- 网络管理员知道这些资源，并能成功地完成释放资源或其功能。
- 一个被授权的管理员，可以有权释放被其他管理站预定的资源。
- 如果管理工作站重新启动，能够识别过去预定的资源并发放那些不再使用的资源。

RMON 规范建议，所属标志不能作为口令或访问控制机制使用。在 SNMP 管理框架中唯一的访问控制机制是 SNMP 视图和团体名。如果一个可读写的 RMON 控制表出现在某些管理站的视图中，则这些管理站都可以读写访问。但是控制表行只能由其所有者改变或删除，其他的管理站只能进行读访问。

如果多个网管站访问同一个控制表，就可以通过共享来提高效率。当一个管理站希望使用一个监视器的某项特性功能时，它将首先扫描相关控制表，检查是否存在已经被其他的管理站定义的该功能或是相近功能。如果确实存在这样的功能，那么该管理站可以共享该功能，其方法就是读取和该控制表相关的数据表中的记录。然而，这些记录的原所有者有权在任意时刻修改或删除这些记录，那么有可能出现任何想要该功能的管理站发现它所期望的功能不知何时已经被修改或删除了。

通常情况下，一个监视器在初始化时，应该会配置一组默认的功能集。每个相关的所有者标签都被设为以 monitor 开头的字符串，与这些预定义的功能相关的资源属于监视器本身。某个管理站如果需要的话，能够以只读的方式来使用该功能，但不能更改也不能删除该功能，除非是该监视器的管理员，而且该管理员通常也是网络的管理员。

2.4.4 RMON 的表管理

RMON 规范中对表结构分为控制表和数据表两部分。控制表通过定义参数来定义数据表的结构，数据表用于存储数据。RMON 规范包括了一组规范化的操作，在不修改、不违反 SNMP 管理框架的基础上，提供了清晰而明确的增加和删除行的操作。

增加行

管理站通过 SNMP 的 SET 操作来对表增加一行并遵循以下的原则：

- 管理站用 set request 向设备请求增加一行，如果新行的索引值与表中其他行的索引值不冲突，则代理产生一个新行。
- 新行生成后，对于管理站没有设置新值的列对象，代理可以置为默认值或者让新行维持这种不完整的状态。
- 在管理站生成所要生成的新行前，处于非活跃状态的新行一直保存非活跃状态。
- 如果管理站要求生成的新行已经存在，则返回错误信息。

删除行

只有行的所有者通过发送 set request 操作，可以把行状态对象的值置为无效，这样就删除了一行。

修改行

首先置行状态对象的值为无效，然后用 set request 请求改变对象的值，这样就改变了行对象的值。

2.4.5 RMON 在华为设备的实现

介绍 RMON 在华为设备上的实现。

RMON 的实现

依靠 RMON 自主的管理一个网络的要求是：所有的网段都能得到有效的监控。在局域网中部署 RMON 的探测器的成本很高。并且一个网段监控会削弱 RMON 的作用，给网络也带来的很大的负担。

基于此，很多厂商将 RMON 嵌入到网络设备中，采用这种方式会更经济、有效。华为公司的路由器中嵌入了 RMON Agent 模块，与 AR3200 系统中的其它模块形成一个完整的系统。网管工作站 NMS 可以完全利用 SNMP NMS，网络管理人员无须进行额外的学习。

AR3200 RMON 支持 RFC2819 规定的统计、历史、告警和事件四个组以及华为规定的私有扩展告警组（PERFORMANCE-MIB）。下面对这几个组分别进行介绍。

- 统计组

  统计组统计被监控的每个子网的基本统计信息。它能统计某一网段的流量和各种类型包的分布，还能统计各种类型的错误帧数、碰撞次数等。

  统计组包含一个以太网统计表（ethernetStatsTable）。只有以太网和千兆以太网的主接口（非子接口）上才能创建统计表行。一个接口下只能创建一个统计表行。统计表最多行数为 100。

- 历史组

  历史组定期收集网络状态统计信息并存储，以便后续的处理。

  历史组包含两个表：

  - 历史控制表（historyControlTable）：主要用来设置控制信息如接口，采样周期等。一旦采样完成，数据就存储到一个介质相关的表项里（ethernetHistoryTable）。只有以太网和千兆以太网的主接口（非子接口）上才能创建历史控制表行。历史控制行最大数为 100。
  - 以太网历史表（ethernetHistoryTable）：为网络管理员提供有关网段流量、错误包、广播包、利用率以及碰撞次数等其他统计信息的历史数据。每个这样的表项定义了采样，并且和引起其采样的历史控制表相联系。在每一个采用周期到达后，才进行历史表的创建，并且历史控制表中的每一项控制信息在以太网历史表中最多可以有 10 条历史数据相对应，该数据可以设置，如果超过指定条数，将循环覆盖。RMON 接收到历史控制表行删除的请求后，由于以太网历史表中没有自己的状态，管理员不能删除某一特定行，相连的所有历史统计表被删除。

- 告警组

  告警组允许针对告警变量（可以是本地 MIB 的任意对象）预先定义一组阈值，如果采样数据在相应的方向上越过阈值，监视器会记录日志或者把告警发往网管站。告警组需要先实现事件组。按照 RFC2819，告警功能采用一种滞后机制限制告警事件的产生。应用这个机制后，当采样数据以某方向跨过阈值时，将会产生一个事件。直到相反方向的阈值被跨过以前，不会产生更多的事件。不便于网管站对设备的监控。

  AR3200 在实现时没有采取这种方式，因为滞后机制可能长时间不会产生告警。在 AR3200 中，只要采样值回到正常阈值后就可以重新告警。告警表的实现依赖于事件表的实现，告警表创建后，只有在事件表创建后才有效，并且在告警行各参数有效时，告警表会自动更新告警行的状态为有效，告警表最多行数为 60。

  告警组包括一个告警表（alarmTable）。

- 事件组

  事件组提供关于 RMON 代理所产生的所有事件的表。事件组控制从设备来的事件和提示。事件表的每项描述被触发事件的参数。每个事件行是被 MIB 里其他的相

连的条件激活的。当事件行创建有效时，如果与之联系的告警行和扩展告警行各参数有效，自动更新他们的状态为有效。当事件无效时，更新与之联系状态为有效的告警行和扩展告警行，使其状态为临界状态。事件表最多行数为 60。

当某事件发生时，可以记录日志或发送 TRAP 到网管站。

事件组包括事件表（eventTable）和日志表（logTable）。

- 扩展告警组

  扩展告警组在 RFC2819 基础上增加了用表达式设定告警对象和告警生存时间的功能。并可以限定告警的存在时间。它包括一个扩展告警表（prialarmTable）。当对应事件创建并有效时，如果扩展告警行各参数有效，自动更新扩展告警行的状态为有效，当事件无效时，更新状态为有效扩展告警行，使其状态为临界状态。扩展告警表最多行数为 50。

  在 AR3200 中，为节约系统资源，每个表行都有自己的生存时间，规定了当此行状态不是有效状态 VALID 时，此行可以存在的时间。对于一直处于非 VALID 状态的行，其生存时间将递减。当生存时间为 0 时，行被删除。

各表项的容量和最大生存时间如**表 2-1** 所示。

表 2-1 各表的生存时间

表名称	表项容量(Byte)	最大生存时间（s）
统计表	100	600
历史控制表	100	600
告警表	60	6000
事件表	60	600
事件日志表	600	-
扩展告警表	50	6000

当有接口板或接口卡被拔出时，接口对应的统计表和历史控制表状态会变成 INVALID 状态，此时，对应的统计表和历史控制表的生存时间被设置为 1200s。如果表行的生存时间到达 0，该表行将被删除。

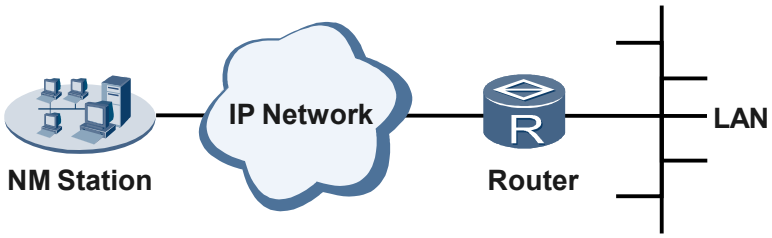
当接口插入时，如果存在对应的表行，则把对应表行的行状态变为 VALID。

2.5 应用

在实际网络的环境中，要对某一网段的状况进行监控、流量统计，可以通过配置 RMON 来实现。在首先配置的 SNMP 情况下，对网络进行监控，得到有关流量和各种类型包数量实际和历史统计信息。对此接口的流量字节数设置告警监控，当每分钟的流量超过设定值时，记录日志。并且怀疑此子网的广播和组播信息量超常，因此对该子网的组播和广播总数进行告警设置，当超过设定值后，主动向网管上报告警信息。

RMON 应用如**图 2-10** 所示。

图 2-10 通过 RMON 协议管理设备示意图



2.6 术语与缩略语

缩略语

缩略语	英文全称	中文全称
RMON	Remote Network Monitoring	远程监控

3 LLDP

关于本章

3.1 介绍

定义

链路层发现协议 LLDP（Link Layer Discovery Protocol）是 IEEE 802.1ab 中定义的第二层发现协议。第二层发现（Layer 2 Discovery）准确定位了诸如哪些设备附带有哪些接口，以及哪些设备与其他设备相互连接等二层信息，并显示出了客户端、交换机、路由器和应用服务器以及网络服务器之间的路径。

目的

通过 LLDP 获取的设备的二层信息对快速获取相连设备的拓扑状态、设备间的配置冲突、查询网络失败的根源将很有帮助。

受益

企业网用户可以通过使用网管系统，对支持运行 LLDP 协议的设备进行链路状态监控。

3.2 参考标准和协议

本特性的参考资料清单如下：

文档	描述	备注
IEEE 802.1ab	IEEE Standard for Local and metropolitan area networks : Station and Media Access Control Connectivity Discovery	-
IEEE 802.3at	Data Terminal Equipment(DTE) Power via the Media Dependent Interface(MDI) Enhancements	-

3.3 可获得性

涉及网元

需要网管配合。

License 支持

无需获得 License 许可，均可获得该特性的服务。

版本支持

产品	最低支持版本
AR3200	V200R001C00

3.4 原理描述

3.4.1 LLDP 的基本概念

MIB

MIB（Management Information Base）：管理信息数据库。MIB 数据库分为 LLDP Local System MIB 和 LLDP Remote System MIB。

LLDP Local System MIB：LLDP 本地系统管理信息数据库（以下简称 LLDP 本地 MIB），用来保存本地设备信息。包括设备 ID、接口 ID、系统名称、系统描述、接口描述、设备能力、网络管理地址。

LLDP Remote System MIB：LLDP 远端系统管理信息数据库（以下简称 LLDP 远端 MIB），用来保存相邻设备的信息。包括设备 ID、接口 ID、系统名称、系统描述、接口描述、设备能力、网络管理地址。

LLDP Agent

LLDP Agent：LLDP 代理，用于管理接口的 LLDP 操作。

LLDP Agent 要完成下列任务：

- 维护 LLDP 本地 MIB 的信息。
- 在本地状态发生变化的情况下，提取 LLDP 本地 MIB 信息并向邻居节点发送。在本地状态没有变化的情况下，按照一定的周期提取 LLDP 本地 MIB 信息向邻居节点发送。
- 识别并处理收到的 LLDP 报文。
- 维护 LLDP 远端 MIB 库。
- LLDP 本地 MIB 或 LLDP 远端 MIB 的状态发生变化的情况下，向网管发送 LLDP 告警。

LLDP 管理地址

LLDP 管理地址（以下简称管理地址）是供网管系统标识 AR3200 并进行管理的地址。管理地址可以明确地标识一台设备，有利于网络拓扑的绘制，便于网络管理。管理地址被封装在 LLDP 报文的 Management Address TLV 字段中传送给邻居节点。

LLDP 告警

LLDP 告警是在 LLDP 本地 MIB 库或远端 MIB 库发生变化的情况下，设备向网管系统发送的更新拓扑的提示信息。触发告警的原因为：

- 全局 LLDP 使能的状态变化。
- 本地管理地址改变。
- 邻居信息发生变化，包括添加/删除/老化邻居和由于邻居过多导致的 LLDP 报文丢弃。邻居管理地址变化，本地不会产生告警信息。

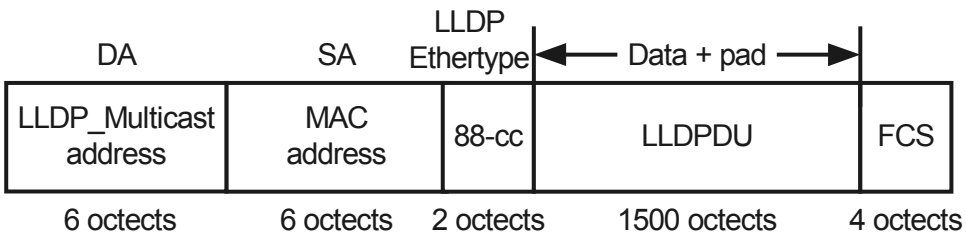
LLDP 告警功能对全局起作用，控制所有接口发送告警的能力。

3.4.2 LLDP 报文结构

LLDP 报文

封装了 LLDP 数据单元 LLDPDU（LLDP Data Unit）的以太网报文称为 LLDP 报文。LLDP 报文以组播方式传送，组播地址为 01-80-C2-00-00-0E，报文类型为 0x88CC。LLDP 报文结构如图 3-1 所示。

图 3-1 LLDP 报文结构



上图中各个缩写词的解释为：

- DA：目的 MAC 地址，LLDP 报文中该地址为组播 MAC 地址 01-80-C2-00-00-0E。
- SA：源 MAC 地址，即发送设备的 MAC 地址。
- LLDP Ethertype：报文类型，LLDP 报文中该类型为 0x88CC。
- LLDPDU：LLDP Data Unit，LLDP 数据单元，它是 LLDP 信息交换的主体。
- FCS：帧检验序列。

LLDPDU

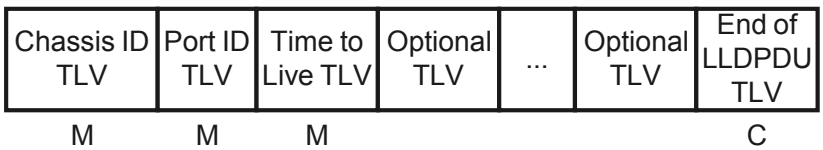
LLDPDU：LLDP 报文的数据单元，是设备将本地信息封装在以太网数据部分的数据单元。

组成 LLDPDU 之前，设备先将本地信息封装成 TLV（Type-Length-Value）格式，再由若干个 TLV 组合成一个 LLDPDU 封装在以太网报文的数据部分进行传送。

LLDPDU 往往根据要求包含了很多种不同的 TLV，根据这些不同的 TLV 来传输或者接收自己和邻近设备的信息。

LLDPDU 的结构如图 3-2 所示。

图 3-2 LLDPDU 结构图



LLDPDU 固定以 Chassis ID TLV、Port ID TLV 和 Time to Live TLV 开始，以 End of LLDPDU TLV 为结束，这四个 TLV 为必选的 TLV。其他均为可选 TLV，可以由设备自行定义是否包含在 LLDPDU 中。

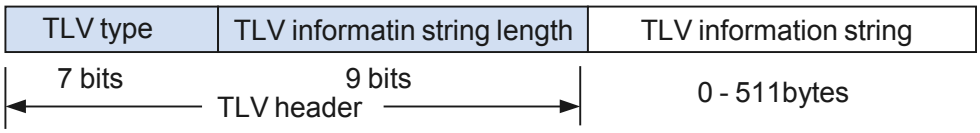
当端口的状态发生改变（去使能 LLDP、端口 shutdown）时，端口会向邻接设备发送一个 LLDP 报文，其中 Time To Live TLV 字段的 Value 值为 0，这个报文称为 shutdown 报文。shutdown 报文不包含任何可选 TLV。

TLV

TLV：组成 LLDPDU 的单元。表示一个对象的类型、长度和信息的单元。

TLV 的结构如图 3-3 所示。

图 3-3 TLV 结构图



- TLV Type 指的 TLV 类型，每个 TLV 的类型值不同，比如 End of LLDPDU TLV 的类型值为 0，Chassis ID TLV 的类型值为 1 等等。
- TLV information string length 是指此 TLV 的长度，它占 9 个 bit。
- TLV information string 其中第一个字节是指此 TLV 的子类型；剩余的字节为 TLV 真正的值。

每一个 TLV 代表一个信息。例如设备 ID、接口 ID、网络管理地址各自对应 Chassis ID TLV、Port ID TLV、Management Address TLV 等固定的 TLV。

3.4.3 LLDP TLV 信息类型

LLDP 可以封装的 TLV 包括基本 TLV、组织定义 TLV 以及媒体终端发现 MED（Media Endpoint Discovery）相关 TLV。基本 TLV 是被视为网络设备管理基础的一组 TLV，组织定义 TLV 和 MED 相关 TLV 是由标准组织以及其他机构定义的 TLV，用于增强对网络设备的管理，可根据实际需要配置是否在 LLDPDU 中发送。

基本 TLV

在基本 TLV 中，有几种类型的 TLV 对于实现 LLDP 功能来说是必选的，即必须在 LLDPDU 中发布，如表 3-1 所示。

表 3-1 基本 TLV 说明

TLV 类型	说明	是否必须发布
End of LLDPDU TLV	标志 LLDPDU 结束。	是
Chassis ID TLV	发送设备的桥 MAC 地址。	是

TLV 类型	说明	是否必须发布
Port ID TLV	用来标识 LLDPDU 发送端的端口。 ● 当设备不发送 MED TLV 时，内容为端口名称； ● 当设备发送 MED TLV 时，内容为端口的 MAC 地址，没有端口 MAC 时使用桥 MAC。	是
Time To Live TLV	本设备信息在邻居设备上的存活时间。	是
Port Description TLV	以太网端口的描述字符串。	否
System Name TLV	设备的名称。	否
System Description TLV	系统描述。	否
System Capabilities TLV	系统的主要功能以及有哪些主要功能被使能。	否
Management Address TLV	管理地址。	否

组织定义 TLV

1. IEEE 802.1 组织定义的 TLV

表 3-2 IEEE 802.1 组织定义的 TLV 说明

TLV 类型	说明
Port VLAN ID TLV	端口 VLAN ID。
Port And Protocol VLAN ID TLV	端口的协议 VLAN ID。
VLAN Name TLV	端口 VLAN 名称。
Protocol Identity TLV	端口支持的协议类型。

2. IEEE 802.3 组织定义的 TLV

表 3-3 IEEE 802.3 组织定义的 TLV 说明

TLV 类型	说明
Link Aggregation TLV	端口是否支持链路聚合以及是否已使能链路聚合。

TLV 类型	说明
MAC/PHY Configuration/Status TLV	端口的速率和双工状态、是否支持端口速率自动协商、是否已使能自动协商功能以及当前的速率和双工状态。
Maximum Frame Size TLV	端口支持的最大帧长度，取端口最大传输单元 MTU（Max Transmission Unit）。
Power Via MDI TLV	端口的供电能力，比如是否支持 PoE，是供电设备还是受电设备。

LLDP-MED 相关 TLV

LLDP-MED 相关 TLV 为 VoIP 提供了许多高级的应用，包括基本配置、网络策略配置、地址信息以及目录管理等，满足了语音设备的不同生产厂商在成本有效、易部署性、易管理性等方面的要求，并解决了在以太网中部署语音设备的问题，为语音设备的生产者、销售者以及使用者提供极大的便利性。

表 3-4 LLDP-MED 相关 TLV 说明

TLV 类型	说明
LLDP-MED Capabilities TLV	当前设备的设备类型以及在 LLDPDU 中可封装的 LLDP-MED TLV 类型。
Inventory TLV	设备的制造厂商。
Location Identification TLV	位置标识信息，供其它设备在基于位置的应用中使用。
Network Policy TLV	Voice VLAN 的 VLAN ID、二层优先级以及 DSCP 值等。
Extended Power-via-MDI TLV	当前设备的供电能力。
Hardware Revision TLV	媒体终端 ME（Media Endpoint）设备的硬件版本，只支持在本地设备查询，不支持做为 TLV 封装在报文中发送给邻居。
Firmware Revision TLV	ME 设备的固件版本，只支持在本地设备查询，不支持做为 TLV 封装在报文中发送给邻居。
Software Revision TLV	ME 设备的软件版本，只支持在本地设备查询，不支持做为 TLV 封装在报文中发送给邻居。
Serial Number TLV	ME 设备的序列号，只支持在本地设备查询，不支持做为 TLV 封装在报文中发送给邻居。

TLV 类型	说明
Model Name TLV	ME 设备的 Model Name，只支持在本地设备查询，不支持做为 TLV 封装在报文中发送给邻居。
Asset ID TLV	ME 设备的资产标识符，以便目录管理和断言跟踪，只支持在本地设备查询，不支持做为 TLV 封装在报文中发送给邻居。

3.4.4 LLDP 工作原理

LLDP 基本原理

链路层发现协议 LLDP（Link Layer Discovery Protocol）是 IEEE 802.1ab 中定义的第二层发现协议。

图 3-4 LLDP 结构框图

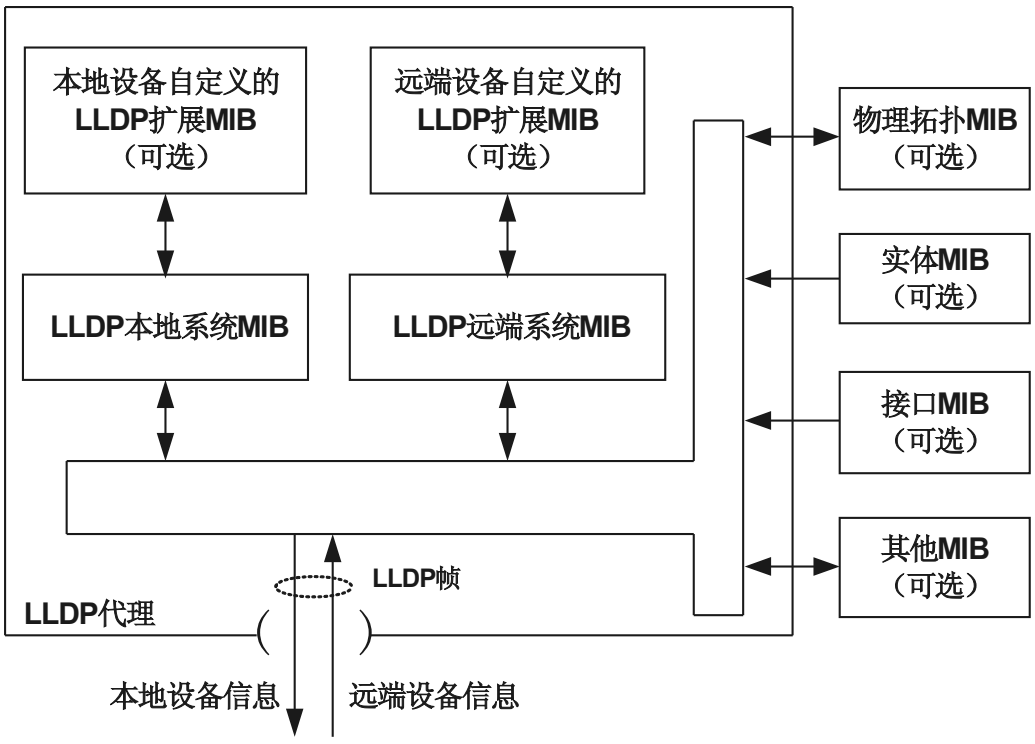


图 3-4 为 LLDP 结构框图。LLDP 与 MIB 密不可分，其基本实现原理为：

- LLDP 模块通过与设备上的 PTOPO MIB、Entity MIB、Interface MIB 以及 Other MIB 的交互，来更新自己的 LLDP local system MIB 库以及自己定义的 LLDP 扩展 MIB（图中的 Organizationally defined local device LLDP MIB extension）
- 然后通过 LLDP 帧，将自己的相关信息通过连接到远端设备的接口，发送给远端设备。

- 同时它接收远端设备发过来的 LLDP 帧，更新自己的 LLDP remote system MIB 即 LLDP 远端系统 MIB 库。

这样，通过 MIB 库，设备就很清楚的知道了自己相邻的设备的信息，包括连接的是远方设备的哪个接口，连接的远端设备的桥 MAC 地址等等。

LLDP 工作模式

LLDP 有以下两种工作模式。

- TxRx：既发送也接收 LLDPDU
- Disable：既不发送也不接收 LLDPDU

当端口的 LLDP 工作模式发生变化时，端口将对协议状态机进行初始化操作。为了避免端口工作模式频繁改变而导致端口不断执行初始化操作，可配置端口初始化延迟时间，当端口工作模式改变时延迟一段时间再执行初始化操作。

LLDPDU 发送机制

当发现一个新邻居（即接收到一个新的 LLDPDU 且本地没有保存发送该 LLDPDU 的设备的信息）、工作模式从 Disable 切换为 TxRx、设备名称变化、端口描述变化时，为了让其它设备尽快发现本设备，将启用快速发送机制，即将 LLDP 报文的发送周期缩短为 1 秒，并连续发送指定数量的 LLDPDU 后再恢复为正常的发送周期。

LLDPDU 接收机制

当端口工作在 TxRx 模式时，将对收到的 LLDPDU 及其携带的 TLV 进行有效性检查。通过有效性检查后，将邻居信息保存到本地设备，并根据 LLDPDU 携带的老化时间 TTL（Time To Live）值设置邻居信息在本地设备的老化时间，如果接收到的 LLDPDU 中的 TTL 值等于零，将立刻老化掉该邻居信息。

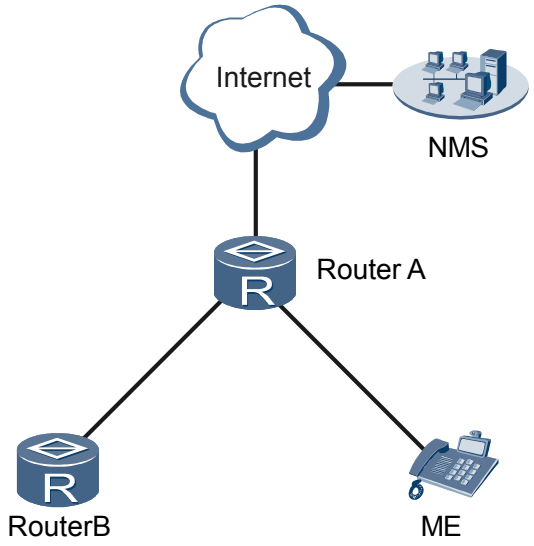
3.5 应用

AR3200 支持在以下三种组网方式下应用 LLDP。

单邻居组网方式

单邻居组网是指路由器设备的端口之间或者路由器与媒体终端 ME（Media Endpoint）的端口之间是直接相连，中间没有跨任何的设备，而且端口只有一个远端邻居设备的情况。单邻居组网如图 3-5 所示，RouterA 和 RouterB 之间以及 RouterA 和 ME 之间均是直接相连，RouterA 和 RouterB 的每一个端口都只有一个远端邻居。

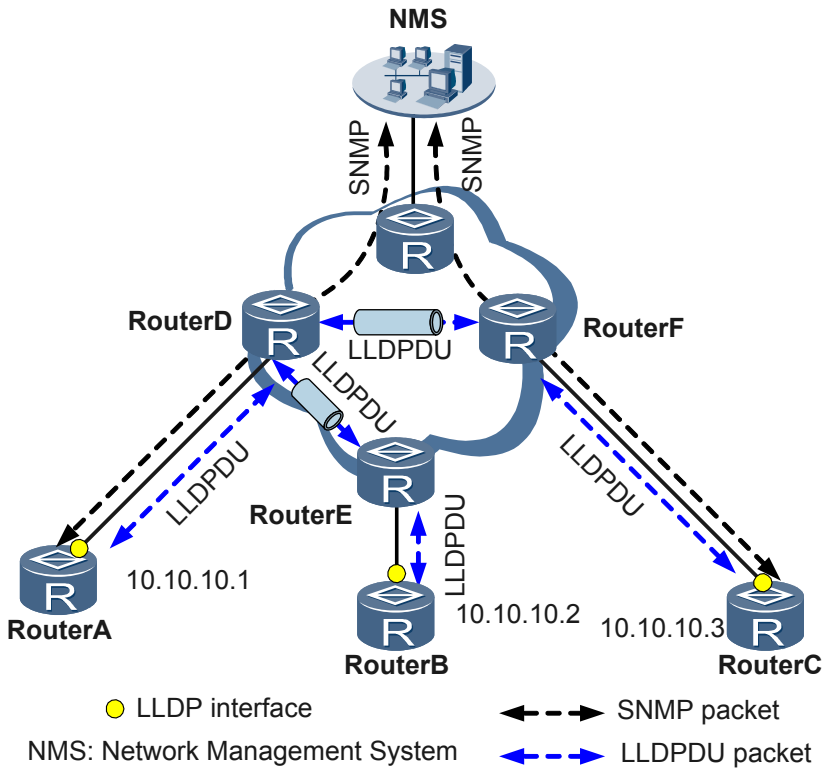
图 3-5 单邻居组网方式



多邻居组网方式

多邻居组网方式是指路由器设备的端口之间不是直接相连，中间跨越了未知网络，这时每个端口的远端邻居不止一个。多邻居组网如图 3-6 所示，RouterA、RouterB 和 RouterC 之间不是直接相连，中间跨越了未知网络，该网络中的设备未使能 LLDP 功能或者无需受 NMS 的管理，但该部分设备可以透传 LLDP 报文。这样 RouterA、RouterB 和 RouterC 的端口都不止有一个远端邻居。

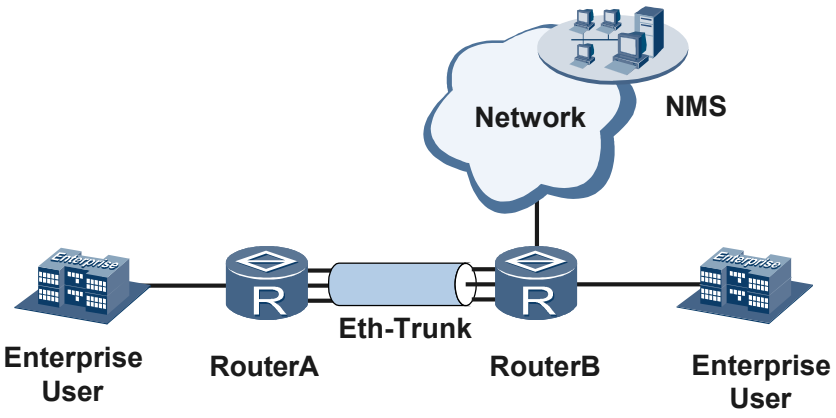
图 3-6 多邻居组网方式



存在链路聚合的组网方式

如图 3-7 所示路由器设备的端口之间存在链路聚合。该情况下，具体到链路聚合之间的每一个端口，都与单邻居组网模式的效果是一样的。

图 3-7 存在链路聚合的组网方式



3.6 术语与缩略语

术语

术语	解释
Agent	被管设备端和管理相关的软件叫做代理进程（agent）。

缩略语

缩略语	英文全称	中文全称
LLDP	Link Layer Discovery Protocol	链路层发现协议
MIB	Management Information Base	管理信息库
TLV	Type Length Value	类型、长度、值
LLDPDU	Link Layer Discovery Protocol Data Unit	LLDP 数据单元
ME	Media Endpoint	媒体终端

4 CWMP

关于本章

4.1 介绍

定义

CPE 广域网管理协议 CWMP（CPE WAN Management Protocol）是由数字用户线路 DSL（Digital Subscriber's Line）论坛发起开发的技术规范之一，编号为 TR-069，所以又被称为 TR-069 协议。它定义了用户侧设备 CPE（Customer Premises Equipment）和自动配置服务器 ACS（Auto-Configuration Server）之间的通信。

目的

目前，终端管理面临很多挑战：

- 终端管理，各自为政：

  各个厂商的网管通过 OMCI（Optical Network Termination Management and Control Interface）、内嵌运行通路 EOC（Embedded Operations Channel）协议管理各自的终端，多个厂商网管需要多次集成。
- 终端种类多，管理复杂：

  丰富的终端接入方式，催生多种新型终端设备，例如 AP（Access Point）、ONT（Optical Network Terminal）、SRG（Shared Risk Group），使管理难度增大。
- 海量终端，维护排障难度大：

  大量故障发生在用户侧，但终端数量繁多、部署分散，使维护排障的工作难度增大。

为了解决以上问题，CWMP 提出通过 ACS 对 CPE 进行远程集中管理，以减少设备的管理困难，节约维护成本和提高排障效率。

4.2 参考标准和协议

文档	描述	备注
CPE/ACS Application	The application uses the CPE WAN Management Protocol on the CPE and ACS, respecitively. The application is locally defined and not specified as part of the CPE WAN Management Protocol	-
RPC Methods	The specific RPC methods that are defined by the CPE WAN Management Protocol	-
SOAP	A standard XML-based syntax used here to encode remote procedure calls. Specifically Simple Object Access Protocol (SOAP) 1.1	不支持配置多个认证密码
HTTP	RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1	-

文档	描述	备注
SSL/TLS	The standard Internet transport layer security protocols. Specifically, either SSL 3.0 (Security Socket Layer), or TLS 1.0 (Transport Layer Security)	-
TCP/IP	Standard TCP/IP	-

4.3 可获得性

License 支持

CWMP 特性是 Huawei 的基本特性，无需获得 License 许可即可获得该特性的服务。

版本支持

产品	最低支持版本
AR3200	V200R001C00

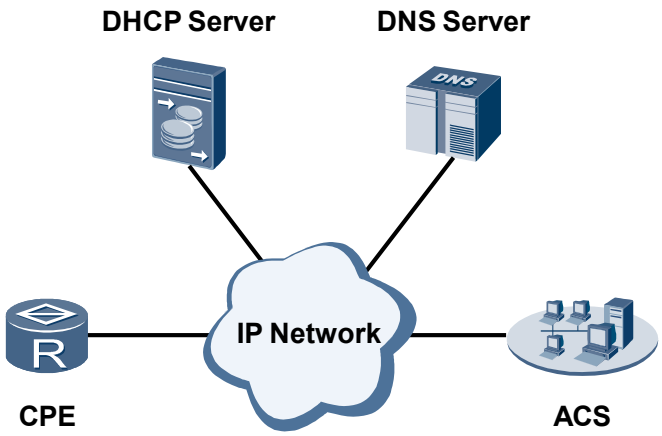
4.4 原理描述

CWMP 协议分为 ACS 侧和 CPE 侧：

- ACS 侧作为协议的服务器端，可以完成对 CPE 设备的配置和维护操作。
- CPE 作为协议的客户端被 ACS 服务器管理。

CWMP 的端到端框架模型如图 4-1 所示。

图 4-1 CWMP 端到端框架模型图



4.4.1 CWMP 基本功能

ACS 和 CPE 之间的会话建立

ACS 和 CPE 之间的会话建立有两种：

- CPE 发起连接
- ACS 操作触发

CPE 发起连接

CPE 通过发送 Inform 报文自动建立与 ACS 的连接，触发连接的方式有以下几种：

- CPE 启动，根据获取的 URL 值找到相应的 ACS，并自动发起连接。
- CPE 使能了周期性发送 Inform 报文功能，当周期（如 1 小时）到达时，CPE 会自动发送 Inform 报文来建立连接。
- CPE 使能了定时发送 Inform 报文功能，当时间点到达时，CPE 会自动发送 Inform 报文来建立连接。
- 如果当前连接异常中断，而且 CPE 自动重新连接的次数还没有达到上限，此时，CPE 也会自动发起连接。

ACS 操作触发

ACS 可以在任何时候自动向 CPE 发起连接请求（Connect Request），通过 CPE 的认证后，可以与 CPE 建立连接。

这种方式需要 ACS 和 CPE 之间通过 CPE 先发送连接请求，与 ACS 进行过至少一次通信。在这次通信中，如果 ACS 希望以后允许 ACS 先发起连接请求，它会将 CPE 的 IP 地址保存在地址列表中。

支持 ACS 对 CPE 的自动配置

当 CPE 上线时，ACS 可以自动下发一些配置给 CPE，完成对 CPE 的自动配置。设备支持的自动配置项主要包括：

- ACS 地址（URL）
- ACS 用户名（Username）
- ACS 密码（Password）
- Inform 报文自动发送使能标志（PeriodicInformEnable）
- Inform 报文周期发送时间间隔（PeriodicInformInterval）
- Inform 报文定期发送日期（PeriodicInformTime）
- CPE 用户名（ConnectionRequestUsername）
- CPE 密码（ConnectionRequestPassword）

支持对 CPE 系统启动文件和配置文件的上传/下载管理

为了实现对重要数据的备份，CPE 支持将当前的配置文件和日志文件上传到指定的服务器。

说明

当前设备支持上传的文件类型有：配置文件和日志文件。

网络管理员可以将系统启动文件、配置文件等重要文件保存在文件服务器上。当 ACS 发现某个文件的版本有更新，将会通知 CPE 进行下载。CPE 收到 ACS 的下载请求后，能够根据ACS 报文中提供的下载地址和文件名，自动到指定的文件服务器下载文件。下载完成后，对下载文件的合法性做相应的检查，并将下载结果（成功或失败）反馈给 ACS。

说明

当前设备不支持以数字签名的方式进行文件下载。

当前设备支持下载的文件类型有：系统启动文件、配置文件。

支持 ACS 对 CPE 的状态和性能监控

ACS 可以监控与其相连的 CPE 的各种参数。由于不同的 CPE 具有不同的性能，可执行的功能也有差异，因此 ACS 必须能识别不同类型 CPE 的性能，并监控到 CPE 的当前配置以及配置的变更。

CWMP 允许网络管理人员自定义监控参数并通过 ACS 获取这些参数，以便了解 CPE 的状态和统计信息。

说明

当前版本不支持状态和性能监控的数据模型（TR-143 定义的数据模型）。

支持 ACS 对 CPE 的故障诊断

ACS 服务器可以完成对 CPE 设备的基本故障诊断，设备支持的诊断方法有：Ping、Traceroute、ATM Loopback、DSL 线路诊断等。

说明

当前版本不支持故障诊断的数据模型（TR-098 定义的数据模型）。

4.4.2 CWMP 方法

ACS 对 CPE 的管理和监控是通过一系列的操作来实现的，这些操作在 CWMP 协议里称为 RPC 方法。主要方法的描述如下：

- Get：ACS 使用该方法可以获取 CPE 上参数的值。
- Set：ACS 使用该方法可以设置 CPE 上参数的值。
- Inform：CPE 与 ACS 建立连接时、底层配置发生改变时、或 CPE 周期性发送本地信息到 ACS 时，CPE 都要通过该方法向 ACS 发起通告信息。
- Download：为了保证 CPE 端硬件的升级以及厂商配置文件的自动下载，ACS 使用该方法可以要求 CPE 到指定的 URL 地址下载指定的文件来更新 CPE 的本地文件。
- Upload：为了方便 ACS 对 CPE 端的管理，ACS 使用该方法可以要求 CPE 将指定的文件上传到 ACS 指定的位置。
- Reboot：当 CPE 故障或者需要软件升级的时候，ACS 使用该方法可以对 CPE 进行远程重启。

支持的标准 RPC 方法有：

- Generic Methods：

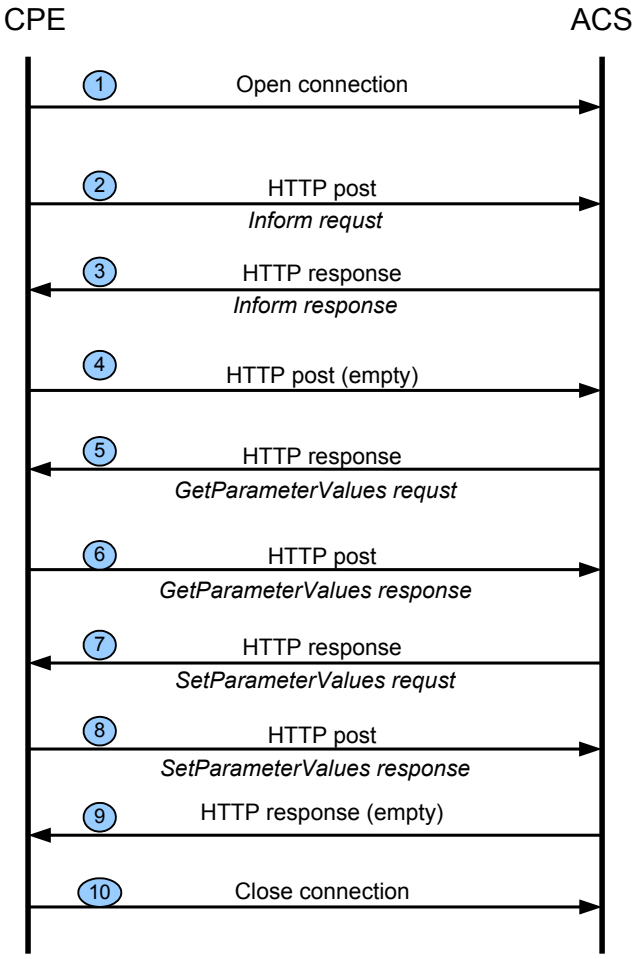
  GetRPCMethods
- CPE Methods：

  - SetParameterValues
  - GetParameterValues
  - GetParameterNames
  - SetParameterAttributes
  - GetParameterAttributes
  - AddObject
  - DeleteObject
  - Download
  - Upload
  - Reboot
- ACS Methods：
  - Inform
  - TransferComplete

4.4.3 CWMP 实现机制

以图 4-2 为例，介绍 CWMP 的消息交互过程。

图 4-2 CWMP 消息交互示例



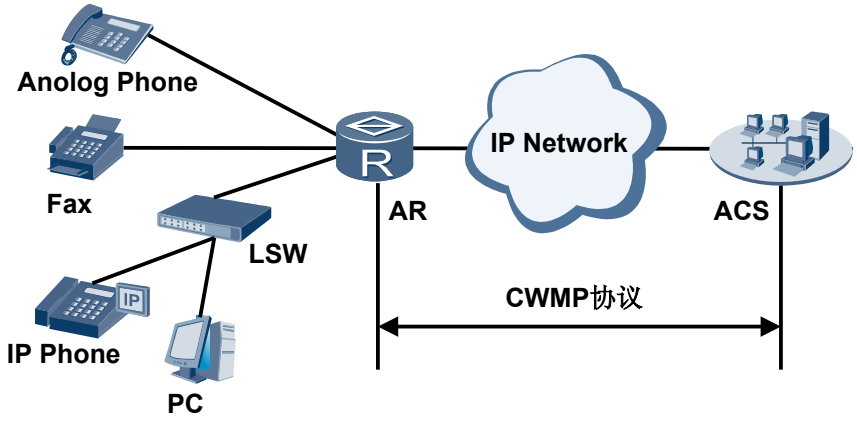
CWMP 的消息交互过程如下：

1. 会话连接初始化。CPE 是会话的发起者。如果 ACS 需要主动发起会话的建立，ACS 先给 CPE 发起 Connection Request（这时 CPE 是作为 HTTP SERVER）来触发 CPE 发起会话建立。
2. CPE 首先调用 ACS 的 Inform RPC 方法，通过向 ACS 发送 Inform 请求，上报设备信息，请求建立 CWMP 连接。
3. 若 CPE 通过认证，ACS 回应 InformResponse，Inform RPC 方法处理结束，CWMP 连接建立成功。
4. CPE 发送 HTTP post (empty)表明 CPE 没有再调用 ACS 支持的 RPC 方法。
5. ACS 通过向 CPE 调用 GetParameterValues RPC 方法，查询 CPE 的相关参数。
6. CPE 发送 GetParameterValuesResponse，告知 ACS 要查询的信息，GetParameterValues RPC 方法调用结束。
7. ACS 通过向 CPE 调用 SetParameterValues RPC 方法，对 CPE 进行相关配置。
8. CPE 发送 SetParameterValuesResponse，告知参数配置情况，SetParameterValues RPC 方法调用结束。
9. ACS 发送 HTTP response (empty)表明 ACS 没有再调用 CPE 支持的 RPC 方法。
10. CPE 拆除连接，完成本次会话。

4.5 应用

如图 4-3 所示，AR3200 作为 CPE 设备，连接电话、传真和交换机。ACS 通过 CWMP 协议完成对 AR3200 的控制和管理。ACS 与 AR3200 设备建立连接后，ACS 支持对设备系统启动文件和配置文件的管理，支持对设备进行配置，支持对设备状态和性能进行监控并进行故障诊断。

图 4-3 ACS 通过 CWMP 管理 AR3200



4.6 术语与缩略语

术语

术语	解释
会话	会话是 ACS 和 CPE 之间的一连串报文的交互过程。
事件	CWMP 协议中的事件特指 CPE 发生的需要通知 ACS 的某种状态或者参数改变。

缩略语

缩略语	英文全称	中文全称
ACS	Auto-Configuration Server	自动配置服务器
CPE	Customer Premises Equipment	用户驻地设备
CWMP	CPE WAN Management Protocol	CPE 广域网管理协议
RPC	Remote Procedure Call	远端过程调用

5 NTP

关于本章

5.1 介绍

定义

NTP（Network Time Protocol，网络时间协议）是用于互联网中时钟同步的应用层协议，其用途是在分布式时间服务器和客户端之间进行时钟同步，把主机的时钟同步到某些时间标准。

目的

NTP 的目的是对网络内所有具有时钟的设备进行时间同步，使网络内所有设备的时间基本保持一致，从而使设备能够提供基于统一时间的多种应用。

受益

企业网用户通过 NTP 协议可以对网络中的设备的时间进行统一的管理。设备可以自动同步系统时间，确保网络中的各个设备的时钟保持一致。

5.2 参考标准和协议

本特性的参考资料清单如下：

文档	描述	备注
RFC 1305	Basis of the NTP module requirements specification	-
RFC 5905	NTP version 4: Protocol and algorithm specification	-
RFC 5906	NTP version 4: Autokey specification	-

5.3 可获得性

涉及网元

无需其它网元的配合。

License 支持

无需获得 License 许可，均可获得该特性的服务。

版本支持

产品	最低支持版本
AR3200	V200R001C00

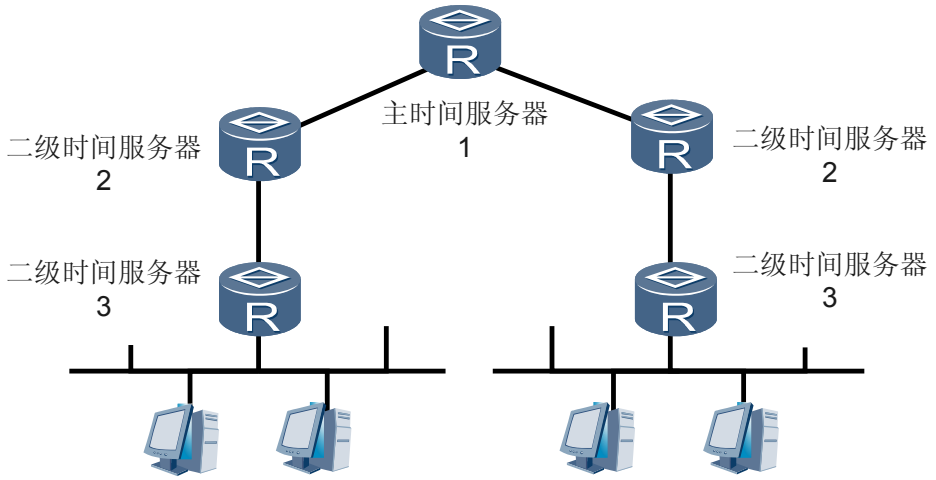
5.4 原理描述

NTP 主要用于在分布式时间服务器和客户端之间进行时间同步，把主机的时间同步到某些时间标准。服务器和客户端的概念是相对而言的，提供时间标准的设备称为时间服务器，接收时间服务的设备称为客户端。对于运行 NTP 的本地系统，既可以接收来自其他时钟源的同步，也可以作为时钟源去同步别的时钟，并且可以通过交换 NTP 报文互相同步。NTP 基于 UDP 传输，使用端口号 123。

5.4.1 网络结构

具体的组网方式是由主时间服务器、二级时间服务器、客户端和它们之间互连的传输路径组成，如图 5-1 所示。

图 5-1 NTP 网络结构



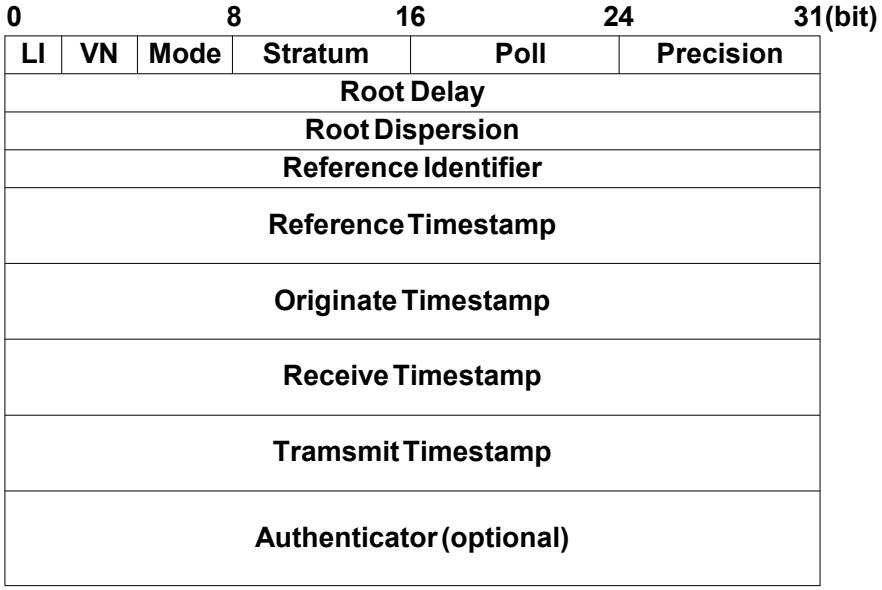
- 主时间服务器直接同步到主参考时钟，主参考时钟通常是 Radio Clock 或卫星定位系统等。
- 二级时间服务器通过网络中的主时间服务器或者其它二级服务器取得同步。二级时间服务器通过 NTP 将时间信息传送到网络内部的其它主机。

在正常情况下，同步子网中的主服务器和次级服务器呈现出一种分层主从结构，在这种分层结构中，主服务器位于根部，次级服务器向叶子节点靠近，层数递增，准确性递减。

5.4.2 NTP 报文结构

NTP 是基于用户数据报协议（UDP）的时间协议，NTP 使用的 UDP 端口是 123。NTP 报文结构如图 5-2 所示。

图 5-2 NTP 报文结构



上图中各字段的解释如表 5-1 所示。

表 5-1 NTP 报文字段说明

名称	长度	含义
LI（Leap Indicator）	2 比特	NTP 跳变指示，不同值表示的含义如下： ● 0：no warning，没有跳变 ● 1：last minute has 61 seconds，上一分钟含有 61 秒 ● 2：last minute has 59 seconds，上一分钟含有 59 秒 ● 3：alarm condition(clock not synchronized)，告警状态，时钟失步中
VN（Version Number）	3 比特	NTP 的版本号，取值为 1，2，3。
Mode	3 比特	NTP 的工作模式。不同值表示的含义如下： ● 0：reserved，保留 ● 1：symmetric active，主动对等体模式 ● 2：symmetric passive，被动对等体模式 ● 3：client，客户端模式 ● 4：server，服务器模式 ● 5：broadcast，广播模式 ● 6：reserved for NTP control message，NTP 控制报文 ● 7：reserved for NTP private use，内部使用预留

名称	长度	含义
Stratum	8 比特	时钟的层数，取值范围为 1 ～ 15，值为 1 的时钟准确度最高，从 1 到 15 依次递减。
Poll	8 比特	轮询间隔，取值范围为 6 ～ 10，值为 6 代表 2^6 秒，10 则代表 2^{10} 秒，依次类推。
Precision	8 比特	时钟的精度，有符号数，值为-128 代表 2^{-128} 秒，10 则代表 2^{10} 秒，依次类推。
Root Delay	32 比特	本地时钟到主参考钟的总往返延迟时间，有符号小数，小数点位于四字节的正中间。
Root Dispersion	32 比特	本地时钟到主参考钟的最大误差，有符号小数，小数点位于四字节的正中间。只有正数才有效。
Reference Clock Identifier	32 比特	参考时钟指示，时钟层数为 0 和 1 时是一个 4 字节左对齐零填充的 ASCⅡ字符串，用于标识参考时钟类型，时钟层数大于等于 2 时是参考时钟的地址。
Reference Timestamp	64 比特	本地时钟最后一次被设定或更新的时间，无符号小数，小数点位于 8 字节的正中间，如果值为 0 表示本地时钟从未被同步过。
Originate Timestamp	64 比特	NTP 报文离开源端时的本地时间，无符号小数，小数点位于 8 字节的正中间。
Receive Timestamp	64 比特	NTP 报文到达目的端的本地时间，无符号小数，小数点位于 8 字节的正中间。
Transmit Timestamp	64 比特	应答报文离开服务器端的本地时间，无符号小数，小数点位于 8 字节的正中间。
Authenticator	96 比特	（可选）验证信息，前面 4 字节是密钥，后面 8 字节是校验和。

5.4.3 工作模式

在实际使用中，为了满足不同情况下的网络时钟同步需求，根据网络部署情况选择适当的工作模式，NTP 的工作模式共有 4 种方式：客户/服务器模式、对等体模式、广播模式和组播模式。

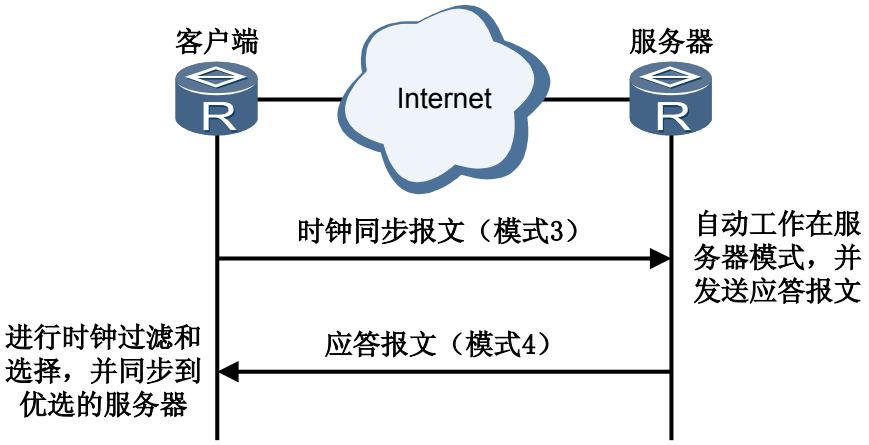
客户/服务器模式

- 客户模式：运行在客户模式的主机定期向服务器端发送报文，报文中的 Mode 字段设置为 3（客户模式）。不管服务器端是否可达及服务器端的层数。运行在这种模式的主机，通常是网络内部的工作站，它可以依照对方的时钟进行同步，但不会修改对方的时钟。
- 服务器模式：运行在服务器模式的主机接收并回应报文，报文中的 Mode 字段设置为 4（服务器模式）。运行在服务器模式的主机，通常是网络内部的时间服务器，它可以向客户端提供同步信息，但不会修改自己的时钟。

运行在客户模式的主机在重新启动时和重新启动后定期向运行在服务器模式的主机发送 NTP 报文。服务器收到客户端的报文后，首先将报文的目的 IP 地址和目的端口号分别

与其源 IP 地址和源端口号相交换，再填写所需的信息，然后把报文发送给客户端。服务器在客户端发送请求之间无需保留任何状态信息，客户端根据本地情况自由管理发送报文的时间间隔。

图 5-3 客户/服务器模式



对等体模式

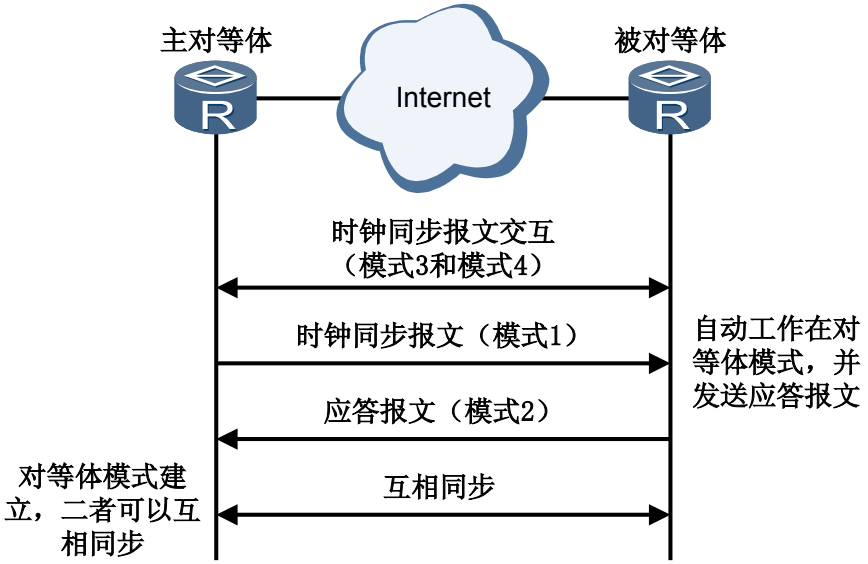
对等体模式下，主动对等体和被动对等体可以互相同步，等级低（层数大）的对等体向等级高（层数小）的对等体同步。主动对等体和被动对等体之间首先交互 Mode 字段为 3（客户端模式）和 4（服务器模式）的 NTP 报文。

- 主动对等体：运行在这一模式下的主机定期发送报文，报文中的 Mode 字段设置为 1（主动对等体）。不考虑它的对等体是否可达以及对等体的层数。运行在这一模式下的主机可以向对方提供同步信息，也可以依照对方的时间信息同步本地时钟。
- 被动对等体：运行在这一模式的主机接收并回应报文，报文中的 Mode 字段设置为 2（被动对等体）。运行在被动对等体模式的主机可以向对方提供同步信息，也可以依照对方的时间信息同步本地时钟。
- 运行被动对等体模式的必备条件：本机接收的报文来自一个运行在主动对等体模式下的对等体，且该对等体的层数等于或低于本机并路由可达。

说明

被动对等体模式运行在同步子网中层次较低层上时。这种模式下，不需要预先知道对等体的特性，因为只有当本机收到 NTP 报文时才建立连接及相关的状态变量。

图 5-4 对等体模式

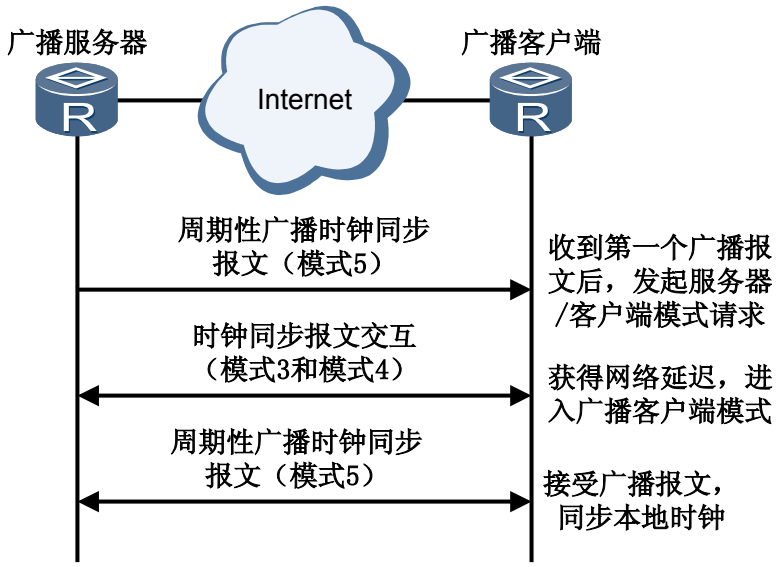


广播模式

- 运行在广播模式下，周期性向广播地址 255.255.255.255 发送时钟同步报文，报文中的 Mode 字段设置为 5（广播模式）。不管它的对等体是否可达或层数为多少。运行在广播模式的主机通常是网络内运行高速广播介质的时间服务器，向所有对等体提供同步信息，但不会修改自己的时钟。
- 客户端侦听来自服务器的广播消息包。当接收到第一个广播消息包后，客户端与服务器交互 Mode 字段为 3（客户模式）和 4（服务器模式）的 NTP 报文，即客户端先启用一个短暂的服务器/客户端模式与远程服务器交换消息，以获得客户端与服务器间的网络延迟。之后恢复广播模式，继续侦听广播消息包的到来，根据到来的广播消息包对本地时钟再次进行同步。

广播模式应用在有多台工作站、不需要很高的准确度的高速网络。典型的情况是网络中的一台或多台时间服务器定期向工作站发送广播报文，广播报文在毫秒级的延迟基础上确定时间。

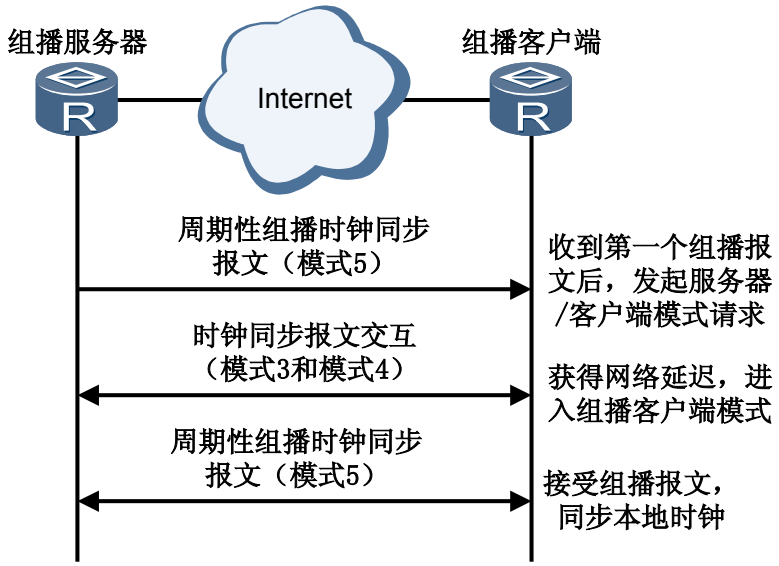
图 5-5 广播模式



组播模式

- 服务器端周期性向组播地址发送时钟同步报文，报文中的 Mode 字段设置为 5（组播模式）。运行在组播模式的主机通常是网络内运行高速广播介质的时间服务器，向所有对等体提供同步信息，但不会修改自己的时钟。
- 客户端侦听来自服务器的组播消息包。当接收到第一个组播消息包后，当客户端接收到第一个组播报文后，客户端与服务器交互 Mode 字段为 3（客户模式）和 4（服务器模式）的 NTP 报文，即客户端先启用一个短暂的服务器/客户端模式与远程服务器交换消息，以获得客户端与服务器间的网络延迟。之后，客户端恢复组播模式，继续侦听组播消息包的到来，根据到来的组播消息包对本地时钟进行同步。

图 5-6 组播模式

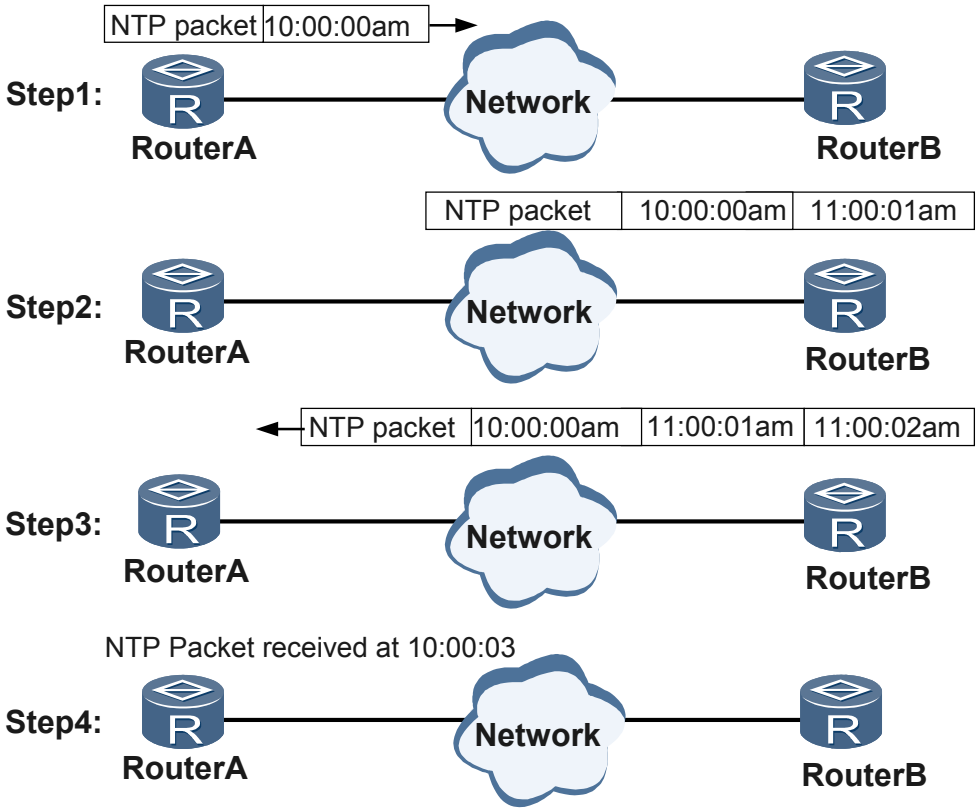


5.4.4 工作原理

NTP 的实现过程如图 5-7 所示：

RouterA 和 RouterB 通过广域网相连，它们都有自己独立的系统时钟，通过 NTP 实现系统时钟自动同步。

图 5-7 NTP 实现过程图



作如下假设：

- 在 RouterA 和 RouterB 的系统时钟同步之前，RouterA 的时钟设定为 10:00:00am，RouterB 的时钟设定为 11:00:00am。
- RouterB 作为 NTP 时间服务器，RouterA 的时钟与 RouterB 的时钟同步。
- 数据包在 RouterA 和 RouterB 之间单向传输需要 1 秒。
- RouterA 和 RouterB 处理 NTP 数据包的时间都是 1 秒。

系统时钟同步的工作过程如下：

- RouterA 发送一个 NTP 报文给 RouterB，该报文中带有它离开 RouterA 时的时间戳 10:00:00am（T1）。
- 此 NTP 报文到达 RouterB 时，RouterB 加上到达时间戳 11:00:01am（T2）。
- 此 NTP 报文离开 RouterB 时，RouterB 再加上离开时间戳 11:00:02am（T3）。
- RouterA 接收到该响应报文时，加上新的时间戳 10:00:03am（T4）。

至此，RouterA 拥有足够信息来计算以下两个重要参数：

- NTP 消息来回一个周期的时延：Delay=（T4-T1）-（T3-T2）。
- RouterA 相对 RouterB 的时间差：Offset=（（T2-T1）+（T3-T4））/2。

RouterA 根据这些信息来设定自己的时钟，实现与 RouterB 的时钟同步。

说明

以上是对 NTP 工作原理的简要描述，NTP 使用标准的 RFC1305 中的算法来确保时钟同步的精确性。

5.4.5 安全机制

当同步子网中的一台时间服务器发生意外或恶意的数据篡改或破坏时，通常不应该导致子网中其它时间服务器的计时错误。因此，NTP 还提供了两种安全机制：访问权限和 NTP 验证功能。这样就对网络的安全性提供了保障。

访问权限

通过设置访问权限保护本地 NTP 服务，设备提供的一种比较简单的安全措施。

支持 4 个等级的访问限制，当 1 个 NTP 访问请求到达本地时，按照最小访问限制到最大访问限制依次匹配，以第 1 个匹配的为准，匹配顺序如下：

- peer：（最小访问限制）可以对本地 NTP 服务进行时间请求和控制查询，本地时钟也可以同步到远程服务器。
- server：可以对本地 NTP 服务进行时间请求和控制查询，但本地时钟不会同步到远程服务器。
- synchronization：只允许对本地 NTP 服务进行时间请求。
- query：（最大访问限制）只允许对本地 NTP 服务进行控制查询。

验证功能

在安全性要求较高的网络中，可以启用 NTP 验证功能。配置 NTP 验证功能分为两部分：配置客户端、配置服务器端。NTP 采用 MD5 和自动密钥进行验证。

在配置 NTP 验证功能时，应注意以下原则：

- 客户端和服务器端均需要完整配置，NTP 验证才能生效。如果使能了 NTP 的 MD5 验证功能，应同时配置密钥，并声明可信的密钥。如果使能了 NTP 的自动密钥认证功能，应在服务器和客户端都安装自动密钥证书。
- 服务器端和客户端应配置相同的密钥。

5.4.6 NTP 的动态连接和静态连接

为了管理到各个同步源之间的同步信息，在 NTP 模块实现中，对每一个同步源都建立了一个 PEER 结构，并把这些 PEER 结构以 Hash 的形式存储成链状。每一个 PEER 结构对应于一个连接，即 Association 或者称为 session（会话）。NTP 协议规定连接的总数是 128，包括静态连接和动态连接，其中动态连接数最大为 100。

静态连接

静态连接就是指连接是通过命令行配置而创建的连接。

动态连接

动态连接就是指连接不是通过命令行配置而创建的，而是在 NTP 运行过程中动态创建的连接。

不同模式下的动态和静态连接

- 在客户/服务器模式中，用户需要在客户端配置要同步的服务器的地址。这个时候在客户端会建立一个静态连接，而服务器端只是被动的响应客户端的请求报文，不会建立连接。
- 在对等体模式中，用户需要在主动对等体端配置被动对等体的地址。

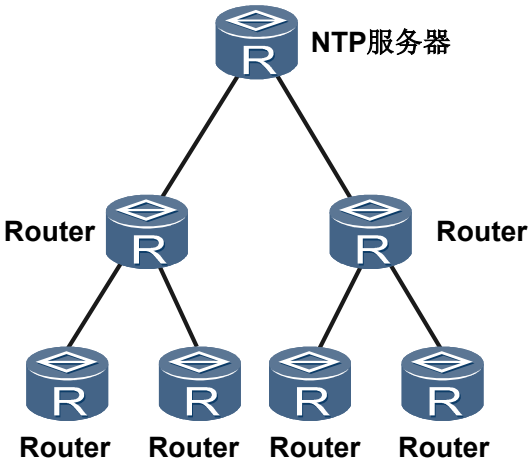
  这个时候在主动对等体上会建立一个静态连接，当被动对等体收到主动对等体发送的第 17 个报文：
  - 在无验证的情况下双方会自动建立动态连接
  - 在有验证的情况下双方必须通过验证才会建立一个动态连接
- 在组播模式中，用户在组播服务器端的接口下配置服务器的地址，这个时候在服务器端会建立一个静态连接，在组播客户端，用户也需要使能客户端模式，但是不建立静态连接，而是在收到组播服务器过来的报文之后会动态建立连接。
- 在广播模式中，用户在广播服务器端的接口下使能服务器模式，这个时候在服务器端会建立一个静态连接，在广播客户端，用户也需要使能客户端模式，但是不建立静态连接，而是在收到广播服务器过来的报文之后会动态建立连接。

5.5 应用

如图 5-8 所示，NTP 服务器作为 NTP 的主时钟，下面的各个 Router 设备作为 NTP 的客户端，同步 NTP 服务器的时钟。这样整个网络的设备的时钟都是保持一致的。

AR 设备作为 NTP 客户端部署。

图 5-8 NTP 应用示意图



5.6 术语与缩略语

术语

术语	解释
NTP	Network Time Protocol—网络时间协议，是用于互联网中时间同步的应用层协议，其用途是在分布式时间服务器和客户端之间进行时间同步，把主机的时间同步到某些时间标准。
时间戳	NTP 时间戳由 64 位无符号定点数表示，这个定点数是一个相对于 1900 年 1 月 1 日 0 点的秒数，整数部分位于前 32 位，小数部分位于后 32 位。 ● Originate Timestamp(peer.xmt, pkt.xmt)：表示 NTP 报文离开发送端时的当地时间（如 T_1），时间戳格式。 ● Receive Timestamp(peer.rec, pkt.rec)：表示 NTP 报文到达远端对等体时的当地时间（如 T_2），时间戳格式。当远端对等体不可达时，该值被置为 0。 ● Transmit Timestamp (peer.org, pkt.org)：表示远端对等体返回 NTP 报文时的当地时间（如 T_3），时间戳格式。当对等体不可达时，该值被置为 0。 ● Reference Timestamp(sys.reftime, peer.reftime, pkt.reftime)：表示 NTP 报文回到发送端时的当地时间（如 T_4），时间戳格式。如果本地时钟从未被同步过，值为 0。
时钟偏移量	时钟偏移量是本地时钟与参考时钟之间的时间差。在数值上等于将本地时钟调节到与所选参考时钟一致所要调节的量。
往返延迟	往返延迟是客户端收回 NTP 报文的时刻，与发出该 NTP 报文的时刻之间的时间差。它规定了本地时钟在指定时间内将一条信息发送到参考时钟的能力。
离差	离差是本地时钟相对于参考时钟的最大误差。
层数	层数是对始终同步情况的一个分级标准，代表了一个时钟的精确度，取值范围 1 ～ 16，数值越小，精确度越高。1 表示时钟准确度最高，16 表示未同步。
时钟过滤	时钟过滤针对本地时钟的同一个对等体，用来从这个给定的对等体选择最好的时间样本。
时钟选择	时钟选择是一种选择参考时钟的方法，基于时钟选择算法。

缩略语

缩略语	英文全称	中文全称
NTP	Network Time Protocol	网络时间协议
VRP	Versatile Routing Platform	通用路由平台

缩略语	英文全称	中文全称
UDP	User Datagram Protocol	用户数据报协议

6 NQA

关于本章

6.1 介绍

定义

NQA（Network Quality Analysis），即网络质量分析，是系统提供的一个特性，位于链路层之上，覆盖网络层、传输层和应用层，独立于底层硬件。

目的

设备提供 NQA 主要目的是帮助运营商实时监视网络 QoS，同时在网络发生故障时进行有效的故障诊断和定位。

为了使网络服务质量可见，使用户能够自行检查网络服务质量是否达到要求，运营商需要采取以下措施：

- 在设备上提供能够说明网络服务质量的数据。

  由于 IP 网络统计复用、流量突发的特点，NQA 只能以一种统计的方法描述。运营商需要在设备侧提供时延、抖动、丢包率等相关统计参数。
- 在网络中部署探针设备能网络服务质量进行监控。

  随着网络规模的不断增大，如果使用专用的探针设备(如第三方的探测设备：Brix)，则所需设备数量将不断增加，运营商的成本就会相应的增加。

设备提供 NQA（Network Quality Analysis）功能。在设备上集成网络质量测试功能，不仅可以实现对网络运行状况的准确测试，输出统计信息。而且，由于不用部署专门的探针设备，设备提供的 NQA 还有效的节约了运营商的成本。

NQA 可以监测网络上运行的多种协议的性能，使运营商能够实时采集到各种网络运行指标，例如：HTTP 的总时延、TCP 连接时延、DNS 解析时延、文件传输速率、FTP 连接时延、DNS 解析错误率等。

使用传统的 **Ping** 命令也可以监测网络质量，但是相比 NQA 来说，Ping 获取的信息非常有限。下文将从功能，配置和调度三方面来比较 NQA 和 Ping 的差异。

表 6-1 NQA 与 Ping 的比较

	NQA	命令行 Ping
功能差异	NQA 可以探测 TCP、UDP、DHCP、FTP、HTTP、SNMP、DNS 等服务是否启动，以及测试各种服务的响应时间。NQA 还可以通过 Jitter 测试来计算网络抖动情况。 NQA 缺省情况下，最多支持配置 256 个测试例。	Ping 功能只能使用 ICMP 协议来测试数据包在本端和指定的目的端之间的往返时间及目的端是否可达。 一个用户只能发起一个 Ping。

	NQA	命令行 Ping
配置差异	NQA 功能可以在客户端执行命令查看 NQA 操作的统计结果。注意事项如下： ● NQA 功能可以通过网管来设置各项操作的参数，并启动测试操作，通过测试结果和历史表中的数据取得统计信息。 ● 在大多数的测试项中，只需要配置 NQA Client 端就可以了。但在进行 TCP、UDP 和 Jitter 类型的测试时，还必须配置 NQA 服务器。 ● NQA 服务器通过监听功能实现对客户端发起的测试的响应。只有在 NQA 服务器上配置了相应的目的地址和端口号，客户端发起的请求才能得到服务器的响应，而且服务器监听服务中指定的 IP 地址、端口号要和客户端配置的参数相一致。	Ping 功能在控制台终端执行 **Ping** 命令，测试指定的 IP 地址是否可达，会实时显示每个包的往返时间或超时。
调度方式	实现对测试例的调度，降低由于测试的并发执行，导致设备负担过重。 对单个测试例，支持多种启动时间、结束时间的设置。 ● 支持立即启动、延迟启动、定时启动。 ● 支持报文发送完后自动结束、立即结束、延迟结束、定时结束、生命周期结束。 多个任务同时启动时，设备主动合理分布启动时间和测试间隔。	只支持命令行下发。

受益

NQA 可以方便的探测多种类型的连接在网络中的质量状况，并可以帮助快速定位网络故障，提升了用户的维护效率。

6.2 参考标准和协议

本特性的参考资料清单如下：

文档	描述	备注
RFC1889	RTP: A Transport Protocol for Real-Time Applications	-
RFC2925	Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations	-
RFC2131	Dynamic Host Configuration Protocol	-
RFC1035	DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION	-

文档	描述	备注
RFC414	FILE TRANSFER PROTOCOL (FTP) STATUS AND FURTHER COMMENTS	-
RFC1945	Hypertext Transfer Protocol – HTTP/1.0	-
RFC2616	Hypertext Transfer Protocol - HTTP/1.1	-
RFC792	INTERNET CONTROL MESSAGE PROTOCOL	-
RFC4379	Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures	-
IEEE 802.1AG DRAFT6.1	IEEE 802.1AG DRAFT6.1	-
RFC792	INTERNET CONTROL MESSAGE PROTOCOL	-
DRAFT-FENNER-TRACEROUTE-IPM-07	DRAFT-FENNER-TRACEROUTE-IPM-07	-
RFC1157	A Simple Network Management Protocol (SNMP)	-
RFC3414	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)	-
RFC1905	Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)	-
RFC793	Transmission Control Protocol	TCP/UDP 测试
RFC862	Echo Protocol	TCP/UDP 测试
RFC1393	Traceroute Using an IP Option	TRACE 测试

6.3 可获得性

涉及网元

网络中各种与 AR 连接的被测试设备。

License 支持

无需获得 License 许可，均可获得该特性的服务。

版本支持

产品	最低支持版本
AR3200	V200R001C00

6.4 原理描述

NQA 把测试两端称为客户端和服务器端，NQA 的测试是由客户端发起。在用户端通过命令行配置测试例或由网管端发送相应配置操作后，NQA 把相应的操作类型放入到测试例队列中进行调度。

启动 NQA 测试例，可以选择立即启动、延迟启动、定时启动。在定时器的时间到达后，则根据测试例的测试类型，构造符合相应协议的报文。但配置的测试报文的大小如果无法满足发送本协议报文的最小尺寸，则按照本协议规定的最小报文尺寸来构造报文发送。

测试例启动后，根据返回的报文，可以对相关协议的运行状态提供数据信息。发送报文时的系统的时间作为测试报文的发送时间，给报文打上时间戳，再发送给目的端。目的端接收报文后，返回给源端相应的回应信息，源端在接收到报文时，再一次读取系统的时间，给报文打上时间戳。根据报文的发送和接收时间，计算出报文的往返时间。

说明

对于 Jitter 测试例，不仅源端需要在源端给报文打时间戳，而且目的端在接收到报文和发送报文时，也要读取自己的本地系统时间，再打上时间戳，从而能够计算出抖动时间。

这样用户就可以通过查看测试数据信息了解到网络的运行情况。

6.4.1 UDP Jitter 测试

UDP Jitter 是以 UDP 报文为承载，通过记录在报文中的时间戳信息来统计时延、抖动、丢包的一种测试方法。Jitter（抖动时间）是指相邻两个报文的接收时间间隔减去这两个报文的发送时间间隔。如图 6-1 所示，Jitter 测试的过程如下：

1. 源端（RouterA）以一定的时间间隔向目的端（RouterB）发送数据包。
2. 目的端每收到一个数据包，就给它打上时间戳，然后再把这个数据包发回到源端。
3. 源端收到数据包后通过计算目的端接收数据包时间间隔和源端发送数据包的时间间隔之差，计算出抖动时间。

   说明

   UDP Jitter 每次测试最大发包数量可配，是探测数（probe-count）与每次探测发送报文（jitter-packetnum）的乘积。

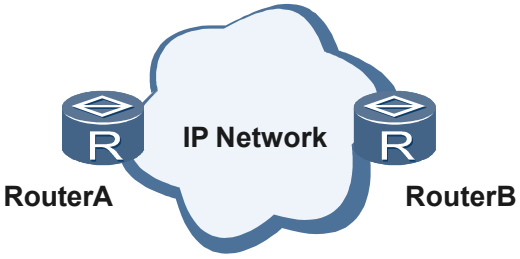
   从源端接收到的信息中计算出：

   - 数据包从源端（Client）到目的端（Server）和从目的端到源端的最大抖动时间、最小抖动时间及平均抖动时间。
   - 从目的端到源端或从源端到目的端的最大单向延时。

   从而清晰的反映出网络状况。

Jitter 测试可以设置单个测试例的连续发包数目，通过这项设置，可以在一段时间内模拟某种数据的真实流量。例如，设置 3000 个 UDP 报文以 20 毫秒的间隔发送，可以在一分钟内模拟 G.711 流量。

图 6-1 UDP Jitter 应用场景



6.4.2 HTTP 测试

如图 6-2 所示，NQA 的 HTTP 测试提供三个阶段的响应速度：

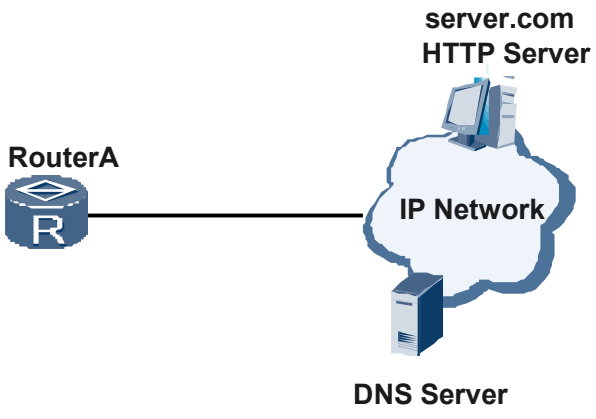
- DNS 解析时间：客户端发送 DNS 报文给名字解析器，将 HTTP 服务器名字解析为 IP 地址，DNS 解析报文返回的时间。
- TCP 建立连接时间：客户端与 HTTP 服务器通过 TCP“三次握手”建立连接所用的时间。
- 交易时间：客户端发送 Get 或 Post 报文给 HTTP 服务器，响应报文到达 HTTP 服务器的时间。

通过 HTTP 测试，从源端接收到的信息中可以计算出：

- 最小 DNS 查询时间、最大 DNS 查询时间及 DNS 查询时间总和。
- 最小 TCP 连接建立时间、最大 TCP 连接建立时间及 TCP 连接建立时间总和。
- 最小 HTTP 交易时间、最大 HTTP 交易时间及 HTTP 交易时间总和。

从而清晰的反映出网络 HTTP 协议的性能状况。

图 6-2 HTTP 应用场景

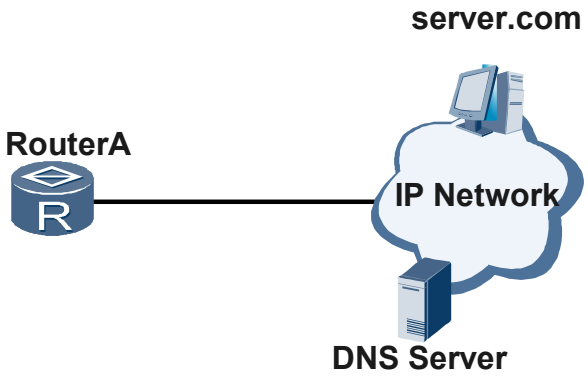


6.4.3 DNS 测试

NQA 的 DNS 测试用于检测将给定的 DNS 名称解析成 IP 地址的速度，以 UDP 报文为承载。如图 6-3 所示，DNS 测试的过程如下：

1. 源端向 DNS Server 发送要求解析给定的 DNS 名称的 Query 报文。
2. DNS Server 收到报文后，通过解析构造 Response 报文，然后再把这个数据包发回到源端。
3. 源端收到数据包后通过计算源端接收报文的时间和源端发送报文的时间的差，计算出 DNS 域名解析时间。从而清晰的反映出网络 DNS 协议的性能状况。

图 6-3 DNS 应用场景



6.4.4 FTP 测试

NQA 的 FTP 测试以 TCP 报文为承载，用于检测从 FTP 服务器下载指定文件或向 FTP 服务器上载指定文件的速度。如图 6-4 所示，FTP 测试提供两个阶段的响应速度：

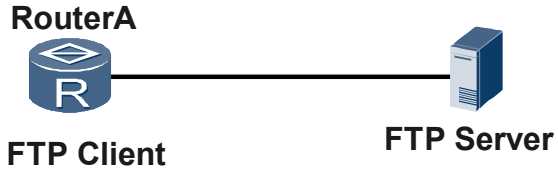
- 控制连接时间：源端与 FTP 服务器通过 TCP“三次握手”建立控制连接的时间以及通过控制连接交互信令的时间。
- 数据连接时间：源端通过数据连接从 FTP 服务器下载指定文件或向 FTP 服务器上载指定文件的时间。

通过 FTP 测试，从源端接收到的信息中可以计算出：

- 最小控制连接时间、最大控制连接时间及平均控制连接时间。
- 最小数据连接时间、最大数据连接时间及平均数据连接时间。

从而清晰的反映出网络 FTP 协议的性能状况。

图 6-4 FTP 应用场景



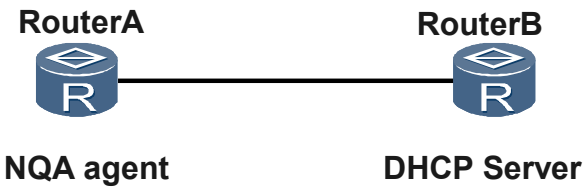
6.4.5 DHCP 测试

NQA 的 DHCP 测试以 UDP 报文为承载，用于检测从 DHCP 服务器获得地址的速度。如图 6-5 所示，DHCP 测试过程如下：

1. 源端从需要获得地址的接口，向接口所在网段广播查询 DHCP Server 的 Discovery 报文。
2. DHCP Server 收到报文后，向源端回送 Offer 报文，报文中包含了 DCHP Server 的 IP 地址。
3. 源端向接口所在网段广播要求获取 IP 地址的 Request 报文，报文中包含了 DHCP Server 的 IP 地址的信息。
4. DHCP Server 收到报文后，向源端回送 ACK 报文，报文中包含了 DHCP Server 分配给相应接口的 IP 地址。

   源端收到数据包后通过计算源端接收报文的时间和源端最初发送 Discovery 报文的时间的差，计算出从 DHCP 服务器获取 IP 地址的时间。从而清晰的反映出网络 DHCP 协议的性能状况。

图 6-5 DHCP 应用场景



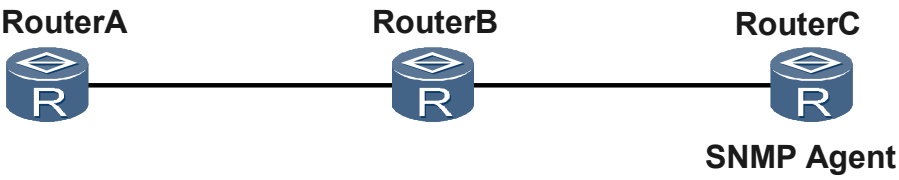
6.4.6 SNMP 测试

NQA 的 SNMP 测试用于检测主机与 SNMP Agent 之间通信的速度，以 UDP 报文为承载。如图 6-6 所示，SNMP 测试的过程如下：

1. 源端（RouterA）向 SNMP Agent（RouterC）发送要求获取系统时间的请求报文。
2. SNMP Agent 收到报文，查询系统时间并构造回应报文，然后再把这个数据包发回到源端。

   源端收到数据包后通过计算源端接收报文的时间和源端发送报文的时间的差，计算出源端与 SNMP Agent 之间通信的时间。从而清晰的反映出网络 SNMP 协议的性能状况。

图 6-6 SNMP 应用场景



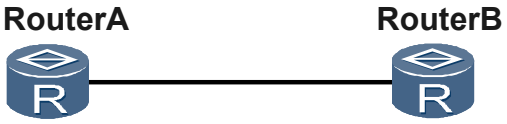
6.4.7 TCP 测试

NQA 的 TCP 测试用于检测主机与 TCP Server 之间经过三次握手建立 TCP 连接的速度。如图 6-7 所示，TCP 测试的过程如下：

1. RouterA 向 RouterB(TCP Server)发送要求建立连接的 TCP SYN 报文。
2. TCP Server 收到报文，接受请求并向源端回应 TCP SYN ACK 报文。
3. 源端收到数据包后，向 TCP Server 回应 ACK 报文，连接建立。

   此后，源端通过接收报文和发送报文的时间的差，计算出与 TCP Server 之间三次握手建立 TCP 连接的时间。从而清晰的反映出网络 TCP 协议的性能状况。

图 6-7 TCP 应用场景



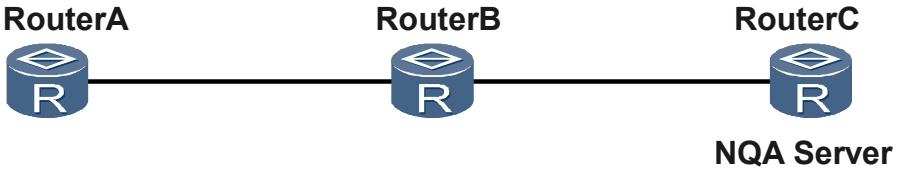
6.4.8 UDP 测试

NQA 的 UDP 测试用于检测主机与 UDP Server 之间通信的速度。如图 6-8 所示，UDP 测试的过程如下：

1. RouterA 向 RouterC(UDP server)发送构造的 UDP 报文。
2. UDP server 收到报文，直接将报文再回送给 RouterA。

   源端收到数据包后通过计算源端接收报文的时间和源端发送报文的时间的差，计算出源端与 UDP server 之间通信的时间。从而清晰的反映出网络 UDP 协议的性能状况。

图 6-8 UDP 应用场景



6.4.9 ICMP 测试

NQA 的 ICMP 测试例用于检测 NQA Client 到目的端的路由是否可达。ICMP 测试提供类似于普通命令行下的 Ping 命令功能，但输出信息更为丰富：

- 默认情况下能够保存最近 5 次的测试结果。
- 结果中能够显示平均时延，丢包率，最后一个报文正确接收的时间等信息。

如图 6-9 所示，ICMP 测试的过程如下：

1. Client(RouterA)端向目的端发送构造的 ICMP echo request 报文。

2. 目的端(RouterB)收到报文后，直接回应 ICMP echo reply 报文给 Client 端。

   Client 端收到报文后，通过计算源端接收时间和源端发送时间之差，计算出源端到目的端的通信时间，从而清晰的反应出网络通路畅通情况。

图 6-9 ICMP 应用场景

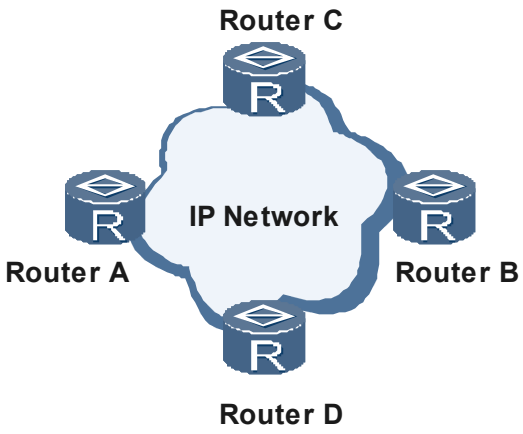
6.4.10 Trace 测试

NQA 的 Trace 测试例用于检测 NQA Client 到目的端的转发路径，并沿该路径记录源设备到中间各个设备的时延等信息。Trace 测试例类似于普通命令行下的 tracert 命令功能，但输出信息更为丰富：每一跳信息中能够显示平均时延，丢包，最后一个包接收时间等信息。如图 6-10 所示，Trace 测试的过程如下：

1. Client 端(RouterA)向目的端(RouterB)发送构造的 UDP 报文，报文中的 TTL 为 1。
2. 第一跳 RouterC 收到该报文后，判断 TTL 后丢弃该报文，返回一份超时 ICMP 报文。
3. Client 端(RouterA)收到该 ICMP 报文后，记录下第一跳的 IP 地址，并重新构造一份 UDP 报文，报文中的 TTL 为 2。
4. 报文到达第二跳 RouterD 后，判断 TTL 后丢弃该报文，返回一份超时 ICMP 报文。
5. 以此类推，最终报文到达最后一跳设备，返回一份端口不可达的 ICMP 报文给 Client 端。

   Client 端(RouterA)收到每跳返回的 ICMP 报文后，统计并打印出从源端到目的端的转发路径和该路径上各设备的有关信息。从而清晰的反映出从源到目的主机的转发路径，以及该路径上各设备的有关的统计信息。

图 6-10 Trace 应用场景



6.5 应用

NQA 进行网络诊断

运营商经常会遇到的问题有：上网时断时续、不能访问相关的站点、上网慢、下载文件慢等。需要在 AR3200 上有服务等级协定 SLA（Service Level Agreement）的统计功能，给出相关的统计，定位出问题的具体位置，以便给用户一个可以接受的解释。这些数据最终需要设备提供。

图 6-11 NQA 进行网络诊断

如图 6-11 所示，用户位于不同的位置的分公司之间，通过运营商提供的 VPN 网络，实现互连。但最近用户反映，网络出现时断时续的情况，即使能够连接，连接的速度也很慢。

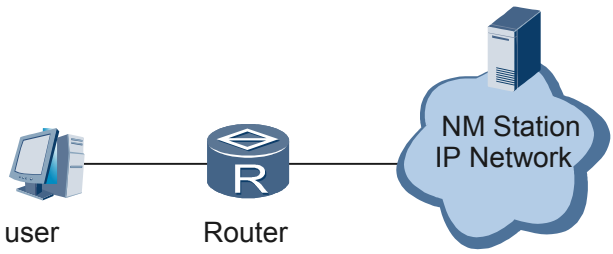
运营商通过在 PE 端部署的 NQA，对网络的质量进行分析。首先测试 PE 和 CE 之间进行 ICMP 的测试，查看网路的连通性。确认网络的连通后，进行 Jitter 测试，查看网络抖动情况。之后在 PE 之间进行同样的测试，通过对测试后的统计数据和用户遇到的问题进行分析，为故障的定位提供依据。

了解网络服务质量

用户需要清楚的了解到运营商对用户提供的网络质量的承诺是否兑现。

如图 6-12 所示，用户 user 通过运营商提供的网络接入到 Internet 中，并和运营商签订相应的服务等级协议。运营商可以通过部署在网络中的网管系统执行的 NQA 测试结果，得到网络运行情况的统计数据。运营商把统计数据提供给用户，用户就可以了解到运营商网络提供的情况，是否兑现了运营商对用户的承诺。

图 6-12 了解网络服务承诺



6.6 术语与缩略语

缩略语

术语	英文全称	中文全称
NQA	Network Quality Analysis	网络质量分析
QoS	Quality of Service	服务质量
LSP	Label Swtich Path	标签交换路径
FTP	File Transfer Protocol	文件传输协议
DHCP	Dynamic Host Cnfiguration Potocol	动态主机配置协议
HTTP	Hypertext Transfer Protocol	超文本传输协议
PWE3	Peudo Wre Eulation Ege-to-Ege	端到端伪线仿真
MPLS	Multiprotocol Label Switching	多协议标记交换
UDP	User Datagram Protocol	用户数据报协议
TCP	Transport Control Protocol	传输控制协议
DNS	Domain Name Service	域名业务
MD	Maintenance Domain	维护域
MEP	Maintenance Association End Point	维护联盟边缘节点
CCM	Continuity Check Message	连续监测报文
LBM	Loopback Message	环回消息
LBR	Loopback Reply	环回回复

7 NetStream

关于本章

7.1 介绍

定义

Internet 的高速发展，为用户提供了更高的带宽和可预测的 QoS，随着网络支持的业务和应用日渐增多，运营商需要对网络进行更加细致的管理，这样就对流量统计分析提出了更高的要求。NetStream 是一种基于网络流信息的统计与发布技术。

目的

1. 计费

   由于 NetStream 采集的信息非常全面，因此可以进行非常灵活的计费：

   - 通过流的时间信息可以分时间段、使用不同的计费标准。
   - 通过对 TOS（服务类型）类型流量的聚合可以按照服务类型（TOS）进行收费。
   - 通过对协议类型和 TCP/UDP 端口号的聚合可以实现从链路层到传输层的统计，并相应地进行收费。
   - 通过计算带宽占用情况可以对带宽的比率进行非线性收费（比如 10M 以下是一个收费标准，10M 以上收费标准增加）。
   - 通过 AS 域的聚合可以对网络内部和网络外部采用不同的收费标准（访问内部服务器和 Internet 收费标准不同），在运营商之间实现按照流量结算。
   - 通过对目的 IP 地址前缀的聚合，可以对访问某一网段的流量进行特殊的计费。

2. 网络规划和分析

   NetStream 可以为网络管理工具提供关键信息，以便优化网络设计和规划，达到最佳的网络性能和可靠性。对于运营商来说，可以进行如下分析：

   - 统计不同 AS 之间的流量。
   - 用统计信息规划网络建设，确定网络带宽是否够用，网络的拓扑结构是否合理。
   - 找出升级设备对流量中断影响最小的时段。
   - 为运营商之间结算网络计费提供依据。
   - 为运营商内部不同的分支机构提供结算网络计费的依据。

3. 应用监控和分析

   通过 NetStream 技术，可以获得详细的网络应用信息。例如，网络管理员可以查看 Web、文件传输协议(FTP)、Telnet 和其它 TCP/IP 应用所占通信量的百分比。Internet 内容和服务提供商可以根据这些信息来规划、分配网络和应用资源以满足用户需求。

4. 用户监控和分析

   NetStream 技术使得网络管理者可以获得用户利用网络和应用资源的详细情况，进而用于高效地规划和分配资源，并保障网络的安全运营。

受益

NetStream 基于“流”进行流量采集和聚合：

可以为基于资源（如线路、带宽、时段等）占用情况的计费提供了精细的数据。

可以为先进的网络管理工具提供关键信息，以便优化网络设计和规划，实现以最小的网络运营成本达到最佳的网络性能和可靠性。

能够实现近于实时的网络监控功能和全网范围内的流量模式，并提供预先故障检测、高效故障排除和快速问题解决功能，提供安全监控等应用和分析。

NetStream 将不断推进网络流量向分析技术的发展，为广大运营商计费结算、网络规划、网络运维提供数据支撑。

7.2 参考标准和协议

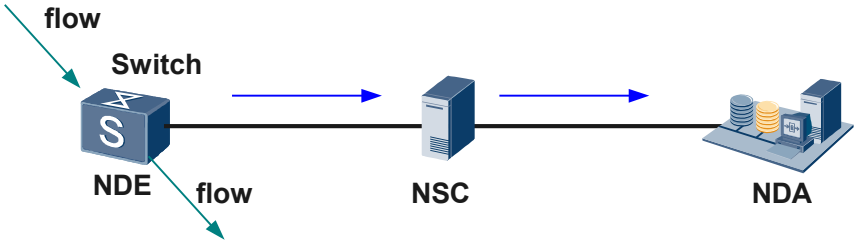
文档	描述	备注
RFC 3917	Requirements for IP Flow Information Export (IPFIX)	-
RFC 3954	Cisco Systems NetFlow Services Export Version 9	-

7.3 可获得性

涉及网元

NetStream 主要包括三个设备 NDE（NetStream Data Exporter），NSC（NetStream Collector），NDA（NetStream Data Analyzer），三个设备之间的关系如下图所示。

图 7-1 NetStream 特性涉及的网元



- NDE：对网络流进行分析处理，提取符合条件的流统计信息，并将统计信息输出给 NSC 设备，输出前也可对数据进行一些处理，比如聚合。
- NSC：NetStream Collector 是负责解析 NDE 的报文，把统计数据收集到数据库中，可供 NDA 进行解析。NSC 可以采集多个 NetStream 设备输出的数据，对数据进行过滤和聚合。
- NDA：NetStream Data Analyzer 是一个网络流量分析工具，从 NSC 中提取统计数据，进行后续处理，为各种业务提供依据（比如网络规划，攻击监测），具有图形化用户界面，使用户可以获取、显示和分析从 NetStream Collector 收集的数据。

License 支持

不涉及。

版本支持

产品	最低支持版本
AR3200	V200R001C00

7.4 原理描述

IP 流的统计

对一个接口接收的 IP 报文按照源地址、目的地址、源端口、目的端口、协议、ToS 和入接口索引进行分类，每一类报文构成一条 NetStream 流，在流分类的基础上进行统计和转发。

流的老化

由于网络上的流是短时间阵发的，在几秒时间就会产生数万条流，而 NDE 的内存的容量是一定的，这样就需要把当前的部分流删除，为后面到来的流提供内存空间，这个过程成为老化。

- 定时老化
- 对不活跃时间（从最后一个报文到达到当前的时间）超过 inactive timeout，或活跃时间（从第一个报文流过到当前的时间）超过 active timeout 的流进行老化。
- 由 TCP 连接的 FIN 和 RST 报文触发老化

  对于 TCP 连接，当有标志为 FIN 或 RST 的报文发送时，表示一次会话结束。因此当一条已经存在的 TCP 协议 NetStream 流中流过一条标志为 FIN 或 RST 的报文时，可以立即把相应的 NetStream 流老化掉。
- 字节数计数溢出老化

  流缓存区中的流需要记录流过的字节数，当报文字节数量超过定义的变量上限时，缓存区就会溢出。所以系统在检测到某条流的字节数统计超过限制时，就立即把该流老化掉。

流的输出

- 原始流的输出方式

  老化后的 NetStream 流中信息被系统统计后，以 UDP 报文形式送入到 NSC，NSC 可以得到流的详细信息，可以对这些流记录进行更为灵活的后续处理。
- 聚合流的输出方式

  老化后的 NetStream 流中信息被系统统计后，原始信息按照一定的规则进行分类、合并后生成聚合的信息，在系统中的定时器到时后，聚合流通过 UDP 报文发送出去。通过对原始流的聚合，可以明显减少网络带宽、CPU 占用率和存储介质空间。支持如表 7-1 所示的聚合方式。

表 7-1 聚合方式列表

聚合方式	说明
As	自治系统聚合，根据 NetStream 流的源、目的自治系统号，输入接口索引，输出接口索引，4 个关键值进行分类，具有相同关键值的流合并成一条聚合的流，并对应一条聚合信息。
as-tos	自治系统-TOS 聚合，根据 NetStream 流的源、目的自治系统号，输入接口索引，输出接口索引，ToS，5 个关键值进行分类，具有相同关键值的流合并成一条聚合的流，并对应一条聚合信息。
protocol-port	协议-端口聚合，根据 NetStream 流的协议号，源端口，目的端口，3 个关键值进行分类，具有相同关键值的流合并成一条聚合流，并对应一条聚合记录。
protocol-port-tos	协议-端口-ToS 聚合，根据 NetStream 流的协议号，源端口，目的端口，ToS，输入接口索引，输出接口索引，6 个关键值进行分类，具有相同关键值的流合并成一条聚合流，并对应一条聚合记录。
source-prefix	源前缀聚合，根据 NetStream 流的源自治系统号，源掩码长度，源前缀，输入接口索引，4 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
source-prefix-tos	源前缀-ToS 聚合，根据 NetStream 流的源自治系统号，源掩码长度，源前缀，ToS，输入接口索引，5 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
destination-prefix	目的前缀聚合，根据 NetStream 流的目的自治系统号，目的掩码长度，目的前缀，输出接口索引，4 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
destination-prefix-tos	目的前缀-Tos 聚合，根据 NetStream 流的目的自治系统号，目的掩码长度，目的前缀，ToS，输出接口索引，5 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
prefix	前缀聚合，根据 NetStream 流的源、目的自治系统号，源、目的掩码长度，源、目的前缀，输入接口索引，输出接口索引，8 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
prefix-tos	前缀-ToS 聚合，根据 NetStream 流的源、目的自治系统号，源、目的掩码长度，源、目的前缀，ToS，输入接口索引，输出接口索引，9 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。

- 灵活流的输出方式：

对于灵活流方式，其流的建立条件是按照自定义的条件设置的，并以 V9 自定义格式发送给 NSC/NDA 分析。灵活流方式相比原始流方式可减少流量的占用。

输出报文版本和格式

目前 Netstream 输出的报文主要有 5、8、9 三个版本，其他的版本处于实验阶段，没有商用。所有的版本都是通过 UDP 协议传递统计信息的。每个数据包都包括一个 Packet Header 再加上一条或者几条流的记录信息。

NetStream 原始流输出报文支持版本 5 和版本 9 两种报文格式，聚合流输出支持版本 8 和版本 9 两种报文格式。

版本 9 和以前的版本有很大的不同，它是基于模板方式的，使统计信息的输出更为灵活，而且更容易扩展新定义流的元素以及生成新的记录。版本 9 和版本 5、版本 8 不兼容。

Netstream 报文如图 7-2 所示。

图 7-2 NetStream 报文格式

- 版本 5

  版本 5 的 NetStream 报文头

  版本 5 报头字段说明如下表所示。

  表 7-2 版本 5 报头字段说明

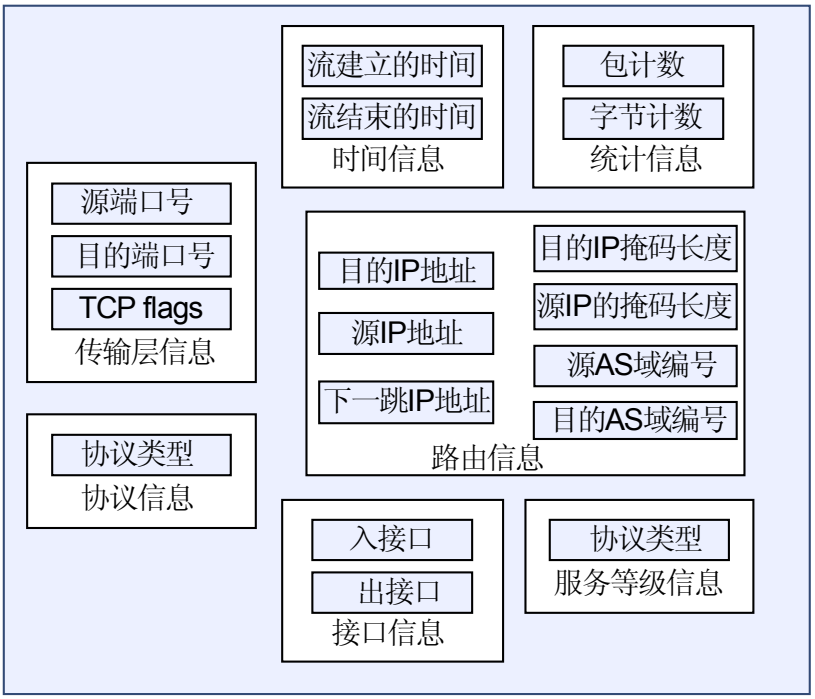
字段	描述
version	NetStream 输出报文格式版本编号，对于 V5，为 0x05
count	当前报文中的流记录数（1-30）
SysUptime	报文产生的时间，是系统启动以来的毫秒数
unix_secs	从 1970 年 1 月 1 日 0 时起，到报文产生时间的整秒数
unix_nsecs	报文产生时间的纳秒数，也即不足一秒的余下的纳秒数

字段	描述
flow_sequence	输出的流记录的顺序号， 在第一个 NetStream 报文中，此值为 0，count = c1， 在第二个 NetStream 报文中，此值为 c1，count = c2， 在第三个 NetStream 报文中，此值为 c2 + c1， … 在第 n － 1 个 NetStream 报文中，此值为 fs(n - 1)，count = c(n - 1) 在第 n 个 NetStream 报文中，此值为 fs(n - 1) + c(n - 1)。 利用此值可以判断报文是否丢失 当流序列号溢出时，按自然溢出继续进行。
engine_type	流交换引擎类型
engine_id	交换引擎槽号
reserved	保留字段，全零

版本 5 的报文包含的信息

版本 5 包括以下信息如图 7-3 所示

图 7-3 版本 5 信息



说明

对于 NetStream 出统计和入统计，分别生成各自独立的 UDP 版本 5 报文，两种报文格式一样，但是带有区别出统计和入统计的标志位。

- 版本 8
  - 版本 8 的 Netstream 报文头

    版本 8 报头字段说明如下表所示。

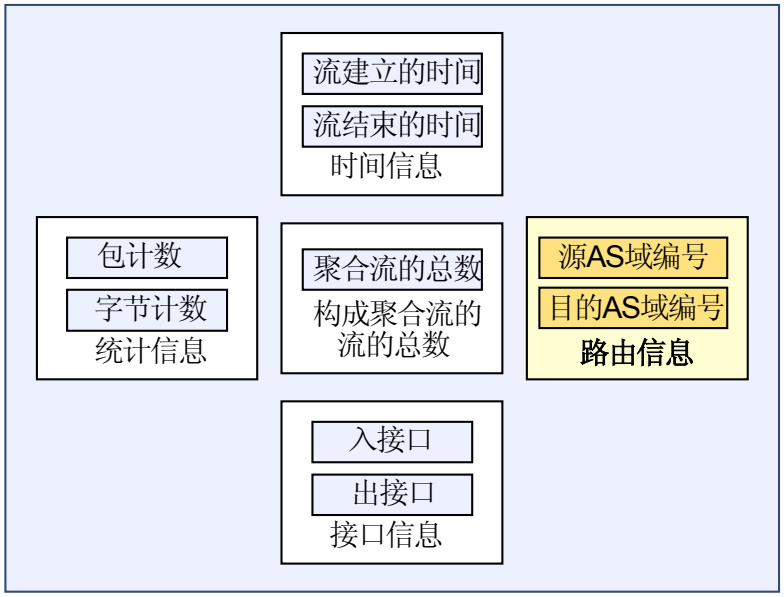
表 7-3 版本 8 报头字段说明

字段	描述
version	NetStream 输出报文格式版本编号，对于 V8，为 0x08。
count	当前报文中的流记录数，而不是流的总数
SysUptime	报文产生的时间，是系统启动以来的毫秒数
unix_secs	从 1970 年 1 月 1 日 0 时起，到报文产生时间的整秒数
unix_nsecs	报文产生时间的纳秒数，也即不足一秒的余下的纳秒数
flow_sequence	输出的流记录的顺序号， 在第一个 NetStream 报文中，此值为 0，Count=c1, 在第二个 NetStream 报文中，此值为 c1, Count=c2, 在第三个 NetStream 报文中，此值为 c1+c2， ... 在第 n-1 个 NetStream 报文中，此值为 fs(n-1)，Count=c(n-1) 在第 n 个 NetStream 报文中，此值为 fs(n-1) + c(n-1)。 利用此值可以判断报文是否丢失 当流序列号溢出时，按自然溢出继续进行。
engine_type	流交换引擎类型
engine_id	交换引擎槽号
aggregation	聚合类型，分别如下： 01 as 02 protocol-port 03 source-prefix 04 destination-prefix 05 prefix 09 as-tos 0a protocol-port-tos 0b source-prefix-tos 0C destination-prefix-tos 0d prefix-tos

字段	描述
agg_version	输出的聚合版本号，为 0x02

- 版本 8 和以前的版本相比有很大的不同。从版本 8 开始，具备了对流进行简单加工的能力——聚合。所谓聚合，就是把流按照某些信息进行归类统计。流聚合的最大好处是可以减少对网络带宽的占用。而在此之前的版本，聚合的工作是通过 NSC 来实现的。
  - 根据 AS 域进行聚合

图 7-4 根据 AS 域进行聚合示意图



这种聚合方式主要统计从一个 AS 域到另一个 AS 域的包和字节信息。可以用于运营商之间结算。

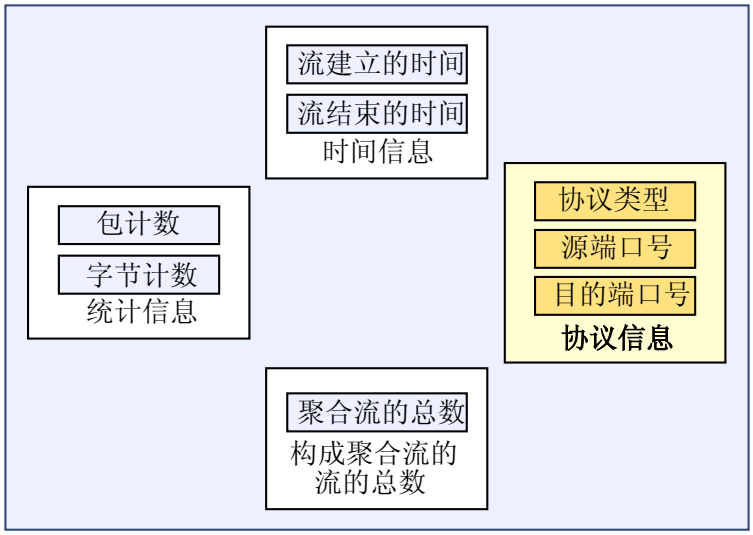
其中的源 AS 如果为 Origin，则源 AS 编号为源地址所属的 AS；如果为 Peer，则源 AS 编号为源地址的 AS PATH 的上一个 AS。对于属于本自治域的源地址，或从路由表中不能获取 AS 号，该值为 0。

对于目的 AS，如果为 Origin，则目的 AS 编号为目的地址所属的 AS；如果为 Peer，则目的 AS 编号为目的地址的 AS PATH 的上一个 AS。对于属于本自治域的目的地址，或从路由表中不能获取 AS 号，该值为 0。

源 AS 编号和目的 AS 编号这个两个字段是根据 AS 进行聚合的关键值。

  - 根据协议类型进行聚合

图 7-5 根据协议类型聚合示意图



这种方式针对传输层的协议类型（对于 TCP 和 UDP，还包括源和目的端口号）进行聚合。

协议类型字段取值为 6，表示是 TCP 协议；取值是 17，表示是 UDP 协议。

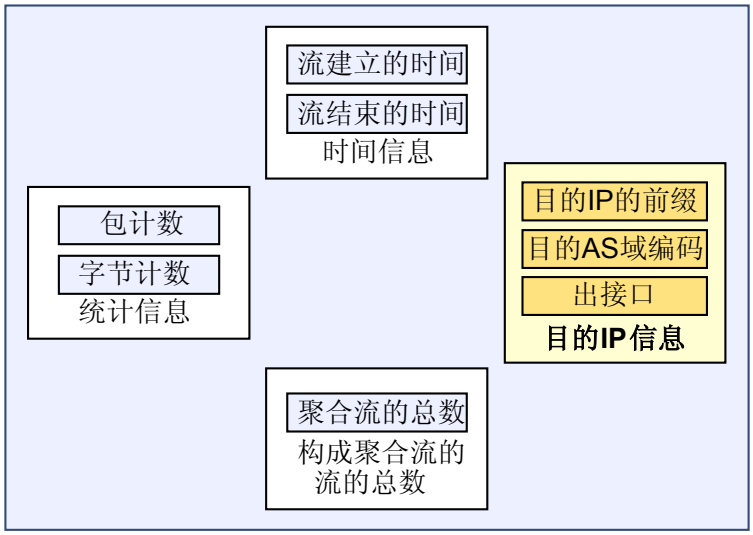
对于非 TCP 或 UDP 的报文，源端口号取值为 0。

对于 ICMP，目的端口号取值是 ICMP 报文的 Type 与 Code 字段。

对于非 TCP 或 UDP 报文，目的端口号取值是 0。

- 根据目的 IP 地址的前缀进行聚合

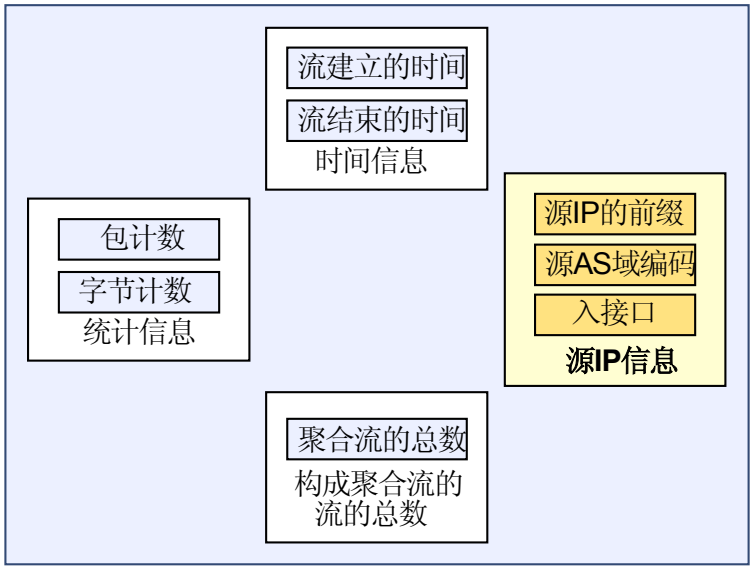
图 7-6 根据目的 IP 地址的前缀聚合示意图



这种方式针对目的 IP 地址的前缀进行聚合。

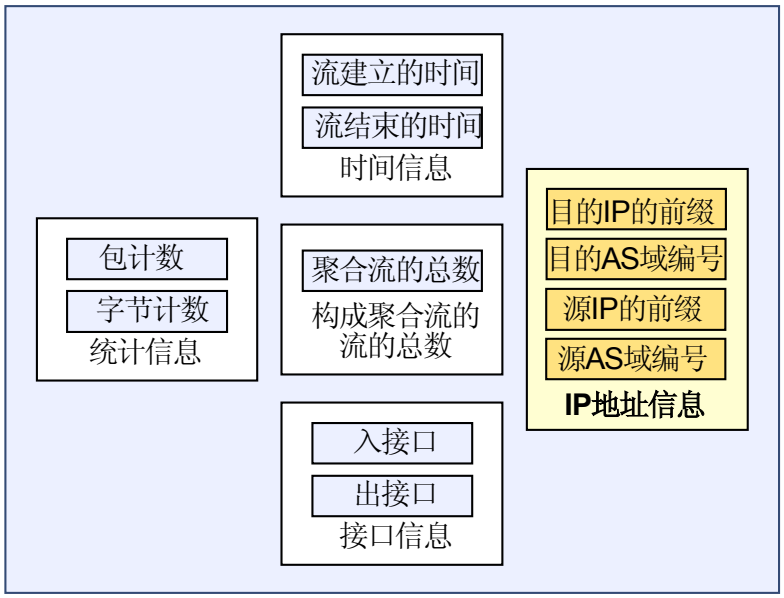
- 根据源 IP 地址的前缀进行聚合

图 7-7 根据源 IP 地址的前缀聚合示意图



- 根据源 IP 地址的前缀和目的 IP 地址的前缀进行聚合

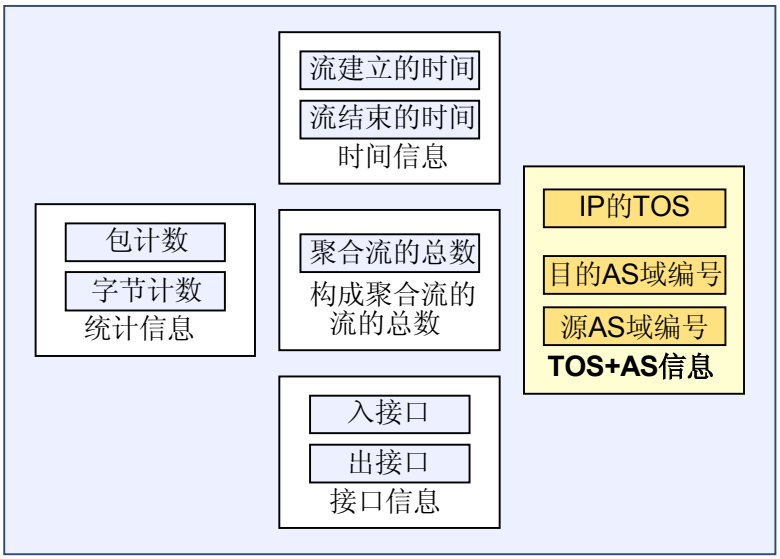
图 7-8 根据源 IP 地址的前缀和目的 IP 地址的前缀进行聚合示意图



这种方式下，源 IP 地址的前缀部分和目的 IP 地址的前缀部分都参与聚合。

- 根据 TOS ＋ AS 域进行聚合

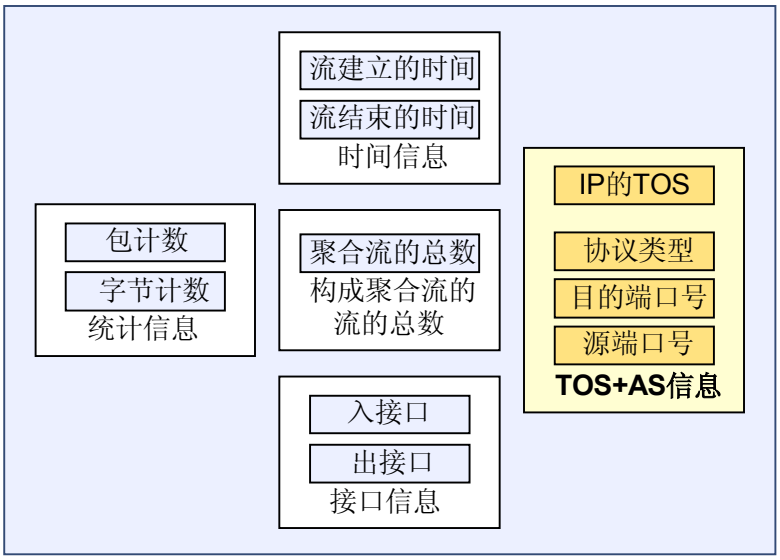
图 7-9 根据 TOS ＋ AS 域进行聚合示意图



其中 TOS（Type of Service）是 IP 报文头中的一个字段，用来设置报文的优先级。

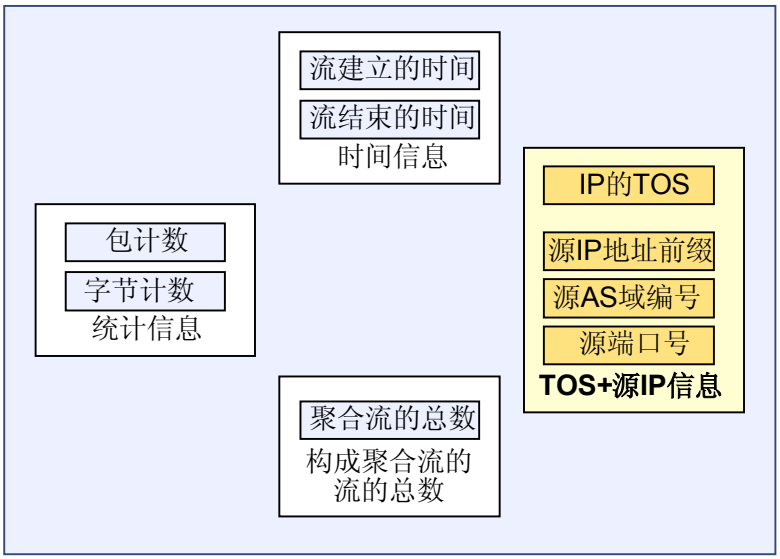
- 根据 TOS ＋协议类型进行聚合

图 7-10 根据 TOS ＋协议类型进行聚合示意图



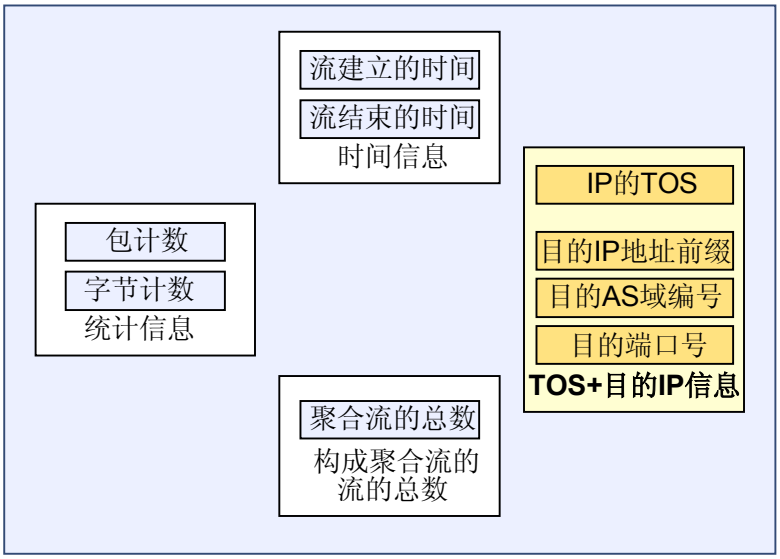
- 根据 TOS ＋源 IP 地址前缀进行聚合

图 7-11 根据 TOS ＋源 IP 地址前缀进行聚合示意图



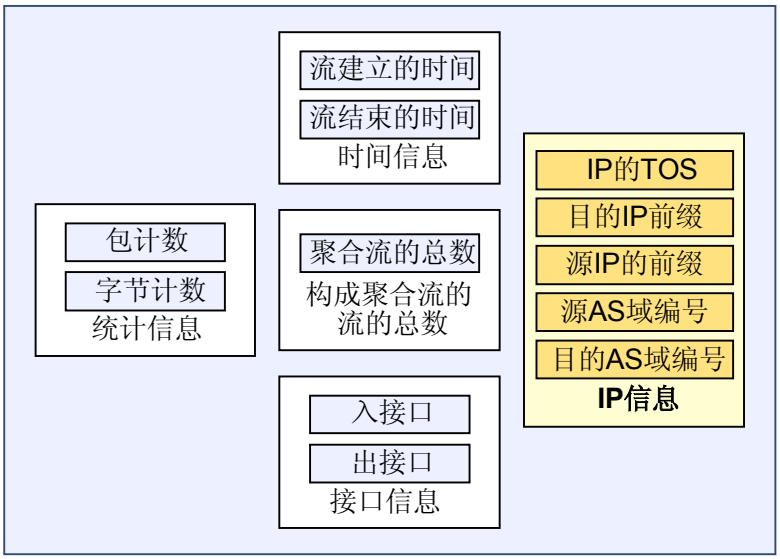
- 根据 TOS ＋目的 IP 地址前缀进行聚合

图 7-12 根据 TOS ＋目的 IP 地址前缀进行聚合



- 根据 TOS ＋ IP 地址前缀进行聚合

图 7-13 根据 TOS + IP 地址前缀进行聚合示意图



- 版本 9

  版本 9 最显著的特点是基于模板的方式，使统计信息的输出更为灵活，更容易扩展新定义流的元素以及生成新的记录。使用版本 9 可以实现 NAT、组播、MPLS、BGP 下一跳的统计。

  - 版本 9 的 Netstream 报文头

    版本 9 报文字段说明如下表所示。

表 7-4 版本 9 报头字段说明

字段	描述
version	NetStream 输出报文格式版本编号，对于 V9，其值是 0x09。
count	该报文包含的 FlowSet records(包括 Template 和 Data)的数目。
system up time	从系统启动到报文产生的时间，单位是毫秒。
unix_secs	从 1970 年 1 月 1 日 0 时起，到报文产生时间的整秒数。
flow sequence	所有输出报文的顺序号。该值是累积的，利用此值可以判断报文是否丢失。 注：该值和 V5、V8 报文头有所不同的是：在 V5 和 V8 中该值描述的是“所有的流”。

字段	描述
source id	Source ID 占 4 个字节，用来保证从 NDE 中输出的所有流的唯一性(Source ID 等同于 V5、V8 报文头中的 engine type 和 engine ID)。该值可以由用户定义。Source ID 前 2 字节保留用作以后的扩展，置 0。第 3 个字节描述输出设备的路由引擎的唯一性，填写设备类型。第 4 个字节描述板卡号，填写 Netstream 板的槽号。

- 版本 9 的 Export Packet，由 Packet Header、Template FlowSet 和 Data FlowSet 构成，格式的示意图如图 7-14 所示：

图 7-14 版本 9 的 Export Packet 格式

其中 Template FlowSet 和 Data FlowSet 是互相独立的。Data FlowSet 中的 Data Record 由 collector 已知的模板解释，也就是说，NSC 已经知道了 Data Record 中的 Template ID 对应的模板了。而 Template FlowSet 是告诉 NSC 一个即将被使用的模板，NSC 使用这个模板的时候只能是针对后续的 Export Packet。

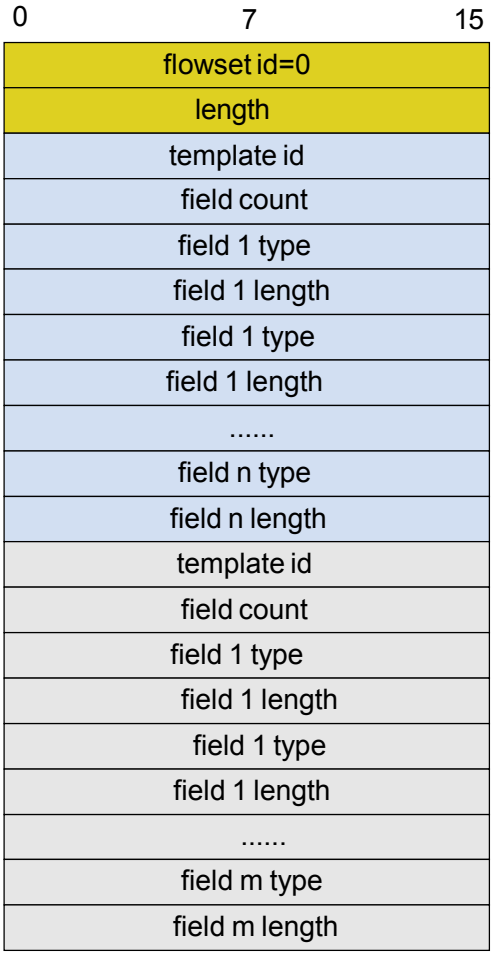
在 Export Packet 中不是必须要包含 Template FlowSet 和 Data FlowSet 的。可能的组合有：

Export Packet 中包含穿插着存放的 Template FlowSet 和 Data FlowSet，对于 NSC 来说，Template FlowSet 和 Data FlowSet 之间没有什么关联。NSC 将收到的 Template FlowSet 中的模板保存下来，用于后续的 Data FlowSet 的模板套用。

Export Packet 中仅有 Data FlowSet。如果 Template ID 都已经定义好了，使能 NetStream 的 NDE 传递给 NSC 的 Export Packet 一般属于这种情况。

Export Packet 中仅有 Template FlowSet，一般情况下，为了更好的利用网络带宽，Template FlowSet 都是和 Data FlowSet 打包到一个 Export Packet 中，因此这种情况是一个特例，一般出现在 NDE 已经配置好模板后重启的时候，NDE 需要尽快把所有的模板传递给 NSC，此时就不用等待生成统计流后 Template FlowSet 和 Data FlowSet 一同发送了。另外，模板具有有效时间，超过有效时间 NSC 会删除超时的模板，因此，需要定时的发送 Template FlowSet 到 NSC，如果需要发送的时候没有 Data FlowSet 生成，则此时只发送 Template FlowSet。

图 7-15 Template FlowSet 的格式

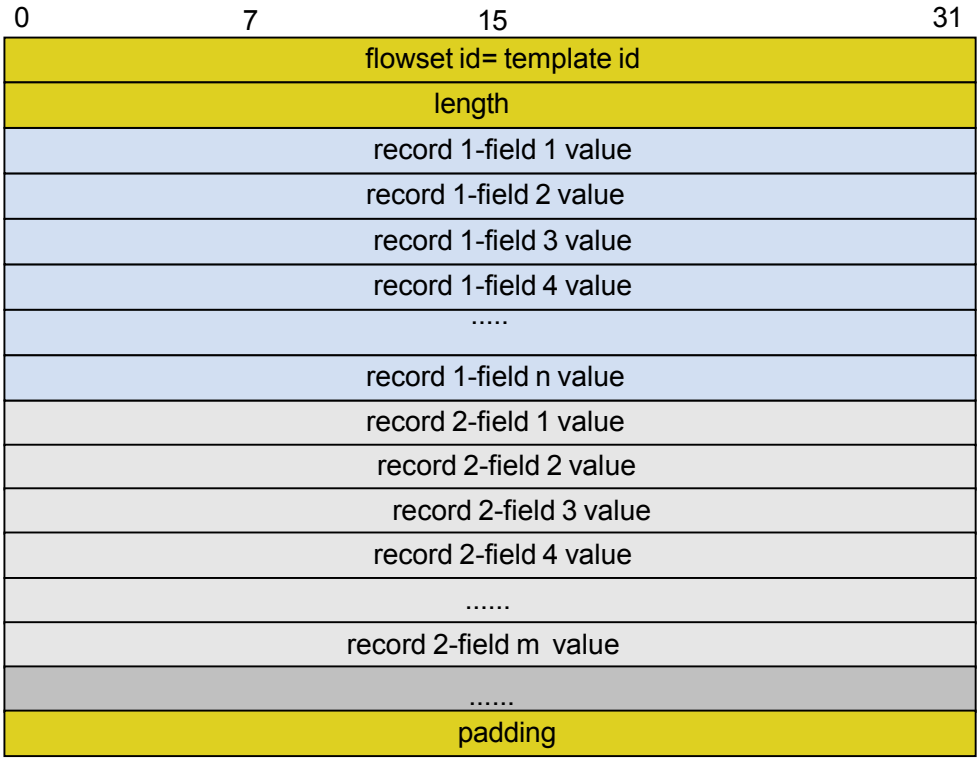


这是一个包含两个 Template Record 的例子。各字段的含义如下表所示。

字段	描述
FlowSet ID	FlowSet ID 用来区分模板记录(Template records)和数据记录(Data records)。对于 Template FlowSet，FlowSet ID 的取值是 0～255，对于 Data FlowSet，取值从 256 开始，这样 collector 就可以在 Export Packet 中识别出 Template FlowSet。
Length	Length 指的是 FlowSet 的总长度。因为一个单独的 Template FlowSet 可能包含多种模板 ID，长度值可以用来判断下一个 FlowSet 记录(可以是 Template FlowSet 或 Data FlowSet)的位置。 Length 是用 type/length/value(TLV)格式描述的，这也意味着该值包括了 FlowSet ID 和它自己的字节长度，也包括了当前 FlowSet 的所有 Template 记录的长度。

字段	描述
Template ID	当 NDE 生成不同的 Template FlowSets 来匹配将要输出的 NetStream 数据类型时，每个模板都会分配一个唯一的 ID。这个唯一性是针对生成模板 ID 的 NDE 的。因为 0 ～ 255 已经被 FlowSet IDs 使用，所以定义数据记录格式的模板 ID 是从 256 开始。
Field Count	该值给出了当前模板记录的域的数目。因为一个 Template Flowset 可能包含多种模板记录，用户可以通过该域判断当前模板记录的结束点和下一个模板记录的开始点。
Field Type	该值描述了域的类型，其取值可以由用户定义。比如如果支持按照目的 IP 地址、协议类型、TOS 和 MPLS 标签进行统计，则这四种信息都有一个 Type 的定义。
Field Length	该值给出了上面所定义域的长度，单位是字节。对于目的 IP 地址，取值是 4，表示 4 个字节。

图 7-16 Data FlowSet 的报文格式

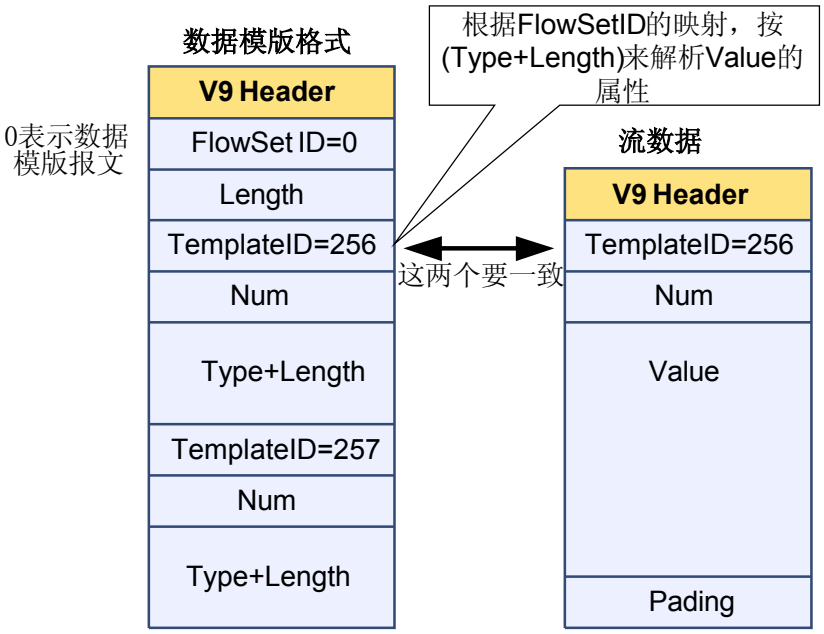


这是一个有两个记录的 Data FlowSet，Data FlowSet ID 就是包含的两个 Data Record 所套用的模板的 ID。Data FlowSet 报文中各字段的含义如下表所示。

Field Name	Value
flowset ID = template ID	该值对应于先前描述的 Template ID。采集器和显示程序通过 FlowSet ID 来对应其后面的域的类型和长度。
length	该值给出了 Data FlowSet 的长度。 Length 是用 type/length/value(TLV)格式描述的，这也意味着该值包括了 FlowSet ID 和它自己的字节长度，也包括了当前数据记录的所有长度。
record n – field n	该部分是 Data FlowSet 的域值的集合。
Padding	Padding 为 32 位长度，插入到 FlowSet 列表的最后面。需要注意的是 length 域包括该值的长度。

- 版本 9 流数据模板与流数据的关系如图 7-17 所示。

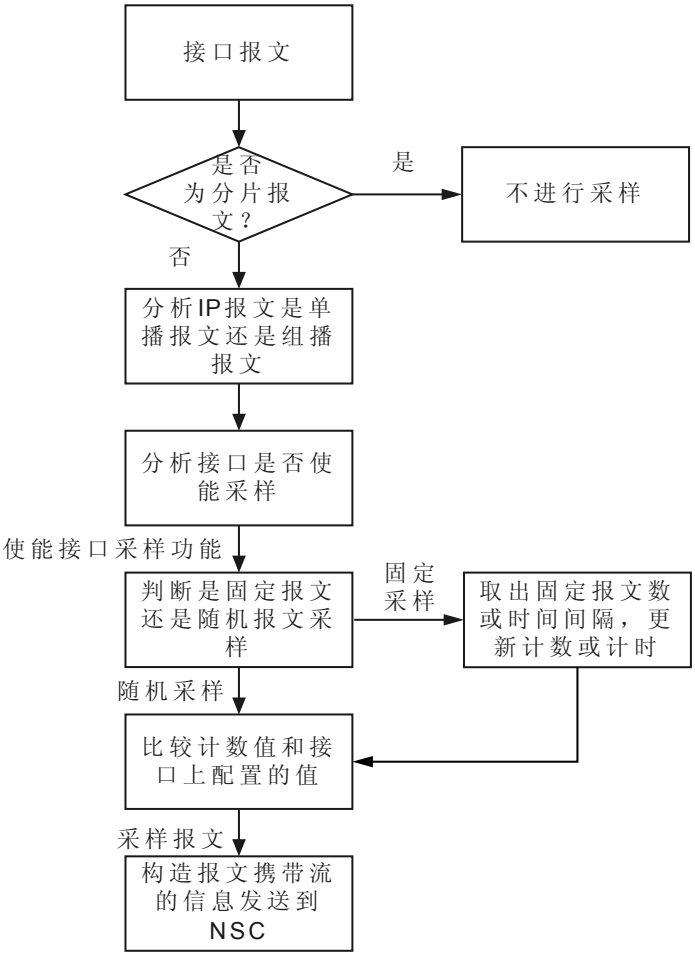
图 7-17 版本 9 流数据模板与流数据



7.4.1 采样的流程

处理的流程如图 7-18 所示：

图 7-18 NetStream 采样处理流程



接口上配置的 NetStream 相关的信息如**表 7-5** 所示。

表 7-5 在接口上配置的和 NetStream 相关的信息

名称	宽度	作用
单播报文使能标志	1bit	表示对目的 IP 地址是单播的报文是否使能
组播报文使能标志	1bit	表示对目的 IP 地址是组播的报文是否使能
采样使能标志	1bit	表示该接口是否使能采样

7.4.2 采样方式分类

使用采样的办法，可以只对少量的报文进行流信息的提取进行分析，从而减少使能 NetStream 对设备性能的影响。

采样有四种方式：

- 随机报文间隔采样：

  在此模式下，报文在配置数目间隔内被随机采样。即，如果报文间隔设置为 100，则每 100 个报文随机采样 1 个报文。

- 固定报文间隔采样：

  在此模式下，报文在配置数目间隔内被周期采样。即，如果报文间隔配置为 100，在第 5 个报文被采样后，每隔 100 个报文都会再次采样，如 105，205 和 305。

- 随机时间间隔采样：

  在此模式下，报文在配置时间间隔内被随机采样。即，如果报文间隔时间为 100，则每 100 毫秒随机采样 1 个报文。

- 固定时间间隔采样：

  在此模式下，报文在配置时间间隔内被周期采样。即，如果报文间隔配置为 100，在第 5 毫秒进行第一次采样后，每隔 100 毫秒都会再次采样，如 105 毫秒，205 毫秒和 305 毫秒。

7.4.3 流的老化

由于网络上的流是短时间阵发的，在几秒时间就会产生数万条流，而 NDE 的内存的容量是一定的，这样就需要把当前的部分流删除，为后面到来的流提供内存空间，这个过程称为老化。

流老化的分类：

- 定时老化
  - 活跃时间老化

    活跃时间是指从第一个报文到达时间到当前时间的时间间隔。当缓存区中的流超过此间隔时间后，等待新的流到来时，系统对缓存区的流进行老化。活跃老化时间主要用于持续时间较长的流量，定期输出统计数据。

  - 非活跃时间老化

    非活跃时间是指从最后一个报文到达时间与当前时间的时间间隔，在超过此间隔时间后，系统立即对流进行老化。非活跃老化时间主用于短时流量，流量停止则立即输出统计数据，节省内存空间。

- 由 TCP 连接的 FIN 和 RST 报文触发老化

  对于 TCP 连接，当有标志为 FIN 或 RST 的报文发送时，表示一次会话结束。因此当一条已经存在的 TCP 协议 NetStream 流中流过一条标志为 FIN 或 RST 的报文时，可以立即把相应的 NetStream 流老化掉。

- 计数溢出老化

  当进入流缓冲区中的流的数量，超过了系统默认的流缓存区中容纳的流的最大数量或超过了配置的流缓存区中最大流的数量时，系统就会自动对流进行老化。

7.4.4 流的输出

原始流输出方式

老化后的 NetStream 流中信息被系统统计后，以 UDP 报文形式送入到 NSC，NSC 可以得到流的详细信息，可以对这些流记录进行更为灵活的后续处理。但这样增加了网络带宽和路由器的 CPU 占有率，而且为了存储这些信息，需要占用大量的存储容量，增加了设备的开销。

聚合流输出方式

老化后的 NetStream 流中信息被系统统计后，原始信息按照一定的规则进行分类、合并后生成聚合的信息，在系统中的定时器到时后，聚合流通过 UDP 报文发送出去。通过对原始流的聚合，可以明显减少网络带宽、CPU 占用率和存储介质空间的占用。支持如表 7-6 所示的聚合方式。

表 7-6 聚合方式列表

聚合方式	说明
as	自治系统聚合，根据 NetStream 流的源、目的自治系统号，输入接口索引，输出接口索引，4 个关键值进行分类，具有相同关键值的流合并成一条聚合的流，并对应一条聚合信息。
as-tos	自治系统-ToS 聚合，根据 NetStream 流的源、目的自治系统号，输入接口索引，输出接口索引，ToS，5 个关键值进行分类，具有相同关键值的流合并成一条聚合的流，并对应一条聚合信息。
protocol-port	协议-端口聚合，根据 NetStream 流的协议号，源端口，目的端口，3 个关键值进行分类，具有相同关键值的流合并成一条聚合流，并对应一条聚合记录。
protocol-port-tos	协议-端口-ToS 聚合，根据 NetStream 流的协议号，源端口，目的端口，ToS，输入接口索引，输出接口索引，6 个关键值进行分类，具有相同关键值的流合并成一条聚合流，并对应一条聚合记录。
source-prefix	源前缀聚合，根据 NetStream 流的源自治系统号，源掩码长度，源前缀，输入接口索引，4 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
source-prefix-tos	源前缀-ToS 聚合，根据 NetStream 流的源自治系统号，源掩码长度，源前缀，ToS，输入接口索引，5 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
destination-prefix	目的前缀聚合，根据 NetStream 流的目的自治系统号，目的掩码长度，目的前缀，输出接口索引，4 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
destination-prefix-tos	目的前缀-Tos 聚合，根据 NetStream 流的目的自治系统号，目的掩码长度，目的前缀，ToS，输出接口索引，5 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。
prefix	前缀聚合，根据 NetStream 流的源、目的自治系统号，源、目的掩码长度，源、目的前缀，输入接口索引，输出接口索引，8 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。

聚合方式	说明
prefix-tos	前缀-Tos 聚合，根据 NetStream 流的源、目的自治系统号，源、目的掩码长度，源、目的前缀，ToS，输入接口索引，输出接口索引，9 个关键值进行分类，具有相同的关键值的流合并成一条聚合的流，并对应一条聚合记录。

7.4.5 Netstream 报文版本

NetStream 输出的报文主要有 5、8、9 三个版本，其他的版本处于实验阶段，没有商用。所有的版本都是通过 UDP 协议传递统计信息的。每个数据包都包括一个 Packet Header 再加上一条或者几条流的记录信息。

NetStream 原始流输出报文支持版本 5 和版本 9 两种报文格式，聚合流输出支持版本 8 和版本 9 两种报文格式。

版本 9 和以前的版本有很大的不同，它是基于模板方式的，使统计信息的输出更为灵活，而且更容易扩展新定义流的元素以及生成新的记录。版本 9 和版本 5、版本 8 不兼容。

NetStream 报文如图 7-19 所示。

图 7-19 NetStream 报文

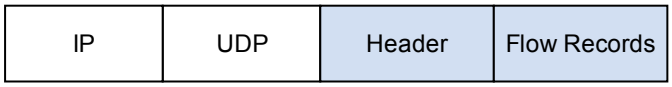
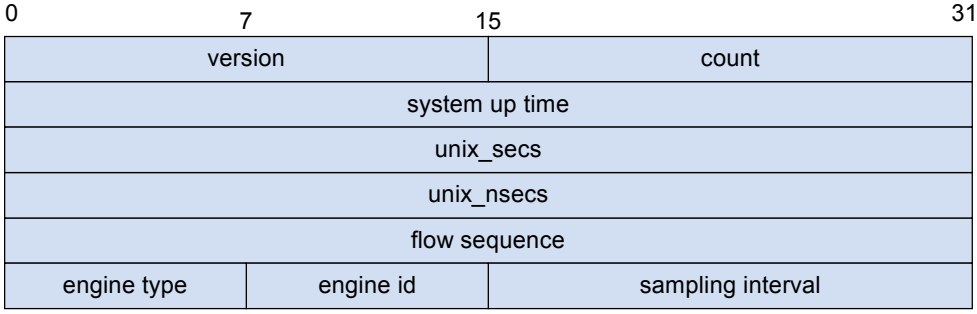


图 7-20 NetStream 版本 5 报文头



版本 5 数据报报头各字段的说明如表 7-7 下表所示。

表 7-7 版本 5 报头字段说明

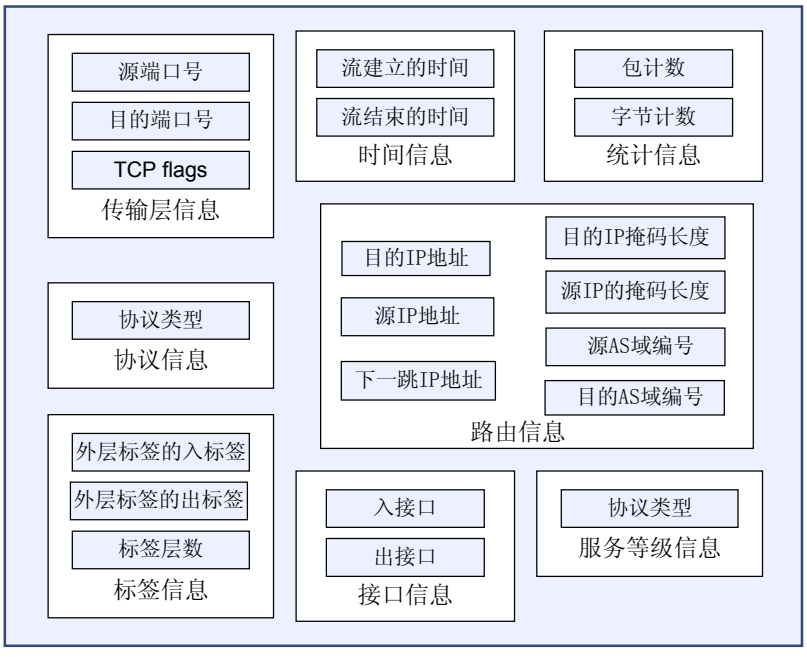
字段	描述
version	NetStream 输出报文格式版本编号，对于 V5，为 0x05

字段	描述
count	当前报文中的流记录数（1 ～ 30）
SysUptime	报文产生的时间，是系统启动以来的毫秒数
unix_secs	从 1970 年 1 月 1 日 0 时起，到报文产生时间的整秒数
unix_nsecs	报文产生时间的纳秒数，也即不足一秒的余下的纳秒数
flow_sequence	输出的流记录的顺序号， 在第一个 NetStream 报文中，此值为 0，count = c1， 在第二个 NetStream 报文中，此值为 c1，count = c2， 在第三个 NetStream 报文中，此值为 c2 + c1， …… 在第 n － 1 个 NetStream 报文中，此值为 fs(n - 1)，count = c(n - 1) 在第 n 个 NetStream 报文中，此值为 fs(n - 1) + c(n - 1)。 利用此值可以判断报文是否丢失 当流序列号溢出时，按自然溢出继续进行。
engine_type	流交换引擎类型。其中填写的是设备类型。
engine_id	交换引擎槽号。其中填写的是 NetStream 板的槽号。
sampling Interval	采样间隔，存放的是全局采样值。

版本 5 的报文包含的信息

版本 5 包括以下信息如图 7-21 深色部分所示。

图 7-21 版本 5 信息

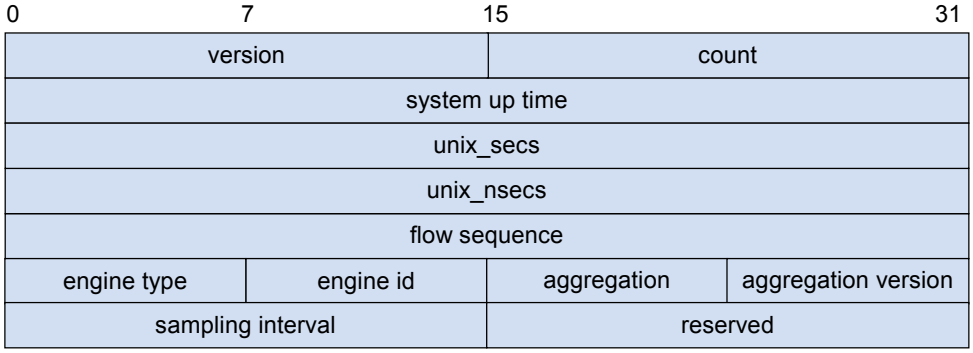


说明

对于 NetStream 出统计和入统计，分别生成各自独立的 UDP 版本 5 报文，两种报文格式一样，但是带有区别出统计和入统计的标志位。

版本 8 的报文头

图 7-22 Netream v8 报文头



版本 8 的报头格式信息如下表所示。

表 7-8 版本 8 报头字段说明

字段	描述
version	NetStream 输出报文格式版本编号，对于 V8，为 0x08。
count	当前报文中的流记录数，而不是流的总数
SysUptime	报文产生的时间，是系统启动以来的毫秒数
unix_secs	从 1970 年 1 月 1 日 0 时起，到报文产生时间的整秒数
unix_nsecs	报文产生时间的纳秒数，也即不足一秒的余下的纳秒数
flow sequence	输出的流记录的顺序号， 在第一个 NetStream 报文中，此值为 0，Count=c1, 在第二个 NetStream 报文中，此值为 c1, Count=c2, 在第一个 NetStream 报文中，此值为 c2+c1， ... 在第 n-1 个 NetStream 报文中，此值为 fs(n-1)，Count=c(n-1) 在第 n 个 NetStream 报文中，此值为 fs(n-1) + c(n-1)。 利用此值可以判断报文是否丢失 当流序列号溢出时，按自然溢出继续进行。
engine type	流交换引擎类型。其中填写的是设备类型。
engine id	交换引擎槽号。其中填写的是 NetStream 板的槽号。

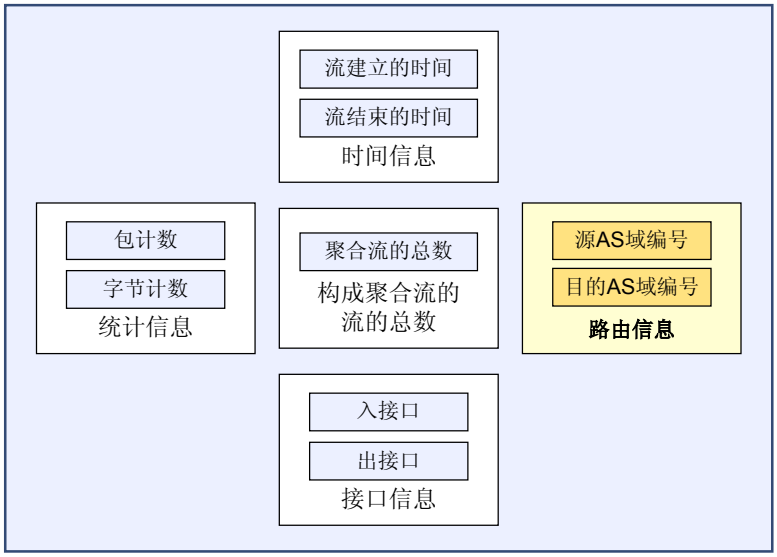
字段	描述
aggregation	聚合类型，分别如下： 01 as 02 protocol-port 03 source-prefix 04 destination-prefix 05 prefix 09 as-tos 0a protocol-port-tos 0b source-prefix-tos 0c destination-prefix-tos 0d prefix-tos
aggregation version	输出的聚合版本号，为 0x02
sampling Interval	采样间隔，存放的是全局采样值。
reserved	保留字段，为全 0。

版本 8 的报文包含的信息

从版本 8 开始，具备了对流进行简单加工的能力---聚合。所谓聚合，就是把流按照某些信息进行归类统计。流聚合的最大好处是可以减少对网络带宽的占用。而在此之前的版本，聚合的工作是通过 NSC 来实现的。

- 根据 AS 域进行聚合

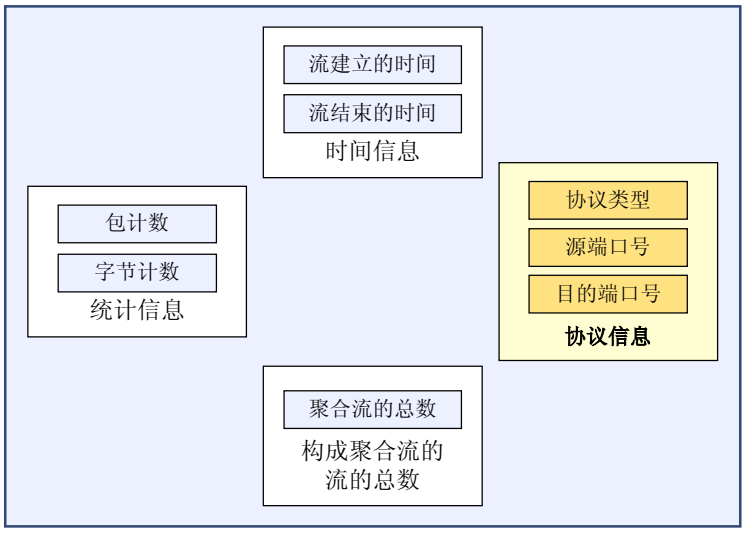
图 7-23 根据 AS 域进行聚合示意图



这种聚合方式主要用于统计在一个路由器上从一个 AS 域到另一个 AS 域的包和字节信息。运营商之间可以根据统计到的 AS 域报文的流量进行付费结算。其中的源 AS 如果为 Origin，则源 AS 编号为源地址所属的 AS；如果为 Peer，则源 AS 编号为源地址的 AS PATH 的上一个 AS。对于属于本自治域的源地址，或从路由表中不能获取 AS 号，该值为 0。对于目的 AS，如果为 Origin，则目的 AS 编号为目的地址所属的 AS；如果为 Peer，则目的 AS 编号为目的地址的 AS PATH 的上一个 AS。对于属于本自治域的目的地址，或从路由表中不能获取 AS 号，该值为 0。源 AS 编号和目的 AS 编号这两个字段是根据 AS 进行聚合的关键值。

- 根据协议类型进行聚合

图 7-24 根据协议类型聚合示意图



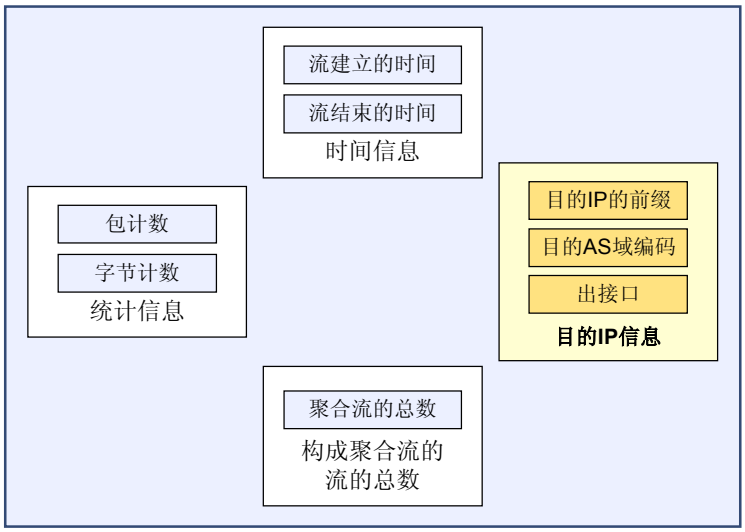
这种方式针对传输层的协议类型（对于 TCP 和 UDP，还包括源和目的端口号）进行聚合。 协议类型字段取值为 6，表示是 TCP 协议；取值是 17，表示是 UDP 协议。

对于非 TCP 或 UDP 的报文，源端口号取值为 0。 对于 ICMP，目的端口号取值是 ICMP 报文的 Type 与 Code 字段。

对于非 TCP 或 UDP 报文，目的端口号取值是 0。

- 根据目的 IP 地址的前缀进行聚合

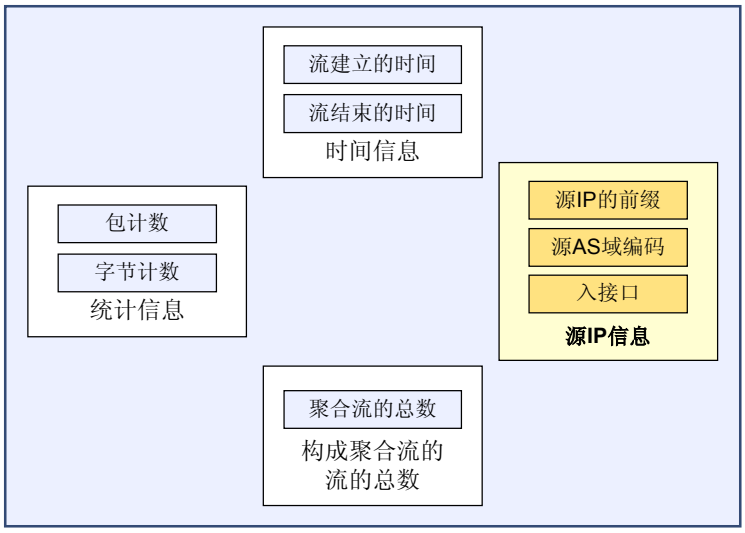
图 7-25 根据目的 IP 地址的前缀聚合示意图



这种方式针对目的 IP 地址的前缀进行聚合。

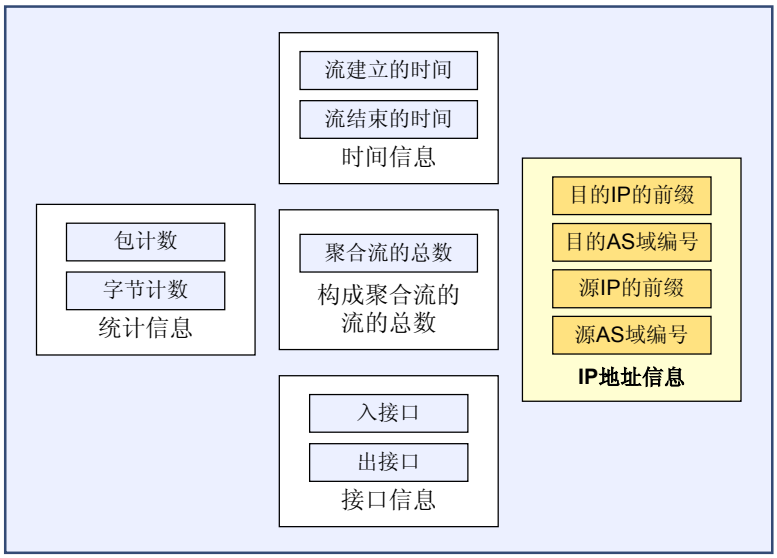
- 根据源 IP 地址的前缀进行聚合

图 7-26 根据源 IP 地址的前缀聚合示意图



- 根据源 IP 地址的前缀和目的 IP 地址的前缀进行聚合

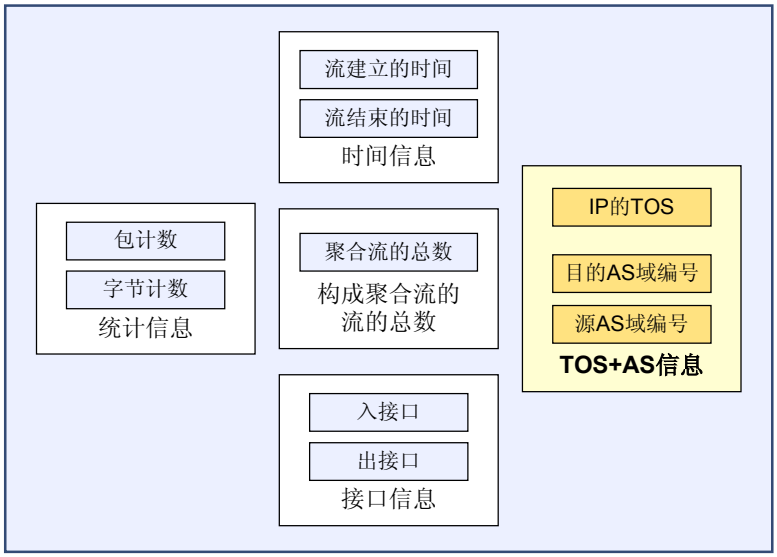
图 7-27 根据源 IP 地址的前缀和目的 IP 地址的前缀进行聚合示意图



这种方式下，源 IP 地址的前缀部分和目的 IP 地址的前缀部分都参与聚合。

- 根据 TOS ＋ AS 域进行聚合

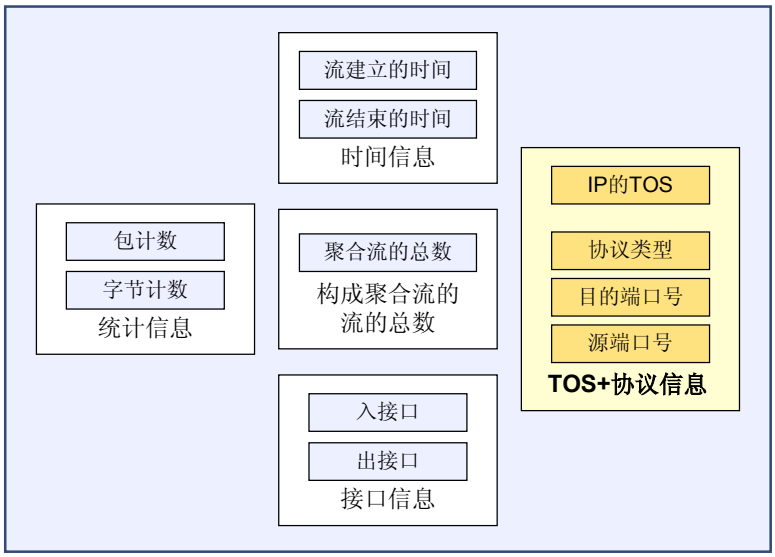
图 7-28 根据 TOS ＋ AS 域进行聚合示意图



其中 TOS（Type of Service）是 IP 报文头中的一个字段，用来设置报文的优先级。

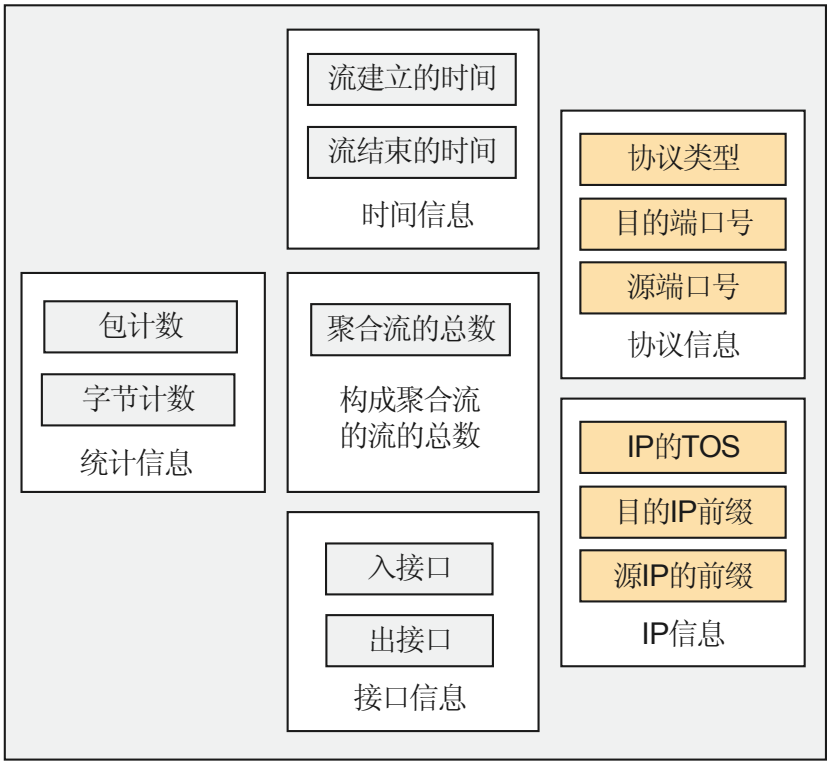
- 根据 TOS ＋协议类型进行聚合

图 7-29 根据 TOS ＋协议类型进行聚合示意图



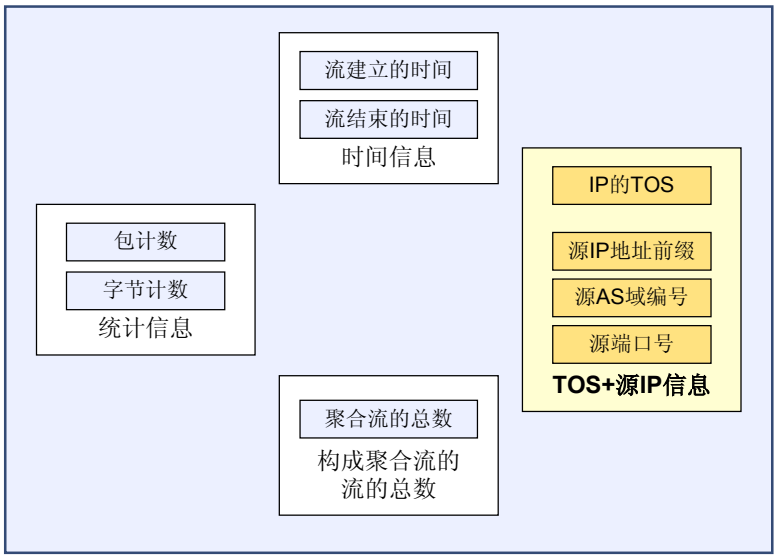
- 根据 IP 地址前缀＋ TOS ＋协议类型进行聚合

图 7-30 根据 IP 地址前缀＋ TOS ＋协议类型进行聚合



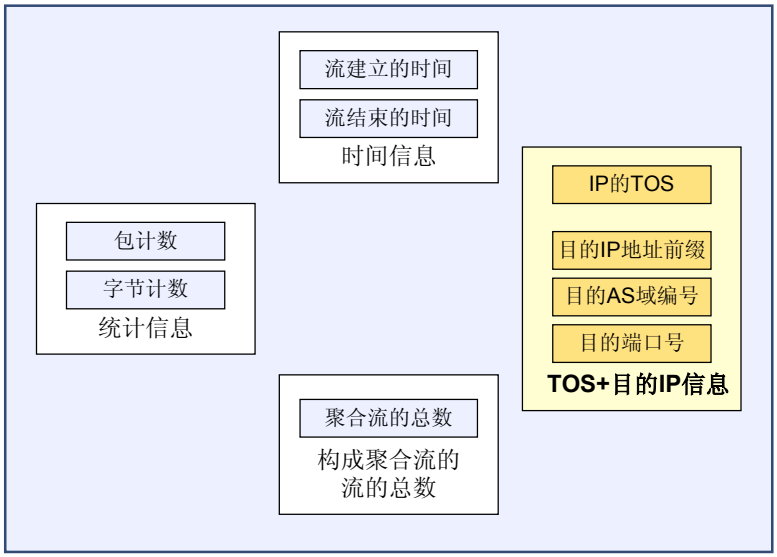
- 根据 TOS ＋源 IP 地址前缀进行聚合

图 7-31 根据 TOS ＋源 IP 地址前缀进行聚合示意图



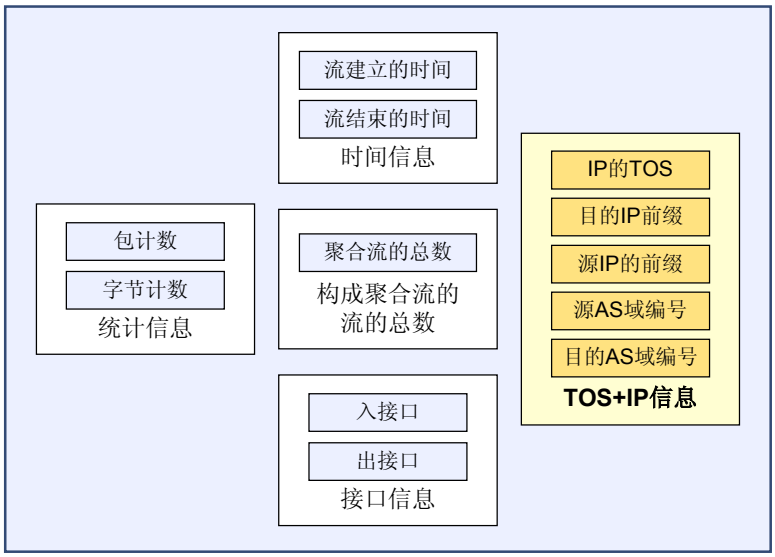
- 根据 TOS ＋目的 IP 地址前缀进行聚合

图 7-32 根据 TOS ＋目的 IP 地址前缀进行聚合



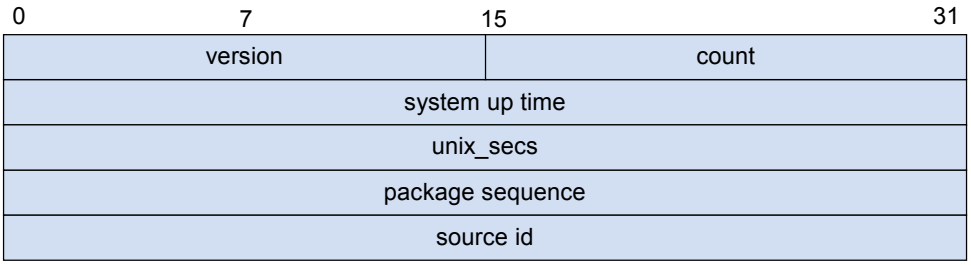
- 根据 TOS ＋ IP 地址前缀进行聚合

图 7-33 根据 TOS ＋ IP 地址前缀进行聚合示意图



版本 9 的报文头

图 7-34 版本 9 的报文头格式



报文头中各字段的含义如下表所示。

表 7-9 版本 9 报头字段说明

字段	描述
version	NetStream 输出报文格式版本编号，对于 V9，其值是 0x09。
count	该报文包含的 FlowSet records(包括 Template 和 Data)的数目。
system up time	从系统启动到报文产生的时间，单位是毫秒。
unix_secs	从 1970 年 1 月 1 日 0 时起，到报文产生时间的整秒数。

字段	描述
package sequence	所有输出报文的顺序号。该值是累积的，利用此值可以判断报文是否丢失。 **说明** 该值和 V5、V8 报文头有所不同的是：在 V5 和 V8 中该值描述的是“所有的流”。
source id	Source ID 占 4 个字节，用来保证从一台路由器中输出的所有流的唯一性(Source ID 等同于 V5、V8 报文头中的 engine type 和 engine ID)。该值可以由用户定义。Source ID 前 2 字节保留用作以后的扩展，置 0。第 3 个字节描述输出设备的路由引擎的唯一性，填写设备类型。第 4 个字节描述板卡号，填写 NetStream 板的槽号。

版本 9 的相关术语

- Template FlowSet

  表明 Export Packet 中的流信息的含义，由使能 NetStream 的路由器把它的模板放到 Export Packet 中发给 NSC，实现路由器和 NSC 之间对 Export Packet 中流信息解释建立约定。Template FlowSet 是一个集合，其中包含很多个 Template Record。Template FlowSet 是版本 9 的核心。使用模板后，NSC 的程序无需预先设置好按照什么样的格式解析 Export Packet，只需做成通用的方式，然后通过路由器发过来的模板来解释流记录的信息。模板极大的增强了 NetStream 流记录的灵活性和可扩展性，方便了第三方软件的开发，以及后续 NetStream 功能的扩展。
- Template Record

  和 Export Packet 中的各个 Data Record 对应。Data Record 中的流信息按照对应的 Template Record 来解释。
- Template ID

  用于标识不同的模板。不同的模板具有不同的 ID。在 Data Record 中包含 Template ID，根据 Template ID 来套用不同的模板。
- Data FlowSet

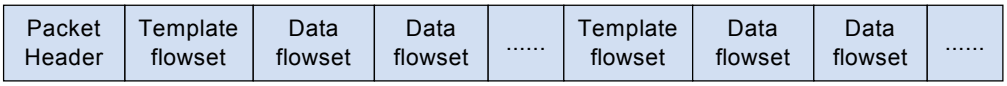
  是组合在一起的一个或者多个 Data Record 的集合。
- Data Record

  对应一个 NetStream 记录。

版本 9 的 Export Packet

版本 9 的 Export Packet，由 Packet Header、Template FlowSet 和 Data FlowSet 构成，格式的示意图如图 7-35 所示：

图 7-35 版本 9 的 Export Packet 的报文格式示意图



其中 Template FlowSet 和 Data FlowSet 是互相独立的。Data FlowSet 中的 Data Record 由 collector 已知的模板解释，也就是说，NSC 已经知道了 Data Record 中的 Template ID 对

应的模板了。而 Template FlowSet 是告诉 NSC 一个即将被使用的模板，NSC 使用这个模板的时候只能是针对后续的 Export Packet。

在 Export Packet 中不是必须要包含 Template FlowSet 和 Data FlowSet 的。可能的组合有：

Export Packet 中包含穿插着存放的 Template FlowSet 和 Data FlowSet，对于 NSC 来说，Template FlowSet 和 Data FlowSet 之间没有什么关联。NSC 将收到的 Template FlowSet 中的模板保存下来，用于后续的 Data FlowSet 的模板套用。

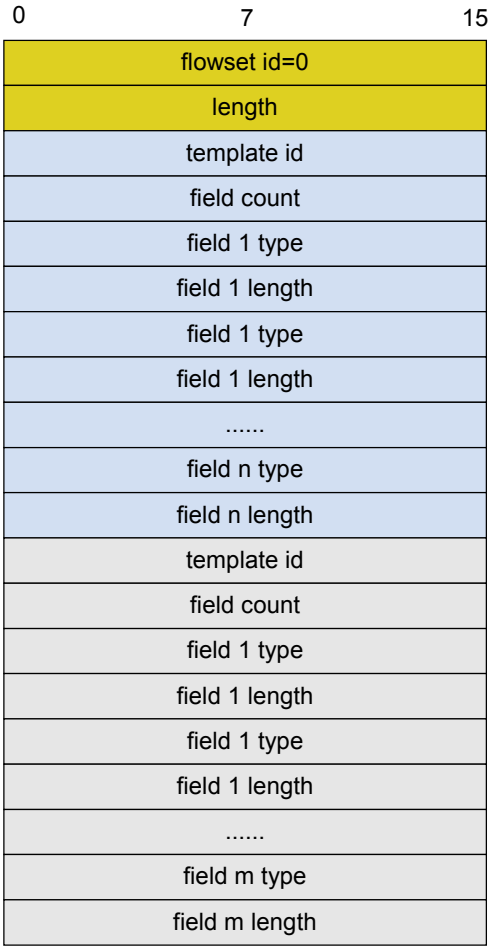
Export Packet 中仅有 Data FlowSet。如果 Template ID 都已经定义好了，使能 NetStream 的路由器传递给 NSC 的 Export Packet 一般属于这种情况。

Export Packet 中仅有 Template FlowSet，一般情况下，为了更好的利用网络带宽，Template FlowSet 都是和 Data FlowSet 打包到一个 Export Packet 中，因此这种情况是一个特例，一般出现在路由器在已经配置好模板后重启的时候，路由器需要尽快把所有的模板传递给 NSC，此时就不用等待生成统计流后 Template FlowSet 和 Data FlowSet 一同发送了。另外，模板具有有效时间，超过有效时间 NSC 会删除超时的模板，因此，需要定时的发送 Template FlowSet 到 NSC，如果需要发送的时候没有 Data FlowSet 生成，则此时只发送 Template FlowSet。

Template FlowSet 的报文格式

Template FlowSet 的格式如图 7-36 所示。

图 7-36 Template FlowSet 的格式



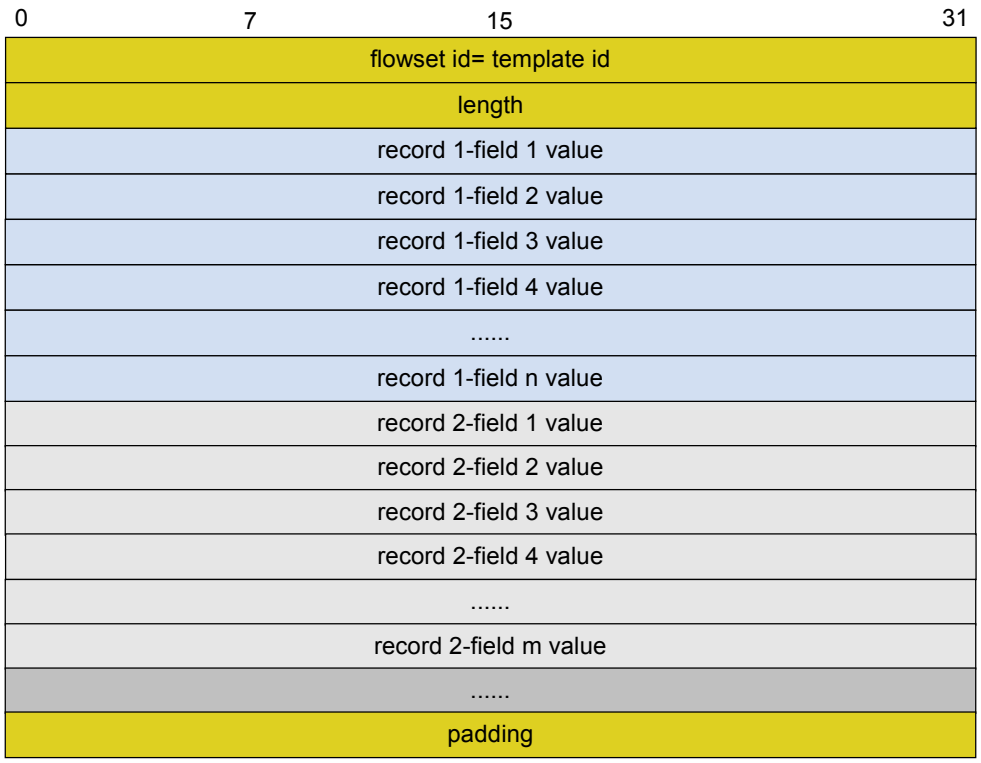
这是一个包含两个 Template Record 的例子。各字段的含义如下表所示。

字段	描述
FlowSet ID	FlowSet ID 用来区分模板记录(Template records)和数据记录(Data records)。对于 Template FlowSet，FlowSet ID 的取值是 0 ～ 255，对于 Data FlowSet，取值从 256 开始，这样 collector 就可以在 Export Packet 中识别出 Template FlowSet。
Length	Length 指的是 FlowSet 的总长度。因为一个单独的 Template FlowSet 可能包含多种模板 ID，长度值可以用来判断下一个 FlowSet 记录(可以是 Template FlowSet 或 Data FlowSet)的位置。 Length 是用 type/length/value(TLV)格式描述的，这也意味着该值包括了 FlowSet ID 和它自己的字节长度，也包括了当前 FlowSet 的所有 Template 记录的长度。
Template ID	当路由器生成不同的 Template FlowSets 来匹配将要输出的 NetStream 数据类型时，每个模板都会分配一个唯一的 ID。这个唯一性是针对生成模板 ID 的路由器的。因为 0 ～ 255 已经被 FlowSet IDs 使用，所以定义数据记录格式的模板 ID 是从 256 开始。
Field Count	该值给出了当前模板记录的域的数目。因为一个 Template Flowset 可能包含多种模板记录，用户可以通过该域判断当前模板记录的结束点和下一个模板记录的开始点。
Field Type	该值描述了域的类型，其取值可以由用户定义。比如如果支持按照目的 IP 地址、协议类型、TOS 和 MPLS 标签进行统计，则这四种信息都有一个 Type 的定义。
Field Length	该值给出了上面所定义域的长度，单位是字节。对于目的 IP 地址，取值是 4，表示 4 个字节。

Data FlowSet 的报文格式

Data FlowSet 的报文格式如图 7-37 所示。

图 7-37 Data FlowSet 的报文格式



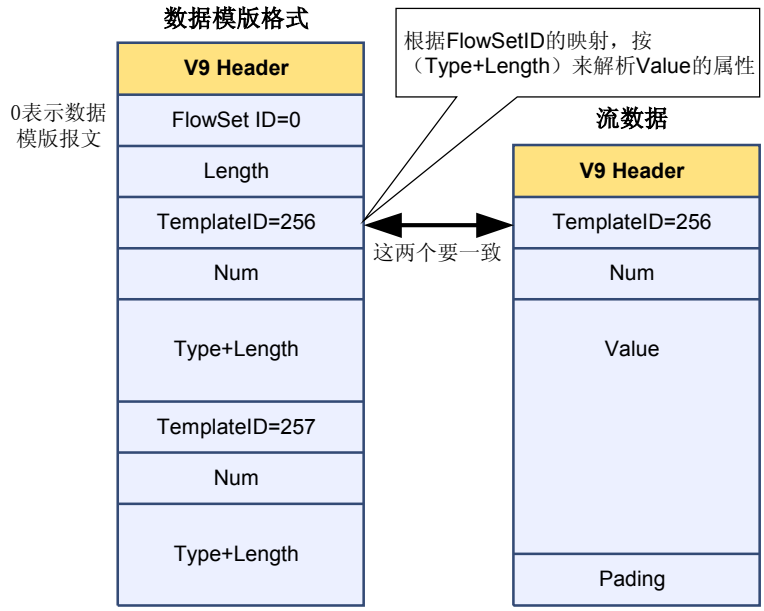
这是一个有两个记录的 Data FlowSet，Data FlowSet ID 就是包含的两个 Data Record 所套用的模板的 ID。Data FlowSet 报文中各字段的含义如下表所示。

Field Name	Value
flowset ID = template ID	该值对应于先前描述的 Template ID。采集器和显示程序通过 FlowSet ID 来对应其后面的域的类型和长度。
length	该值给出了 Data FlowSet 的长度。Length 是用 type/length/value(TLV)格式描述的，这也意味着该值包括了 FlowSet ID 和它自己的字节长度，也包括了当前数据记录的所有长度。
record n - field n	该部分是 Data FlowSet 的域值的集合。
Padding	Padding 为 32 位长度，插入到 FlowSet 列表的最后面。需要注意的是 length 域包括该值的长度。

版本 9 流数据模板与流数据的关系

版本 9 流数据模板与流数据的关系如下图所示。

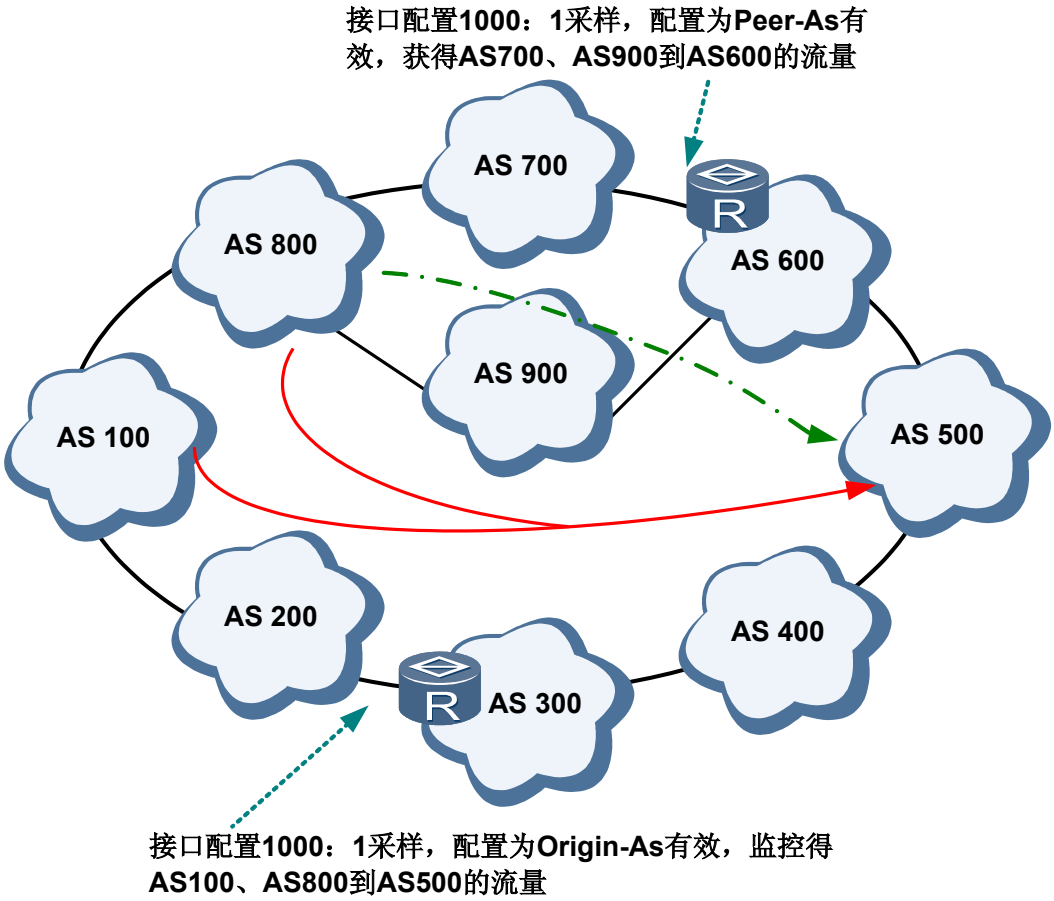
图 7-38 版本 9 流数据模板与流数据



7.5 应用

AS 域间规划

图 7-39 AS 域间规划示意图



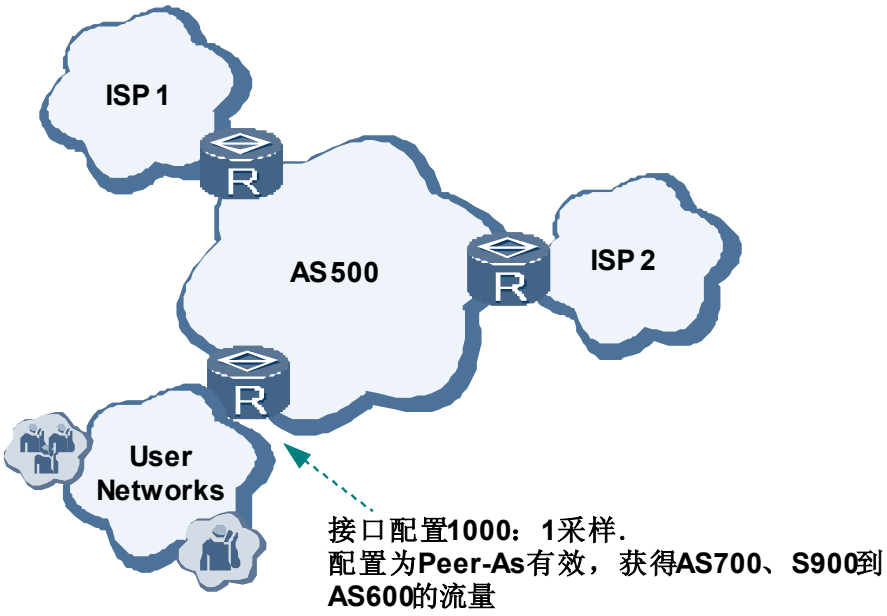
通过配置可以指定 NetStream 报文携带 BGP 路由 AS 类型：origin-as 或 peer-as，二者是互斥的。Origin-as 表示发布该路由的 AS，peer-as 表示路由通过哪个邻接的 AS 透传过来。

通过在 AS 域间部署 NetStream，可以实现对路由发布 AS 或邻接 AS 到达本地 AS 的流量进行统计和监控，为网络部署提供依据。

如上图所示：（1）流量从 AS800 经过 AS700、AS900、AS600 到达 AS500，如果在 AS600 使能 NetStream，配置为 peer-as 有效，则可以监控到 AS700 和 AS900 到 AS600 的流量；（2）流量从 AS800 和 AS100 经过 AS200、AS300、AS400 到达 AS500，如果在 AS300 使能 NetStream，配置为 origin-as 有效，则可以监控到 AS800、AS100 到 AS500 的流量。

ISP 间根据 IP 前缀按流量分帐计费

图 7-40 ISP 间根据 IP 前缀按流量分帐计费



不同的 ISP 具有不同的 IP 地址前缀，可以通过 destination-prefix 聚合，根据目的 IP 和掩码，对访问不同 ISP 的流量进行统计，从而实现分账计费。

7.6 术语与缩略语

术语

术语	解释
流	通过某个网络设备且具有某些公共属性的一系列单方向的数据包，通常使用源 IP 地址、目的 IP 地址、源端口号、目的端口号、协议类型、服务类型（ToS）、输入或输出接口来唯一地标识一个流。
Template FlowSet	表明 Export Packet 中的流信息的含义，由使能 NetStream 的 NDE 把它的模板放到 Export Packet 中发给 NSC，实现 NDE 和 NSC 之间对 Export Packet 中流信息解释建立约定。Template FlowSet 是一个集合，其中包含很多个 Template Record。Template FlowSet 是版本 9 的灵魂。使用模板后，NSC 的程序无需预先设置好按照什么样的格式解析 Export Packet，只需做成通用的方式，然后通过 NDE 发过来的模板来解释流记录的信息。模板极大的增强了 NetStream 流记录的灵活性和可扩展性，方便了第三方软件的开发，和后续 NetStream 功能的增强。
Template Record	和 Export Packet 中的各个 Data Record 对应。Data Record 中的流信息按照对应的 Template Recode 来解释。

术语	解释
Template ID	用于标识不同的模板。不同的模板具有不同的 ID。在 Data Record 中包含 Template ID，根据 Template ID 来套用不同的模板。
Data FlowSet	是组合在一起的一个或者多个 Data Record 的集合。
Data Record	对应一个 NetStream 记录。

缩略语

缩略语	英文全称	中文全称
NDE	NetStream Data Exporter	流量输出器
NSC	NetStream Collector	流量收集器
NDA	NetStream Data Analyzer	流量分析器

8 Ping 和 Tracert

关于本章

8.1 介绍

定义

Ping 命令是最常见的用于检测网络设备可访问性的调试工具，它使用 ICMP 的 echo 信息来决定：

- 远程设备是否可用。
- 与远程主机通信的来回旅程（round-trip）的延迟（delay）。
- 包（packet）的丢失情况。
- Tracert 命令用于测试数据包从发送主机到目的地所经过的网关。

目的

当设备出现故障时，可以首先使用 Ping 与 Tracert 命令测试网络连接是否正常工作。

Ping（Packet Internet Groper）命令主要用于检查网络连接及主机是否可达。源主机向目的主机发送 ICMP 请求报文，目的主机向源主机发送 ICMP 回应报文。

Tracert 主要检查网络连接是否可达以及分析网络什么地方发生了故障。

受益

通过 Ping 和 Tracert 命令可以方便探测网络的 IP 互通情况，以及定位网络故障点，方便了用户进行网络维护。

8.2 参考标准和协议

本特性的参考资料清单如下：

文档	描述	备注
RFC4379	Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures	-
RFC5085	Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires	-
RFC791	Internet Protocol	-
RFC1256	ICMP Router Discovery Messages	-
RFC1393	Traceroute Using an IP Option	-

8.3 可获得性

涉及网元

无需其它网元的配合。

License 支持

无需获得 License 许可，均可获得该特性的服务。

版本支持

产品	最低支持版本
AR3200	V200R001C00

8.4 原理描述

8.4.1 Ping 的工作过程

Ping 会发送 echo request message 到某个地址，然后等待应答（reply），当 echo request 到达目标地址以后，在一个有效的时间内（timeout 之前）返回 echo reply message 给源地址，则说明目的地可达。如在有效时间内，没有收到回应，则在发送端显示超时。

Ping 命令是把 ICMP 报文中的标示符置为发送该 ICMP 报文的进程，这样在对端可以区分出从本端运行的多个 Ping 实例。

Ping 命令每发送一个 ICMP 回显请求报文，顺序号就加 1，顺序号从 1 开始，不同的系统发送回显请求的数量不同，默认情况发送 5 个回显请求报文。也可以通过命令行参数设置发送回显请求报文的个数，如果对端可达，则在对端会相应回应 5 个和请求端同样序号的 ICMP 回应报文。

报文在转发过程中，如果 TTL 字段的值减为 0，报文到达的设备就会向源端发送 ICMP 超时报文，表明远程设备不可达。

8.4.2 Tracert 的工作过程

源端首先发送 3 个 TTL 字段的值都为 1 的 UDP 数据报给远程设备，使用随机的任何大于 32768 的端口地址作为目标设备的接受报文端口，TTL 为 1 的数据报到达某个路由器以后随即超时，路由器响应源设备一个 ICMP 的超时报文，之后源端再发送 3 个 UDP 数据报，这次更改 TTL 值为 2，即经过 2 个路由器以后，响应源端 ICMP 超时报文，依次类推，直到这些 UDP 报文到达了目标设备。

由于发送的报文中的目的端口，目标设备接收到 ICMP 报文后，由于报文的端口是一个在目标设备没有应用的端口，目标设备就会响应 ICMP port unreachable 信息给源端，表示目标端口不可达，同时说明 Tracert 执行完毕。从而可以从源端显示的结果中，看到到目标设备所经过的路径。

Tracert 发送数据报的 TTL 值最大可以到 30，每一次发送如果在指定的时间的内没有回应报文，在发送端就会显示超时，如果发送 30 跳的值后，仍然显示为超时，表明无法

达到目标设备，测试失败。默认情况没有发送报文的超时时间为 5 秒，可以在 0ms ～ 65535ms 之间进行设置。

8.4.3 LSPV

LSPV（Label Switched Path Verification）是通过 MPLS Ping/Tracert 为用户提供发现 LSP 错误，并及时定位失效节点的机制。

MPLS 隧道技术支持多种高层协议与业务。在 MPLS 中，负责建立 LSP 的 MPLS 控制平面将无法检测到 LSP 转发数据失败，这会给网络维护带来困难。类似于普通 IP 的 Ping/Traceroute，MPLS Ping/Tracert 用于检测 LSP 的可用性。

MPLS Ping/Tracert 使用 MPLS Echo Request 和 MPLS Echo Reply 这两种消息来检测 LSP 的连通性。报文以 UDP 报文格式发送，端口号为 3503。接收端通过 UDP 端口号识别出 MPLS Echo Request 和 MPLS Echo Reply 报文。MPLS Echo Request 中携带需要检测的 FEC 信息，和其他属于此 FEC 的报文一样沿 LSP 发送，从而实现对 LSP 的检测。MPLS Echo Request 通过 MPLS 转发给目的端，而 MPLS Echo Reply 则通过 IP 转发给源端。为了防止消息到达 Egress 节点后又被转发给其他节点，Echo Request 消息的 IP 头中目的地址设置为 127.0.0.1/8（本机环回地址），IP 头中的 TTL 值＝ 1。

支持如下两种链路类型的诊断。

- 支持 P2P LDP LSP Ping&Tracert
- 支持 L3VPN LSP Ping

P2P LDP LSP Ping&Traceroute

根据用户指定的 FEC、掩码与 NextHop，在 Ingress 上发起到 Egress 的 Ping/Tracert，以达到检测 LSP 隧道连通性的目的。

说明

若存在多条 LDP LSP，则通过指定 NextHop 地址来检测选定 LSP 的连通性。

L3VPN LSP Ping

L3VPN LSP Ping 检测可以实现在本端 PE 上发起到对端 PE 的 Ping，以达到检测通过 BGP 建立的私网 LSP 隧道的连通性目的。

8.4.4 Ping 检测 Trunk 成员口

Ping 检测 Trunk 成员口的应用背景

当设备出现故障时，可以首先使用 ping 与 tracert 命令测试网络连接是否正常工作。Ping（Packet Internet Groper）命令主要用于检查网络连接及主机是否可达。源主机向目的主机发送 ICMP 请求报文，目的主机向源主机发送 ICMP 回应报文。

在进行 Ping 检测时，如果出接口是 Trunk 主接口时，由于 Trunk 的每一个成员端口经过的传输路径都不一样，所以每一个端口承载业务的相应时延、抖动、丢报率等都可能不相同。所以如果当一个 TRUNK 承载的业务质量大幅下降的时候，无法准确判断是 Trunk 中的哪一个成员端口出了问题，也无法进行准确测试，没有测试手段。

通过 Ping 检测 Trunk 成员口，实现对某一实际物理链路的检测。方便用户精确定位。Ping 检测 Trunk 成员口的相关规格如下：

- 用于检测直连链路
- 用于检测三层 Trunk 口，Trunk 子接口
- 检测的目的地址需为对端 Trunk 接口的地址，不能为 Loopback 口或其他接口地址

Ping 检测 Trunk 成员口的基本原理

要实现对某一个指定的 Trunk 成员口端到端的链路状态进行检测，需要 request 报文和 reply 报文走同一条成员口链路，reply 报文通过 request 报文的入接口发送。

- 发起端

  Ping 检测命令行指定成员接口。根据指定的出接口索引，从 Trunk 模块获取此成员口对应的 Trunk 逻辑口接口索引，判断此接口是否为 Trunk 的成员口。并判断此 Trunk 逻辑口是否 UP。若逻辑口 UP，发起检测。报文通过指定出接口发送，不再由 Trunk 模块进行 Hash 选口。

- 接收端

  在接收端配置全局命令行，使能该命令后，将使能 Trunk 成员口回包走指定出接口的状态下发到各个接口板。目的节点接收到 request 报文后，记录 request 报文的实际入接口索引。Reply 报文设置指定出接口发送方式，reply 报文的出接口如果是 Trunk 接口，则 reply 报文从 request 报文的入接口发送。

8.5 术语与缩略语

缩略语

缩略语	英文全称	中文全称
MPLS	Multiprotocol Label Switch	多协议标签交换
LSP	Label Switched Path	标签交换路径
LDP	Label Distribution Protocol	标签发布协议
L3VPN	Layer3 Virtual Private Network	三层虚拟专用网
PE	Provider Edge	服务提供商边缘设备