目　录

插图目录

表格目录

4 SCTP 协议

关于本章

本章描述内容如下表所示。

标题	内容
4.1 概述	概述 SCTP 协议制定的背景
4.2 SCTP 相关术语	介绍 SCTP 的相关术语。
4.3 SCTP 功能	介绍 SCTP 的功能。
4.4 SCTP 原语	介绍 SCTP 的原语。
4.5 SCTP 协议消息	介绍 SCTP 的协议消息结构、数据块格式等。
4.6 SCTP 基本信令流程	介绍 SCTP 的基本信令流程。

4.1 概述

在流控制传输协议 SCTP（Stream Control Transmission Protocol）制定以前，在 IP 网上传输七号信令使用的是 UDP 和 TCP 协议。UDP 是一种无连接的传输协议，无法满足七号信令对传输质量的要求。TCP 协议是一种有连接的传输协议，可以信令的可靠传输，但是 TCP 协议具有行头阻塞、实时性差、支持多归属比较困难、易受拒绝服务攻击（Dos）的缺陷。因此 IETF（Internet Engineering Task Force）RFC2960 制定了面向连接的基于分组的可靠传输协议 SCTP 协议。SCTP 对 TCP 的缺陷进行了完善，使得信令传输具有更高的可靠性，SCTP 的设计包括适当的拥塞控制、防止泛滥和伪装攻击、更优的实时性能和多归属性支持，因此，SCTP 成为 SIGTRAN 协议族中的传输协议。

SCTP 被视为一个传输层协议，它的上层为 SCTP 用户应用，下层作为分组网络。在 SIGTRAN 协议的应用中，SCTP 上层用户是 SCN 信令的适配模块（如 M2UA、M3UA），下层是 IP 网。

4.2 SCTP 相关术语

传送地址

传送地址由 IP 地址、传输层协议类型和传输层端口号定义。由于 SCTP 在 IP 上传输，所以一个 SCTP 传送地址由一个 IP 地址加一个 SCTP 端口号决定。SCTP 端口号就是 SCTP 用来识别同一地址上的用户，和 TCP 端口号是一个概念。比如 IP 地址 10.105.28.92 和 SCTP 端口号 1024 标识了一个传送地址，而 10.105.28.93 和 1024 则标识了另外一个传送地址，同样，10.105.28.92 和端口号 1023 也标识了一个不同的传送地址。

主机和端点

- 主机（Host）

主机配有一个或多个 IP 地址，是一个典型的物理实体。

- 端点（SCTP Endpoint）

端点是 SCTP 的基本逻辑概念，是数据报的逻辑发送者和接收者，是一个典型的逻辑实体。

一个传送地址（IP 地址＋SCTP 端口号）唯一标识一个端点。一个端点可以由多个传送地址进行定义，但对于同一个目的端点而言，这些传送地址中的 IP 地址可以配置成多个，但必须使用相同的 SCTP 端口。

说明
一个主机上可以有多个端点。

偶联和流

- 偶联（Association）

偶联就是两个 SCTP 端点通过 SCTP 协议规定的 4 步握手机制建立起来的进行数据传递的逻辑联系或者通道。

SCTP 协议规定在任何时刻两个端点之间能且仅能建立一个偶联。由于偶联由两个端点的传送地址来定义，所以通过数据配置本地 IP 地址、本地 SCTP 端口号、对端 IP 地址、对端 SCTP 端口号等四个参数，可以唯一标识一个 SCTP 偶联。正因为如此，在 GTSOFTX3000 中，偶联可以被看成是一条 M2UA 链路或 M3UA 链路。

- 流（Stream）

流是 SCTP 协议的一个特色术语。SCTP 偶联中的流用来指示需要按顺序递交到高层协议的用户消息的序列，在同一个流中的消息需要按照其顺序进行递交。严格地说，“流”就是一个 SCTP 偶联中，从一个端点到另一个端点的单向逻辑通道。一个偶联是由多个单向的流组成的。各个流之间相对独立，使用流 ID 进行标识，每个流可以单独发送数据而不受其他流的影响。

说明

- 一个偶联中可以包含多个流，可用流的数量是在建立偶联时由双方端点协商决定，而一个流只能属于一个偶联。同时，出局的流数量可以与入局流数量的取值不同。
- 顺序提交的数据必须在一个流里面传输。

通路（Path）和首选通路（Primary Path）

- 通路（Path）

通路是一个端点将 SCTP 分组发送到对端端点特定目的传送地址的路由。如果分组发送到对端端点不同的目的传送地址时，不需要配置单独的通路。

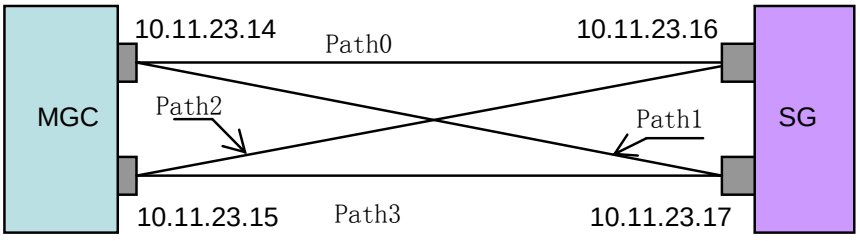
- 首选通路（Primary Path）

首选通路是在默认情况下，目的地址、源地址在 SCTP 分组中发到对端端点的通路。如果可以使用多个目的地址作为到一个端点的目的地址，则这个 SCTP 端点为多归属。如果发出 SCTP 分组的端点属于多归属节点时，如果定义了目的地址、源地址，能够更好控制响应数据块返回的通路和数据包被发送的接口。

一个 SCTP 偶联的两个 SCTP 端点都可以配置多个 IP 地址，这样一个偶联的两个端点之间具有多条通路，这就是 SCTP 偶联的多地址性。SCTP 偶联的多地址性是 SCTP 与 TCP 最大的不同。

一个偶联可以包括多条通路，但只有一个首选通路。如图 4-1 所示，MGC（如 GTSOFTX3000）一个端点包括两个传送地址（10.11.23.14：2905 和 10.11.23.15：2905），而 SG 一个端点也包括两个传送地址（10.11.23.16：2904 和 10.11.23.17：2904）。

图4-1 SCTP 双归属



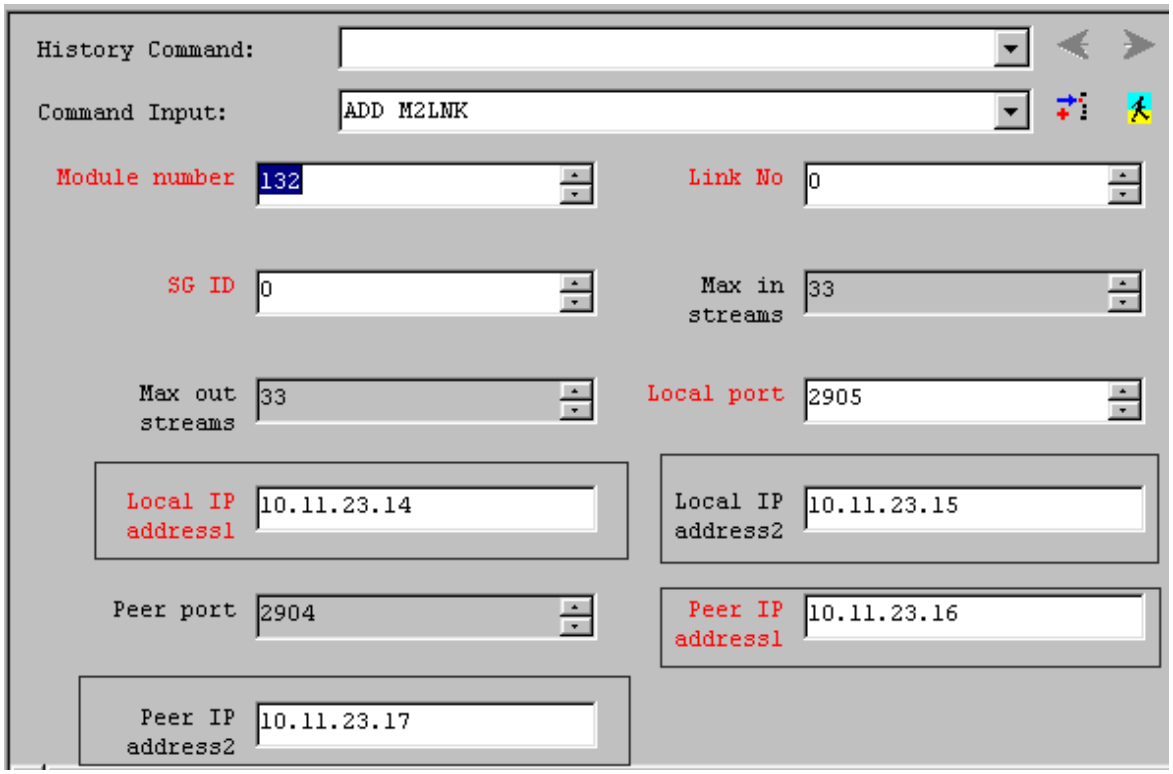
此两个端点决定了一个偶联，该偶联包括 4 条通路（Path0、Path1、Path2 和 Path3）。根据数据配置可以确定此 4 条通路的选择方式，如图 4-2 所示。图中定义了 4 条通路，而且首选通路为 Path0：

- Path0：本端传送地址 1（10.11.23.14：2905）发送 SCTP 分组到对端传送地址 1（10.11.23.16：2904）。
- Path1：本端传送地址 1（10.11.23.14：2905）发送 SCTP 分组到对端传送地址 2（10.11.23.17：2904）。
- Path2：本端传送地址 2（10.11.23.15：2905）发送 SCTP 分组到对端传送地址 1（10.11.23.16：2904）。
- Path3：本端传送地址 2（10.11.23.15：2905）发送 SCTP 分组到对端传送地址 2（10.11.23.17：2904）。

端点发送的 SCTP 工作原理为：本端点传送地址 A 发送的 SCTP 包通过首选通路发送到对端端点。当首选通路出现故障后，SCTP 可以自动切换到其他备用通路上，优先切换对端端点的传送地址，再次切换本端端点的传送地址。

SCTP 定义了心跳消息（Heart Beat）。当某条通路空闲时，本端 SCTP 用户要求 SCTP 生成相应的心跳消息并通过该通路发送到对端端点，而对端端点必须立即发回对应的心跳确认消息。这种机制被用来精确测量回路时延 RTT（Round Trip Time），而且可以随时监视偶联的可用情况和保持 SCTP 偶联的激活状态。

图4-2 确定通路选择方式的数据配置



TSN 和 SSN

- 传输顺序号 TSN（Transmission Sequence Number）

SCTP 使用 TSN 机制实现数据的确认传输。一个偶联的一端为本端发送的每个数据块顺序分配一个基于初始 TSN 的 32 位顺序号，以便对端收到时进行确认。

TSN 是基于偶联进行维护的。

说明
在 TCP 协议中，数据的确认传输和顺序递交是通过 TSN 这一种机制实现的。当发现 TSN 不连续时候，TCP 将进行数据重传，直到 TSN 连续以后才将数据向 TCP 层的上层用户递交。这实现机制导致 TCP 协议不能满足七号信令对于低传输时延的要求。

- 流顺序号 SSN（Stream Sequence Number）

SCTP 为本端在这个流中发送的每个数据块顺序分配一个 16 位 SSN，以便保证流内的顺序传递。

在偶联建立时，所有流中的 SSN 都是从 0 开始。当 SSN 到达 65535 后，则接下来的 SSN 为 0。

TSN 和 SSN 的分配是相互独立的。

拥塞窗口 CWND（Congestion Window）

SCTP 也是一个滑动窗口协议，拥塞窗口是针对每个目的地址维护的，它会根据网络状况调节。当目的地址的发送未证实消息长度超过其 CWND 时，端点将停止向这个地址发送数据。

接收窗口 RWND（Receive Window）

RWND 用来描述一个偶联对端的接收缓冲区大小。偶联建立过程中，双方会交换彼此的初始 RWND。RWND 会根据数据发送、证实的情况即时地变化。RWND 的大小限制了 SCTP 可以发送的数据的大小。当 RWND 等于 0 时，SCTP 还可以发送一个数据报，以便通过证实消息得知对方缓冲区的变化，直到达到 CWND 的限制。

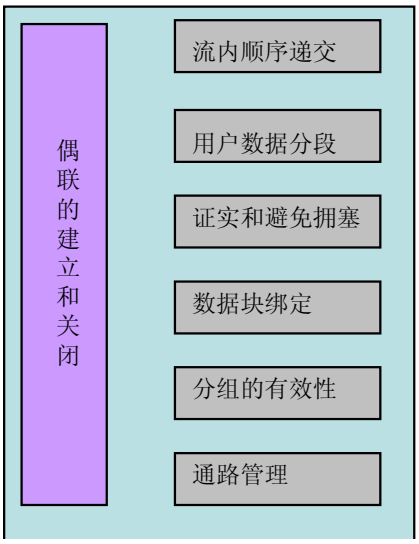
传输控制块 TCB（Transmission Control Block）

TCB 是一种内部数据结构，是一个 SCTP 端点为它与其他端点之间已经启动的每一个偶联生成的。TCB 包括端点的所有状态、操作信息，便于维护和管理相应的偶联。

4.3 SCTP 功能

如图 4-3 所示，SCTP 的功能主要包括：
偶联的建立和关闭、流内消息顺序递交、用户数据分段、证实和避免拥塞、消息块绑定、分组的有效性和通路管理。

图4-3 SCTP 功能示意图



4.3.1 偶联的建立和关闭

偶联的建立是由 SCTP 用户（如 M2UA、M3UA 等）发起请求来启动的。而且建立过程相对于 TCP 连接而言比较复杂，是个“四次握手”过程，并用到了“COOKIE”的机制。COOKIE 是一个含有端点初始信息和加密信息的数据块，通信的双方在关联建立时需要处理并交换，从而增加协议的安全性，防止拒绝服务和伪装等潜在的攻击。

SCTP 提供了对激活偶联的正常的关闭程序，它必须根据 SCTP 用户的请求来执行，当然 SCTP 也提供一种非正常（即中止）程序，中止程序的执行既可以根据 SCTP 用户的请求来启动，也可以由 SCTP 协议检查出差错来中止。

SCTP 不支持半打开状态（即一端可以在另一端结束后继续发送数据）。无论是哪个端点执行了关闭程序，偶联的两端都应停止接受从 SCTP 用户发来请求原语。

4.3.2 流内消息顺序递交

SCTP 提供数据报的顺序传递，顺序传递的数据报必须放在一个“流”中传递。流是顺序传递的基石。

通过流，SCTP 将数据的确认和传输的有序递交分成两种不同机制。SCTP 使用 TSN 机制实现了数据的确认传输，使用流号和 SSN（流顺序号）则实现数据的有序递交。当 SCTP 收到数据的 SSN 连续的时候，SCTP 就可以将数据向 SCTP 用户递交，而不用等到数据的 TSN 号连续以后才向 SCTP 用户递交。

当一个流被闭塞时，期望的下一个连续的 SCTP 用户消息可以从另外的流上进行递交。

SCTP 也提供非顺序递交的业务，接收到的用户消息可以使用这种方式立即递交到 SCTP 用户，而不需要保证其接收顺序。

4.3.3 用户数据分段

SCTP 通过对传送通路上最大 PMTU（Path Maximum Transmission Unit）的检测，实现在 SCTP 层将超大用户数据分片打包，避免在 IP 层的多次分片、重组，可以减少 IP 层的数据负担。

- 在发送端，SCTP 可以对大的用户数据报进行分片以确保 SCTP 数据报传递到低层时适合通路 MTU（Maximum Transmission Unit）。
- 在接收端，SCTP 将分片重组为完整的用户数据报，然后传递给 SCTP 用户。

4.3.4 证实和避免拥塞

证实和重传是协议保证传输可靠性的策略，SCTP 也一样。证实机制是 SCTP 保证传输可靠性的基石。避免拥塞沿袭了 TCP 的窗口机制，进行合适的流量控制。

- SCTP 在将数据（数据分片或未分片的用户数据报）发送给底层之前顺序地为之分配一个发送顺序号（TSN）。
- TSN 和 SSN（流顺序号）是相互独立的，TSN 用于保证传输的可靠性，SSN 用于保证流内消息的顺序传递。
- TSN 和 SSN 在功能上使可靠传递和顺序传递分开。接收端证实所有收到的 TSNs，即使其中有些尚未收到。
- 包重发功能负责 TSN 的证实，还负责拥塞消除。

4.3.5 消息块绑定

如果长度很短的用户数据被带上很大一个 SCTP 消息头，其传递效率会很低，因此，SCTP 将几个用户数据绑定在一个 SCTP 报文里面传输，以提高带宽的利用率。

- SCTP 分组由公共分组头和一个/多个信息块组成，信息块可以是用户数据，也可以是 SCTP 控制信息。
- SCTP 用户能够可选地使用捆绑功能，决定是否将多个用户数据报捆绑在一个 SCTP 分组中。
- 为提高效率，拥塞/重发时，捆绑功能可能仍被执行，即使用户已经禁止捆绑。

4.3.6 分组的有效性

分组的有效性是 SCTP 提供无差错传输的基石。SCTP 分组的公共分组头包含一个验证标签（Verification Tag）和一个可选的 32 位校验码（Checksum）。

验证标签的值由偶联两端在偶联启动时选择。如果收到的分组中如果没有期望的验证标签值，接收端将丢弃这个分组，以阻止攻击和失效的 SCTP 分组。

校验码由 SCTP 分组的发送方设置，以提供附加的保护，用来避免由网络造成的数据差错。接收端将丢弃包含无效校验码的 SCTP 分组。

4.3.7 通路管理

发送端的 SCTP 用户能够使用一组传送地址作为 SCTP 分组的目的地。SCTP 管理功能可以根据 SCTP 用户的指令和当前合格的目的地集合的可达性状态，为每个发送的 SCTP 分组选择一个目的地传送地址。当其他分组业务量不能完全表明可达性时，通路管理功

能可以通过心跳消息来监视到某个目的地址的可达性，并当任何对端传送地址的可达性发生变化时，向 SCTP 用户提供指示。通路功能也用来在偶联建立时，向对端报告合格的本端传送地址集合，并把从对端返回的传送地址报告给本地的 SCTP 用户。

在偶联建立时，为每个 SCTP 端点定义一个首选通路，用来正常情况下发送 SCTP 分组。

在接收端，通路管理功能在处理 SCTP 分组前，用来验证入局的 SCTP 分组属于的偶联是否存在。

4.4 SCTP 原语

SCTP 通过接收高层协议（SCTP 用户）发送的原语请求，为 SCTP 的用户提供服务。同时 SCTP 可以根据不同事件向 SCTP 用户发送通知原语。

SCTP 原语描述使用了如下格式：

原语名：必备属性，任选属性

返回结果：必备属性，任选属性

4.4.1 SCTP 用户向 SCTP 发送的请求原语

SCTP 用户向 SCTP 发送的请求原语共有 16 种，含义如表 4-1 所示。

表4-1 SCTP 请求原语

原语名	功能
INITIALIZE	允许 SCTP 启动其内部的数据结构，并为建立操作环境分配所需的资源，一旦 SCTP 启动后，则高层协议在与其他 SCTP 端点之间通信时就不需要再调用此原语。 SCTP 将向高层协议返回本地准备处理 SCTP 偶联的事件号（实例）。
ASSOCIATE	由高层启动一个到特定端点的偶联。对端端点将按照该端点定义的传送地址的方式进行规定。如果偶联事件尚未启动，则认为该原语是一个错误。 用来本地处理 SCTP 偶联的偶联 ID，将作为返回结果用来返回偶联是否成功建立。如果偶联建立不成功，则返回一个差错。如果偶联成功，则返回结果中还应包含到对端的完整传送地址以及本端端点出局的流数量，同时还应从返回的目的地址中选择一个传送地址，该传送地址将作为本地端点向对端端点发送 SCTP 分组的首选通路。返回的“目的地传送地址列表”可以由 SCTP 用户用来改变首选通路或者是向一个特定传送地址强制发送一个 SCTP 分组。 返回结果：偶联 ID

原语名	功能
SHUTDOWN	用来正常地关闭一个偶联，任何以在本地发送队列中的用户数据都将被递交到对端。该偶联将在收到所有发送的 SCTP 分组的证实后停止。 返回结果用来指示是否成功关闭了该偶联。如果成功关闭，则反馈一个成功关闭偶联编码；如果关闭失败，则返回一个差错编码。
ABORT	用来非正常关闭一个偶联，本地发送队列中的用户数据将被丢弃，并发送一个 ABORT 数据块到对端。 返回结果用来指示是否成功中止了该偶联。如果中止成功，则返回一个已经中止的偶联编码。如果中止失败，则返回一个差错编码。
SEND	SCTP 用户使用该原语通知 SCTP 在指定流 ID 中向目的地发送地址发送数据。 返回结果用来指示是否成功发送了数据。
SET PRIMARY	高层协议使用该原语指示本地 SCTP 将指定的目的地传送地址作为发送分组的首选通路。 返回结果为结果编码，指示此操作是否成功执行。如果规定的目的地传送地址没包含在 ASSOCIATE 请求原语或 COMMUNCIATION UP 通知原语返回的“目的地传送地址列表”中，则返回一个差错。
RECEIVE	用来把在 SCTP 队列中的可用的用户消息读到由 SCTP 用户规定的缓冲区中。 所读消息的字节数将作为结果返回。如果有可能根据特定的规定，也可以返回其他消息，如发送方的地址、收到消息的流 ID、是否有消息可以进行恢复等。对于顺序的消息，消息的流顺序号码（SSN）也可以被返回。
STATUS	用来要求 SCTP 返回一个包含以下信息的数据块：偶联连接状态、目的地传送地址表、目的传送地址的可达性状态、当前的接收方窗口大小、当前的拥塞窗口大小、未确认的 DATA 数据块的数量、收到的 DATA 数据块的数量、首选通路、首选通路上最近收到的 SRTT、首选通路的 RTO。 返回结果为要求返回信息的状态。
CHANGE HEARTBEAT	高层协议用该原语指示本地端点允许或禁止向指定的目的地传送地址发送心跳消息。 返回结果用来指示该操作的执行情况。当目的传送地址未空闲时，心跳程序也不执行。
REQUEST HERATBEAT	高层协议用该原语指示本地端点对指定偶联的特定目的地址执行心跳程序。 返回结果用来指示传送给目的地址的 HEART BEART 数据块是否成功。

原语名	功能
GET SRTT REPORT	高层协议用该原语指示本地 SCTP 报告对给定偶联上规定的目的地传送地址的当前 SRTT 测量值。 返回结果是一个包含最近 SRTT 的毫秒值。
SET FRAILURE THRESHOLD	允许本地 SCTP 定制到给目的地传送地址的可达性故障检出的门限。 返回结果用来指示该操作是否成功。
SET PROTOCOL PARAMETERS	允许本地 SCTP 定制协议参数。 返回结果用来指示该操作是否成功。
RECEIVE UNSENT MESSAGE	高层协议用该原语指示本地 SCTP 将收到故障消息在高层协议缓存区储存。 返回结果为一个包含故障消息的字节数。
RECEIVE UNACKNOWLEDGED MESSAGE	高层协议用该原语指示本地 SCTP 将收到的没有应答故障消息在高层协议缓存区储存。 返回结果为一个包含没有应答消息的字节数。
DESTROY	指示本地哪个 SCTP 事件号（实例）被破坏。SCTP 事件号由 INITIALIZE 原语生成的。 返回结果为是否成功。

4.4.2 SCTP 向 SCTP 用户发送的通知原语

SCTP 向 SCTP 用户发送的通知原语共有 8 种，含义如表 4-2 所示。

表4-2 SCTP 通知原语

原语名	功能
DATA ARRIVE	当一个用户消息被成功接收，并且准备向 SCTP 用户递交时，SCTP 使用该原语通知高层用户。 如下信息会被传递： －偶联 ID：本地处理的 SCTP 偶联 －流 ID：用来指示数据从哪个流上接收到的。
SEND FAILURE	当一个消息不能递交时，SCTP 使用该原语通知 SCTP 用户。 如下信息会被传递： －偶联 ID：本地处理的 SCTP 偶联 －数据恢复 ID：用来恢复未发送和未证实数据的标识。 －原语编码：用来指示不能递交的原因，如长度过长、消息存活时间过期等。

原语名	功能
NETWORK STATUS CHANGE	当目的地传送地址被标为未激活（如 SCTP 检测出故障）或标记为激活时（SCTP 检测出故障恢复），SCTP 使用该原语通知 SCTP 用户。 如下信息会被传递： －偶联：本地处理的 SCTP 偶联 －目的地传送地址：指示由于状态变化而影响的对端端点的目的地传送地址。 －新状态：指示新的状态。
COMMUNCIATION UP	SCTP 用该原语通知 SCTP 用户，指示本地 SCTP 已经准备好发送或接收 SCTP 分组，或者时一个丢失通信的端点又已经恢复。 如下信息会被传递： －偶联 ID：本地处理的 SCTP 偶联 －状态：指示发生了哪种类型的事件 －目的地传送地址列表：对端端点的传送地址列表 －出局流数量：SCTP 用户允许使用的最大的流数量 －入局流数量：对端端点对该偶联所请求的流数量，此值可以与出局的流数量取值不同。
COMMUNICATION LOST	当 SCTP 完全丢失了到某一个端点的通信时（用心跳消息），或者是检测出端点已经执行了操作，SCTP 使用该原语通知 SCTP 用户。 如下信息会被传递： －偶联 ID：本地处理的 SCTP 偶联 －状态：指示发生了哪种类型的事件。状态可以指示故障或者是响应 SHUTDOWN 或 ABORT 请求原语的中止事件。 －数据恢复 ID：用来指示恢复未发送或证实数据 －最后证实的 TSN：对端端点最后证实的 TSN －最后发送的 TSN：发送到对端端点最后一个 TSN
COMMUNICATION ERROR	当 SCTP 从对端端点收到了一个 ERROR 数据块，并且确定需要通知高层用户时，才使用该通知原语。 如下信息会被传递： －偶联 ID：本地处理的 SCTP 偶联 －错误信息：指示错误类型并且可以任选地包含一些从 ERROR 数据块中收到的附加信息。
RESTART	当 SCTP 检测出对端端点已经重新启动时，使用该原语通知 SCTP 用户。 偶联 ID 会被传递。

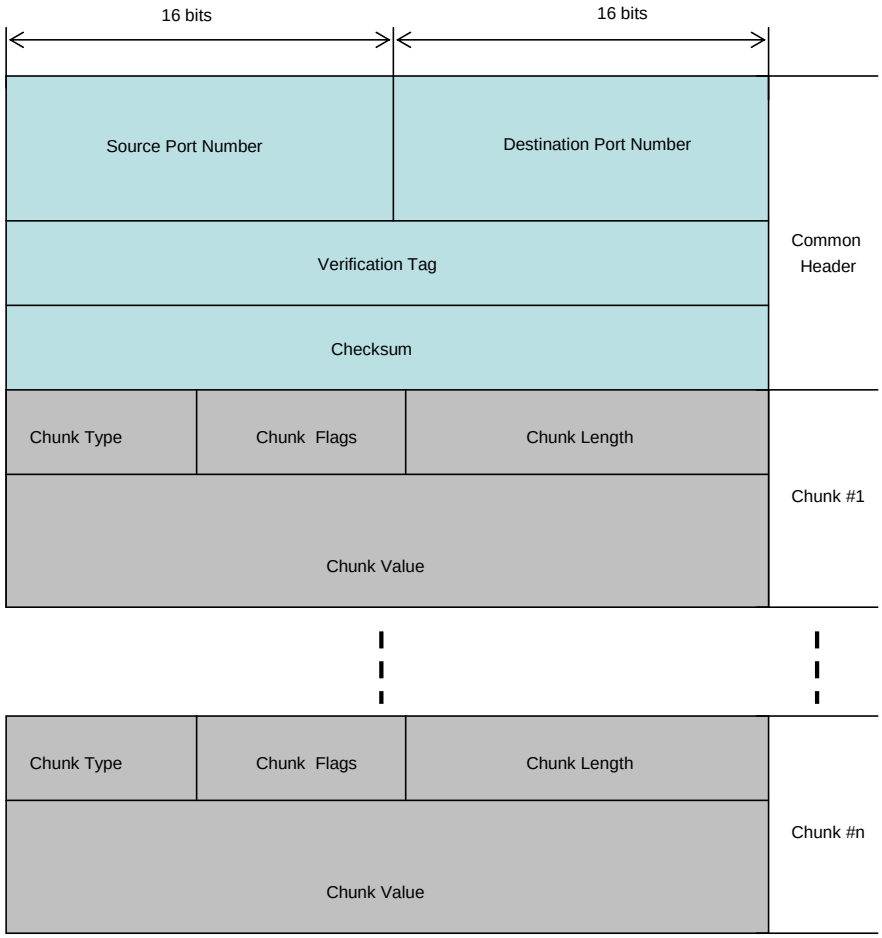
原语名	功能
SHUTDOWN COMPLETE	当本地 SCTP 已经完成了偶联关闭后，是使用此原语通知 SCTP 用户。 本地处理的 SCTP 偶联 ID 会被传递。

4.5 SCTP 协议消息

4.5.1 消息结构

每个 SCTP 分组结构如所图 4-4 示。

图4-4 SCTP 分组结构



一个 SCTP 分组含了一个公共的分组头（Common Header）和若干数据块（Chunk），每个数据块中既可以包含控制信息，也可以包含用户数据。除了 INIT、INIT ACK 和 SHUTDOWN COMPLETE 数据块外，其他类型的多个数据块可以捆绑在一个 SCTP 分

组中，以满足对 MTU 大小的要求。当然，这些数据块也可以不与其他数据块捆绑在一个分组中。如果一个用户消息不能放在一个 SCTP 分组中，这个消息可以被分成若干个数据块。

公共分组头的格式

SCTP 公共分组头中包括了源端口号（Source Port Number）、目的端口号（Destination Port Number）、验证标签（Verification Tag）和校验码（Checksum）。

- 源端口号（16 bits）

源端口号识别 SCTP 发送端点的 SCTP 端口号。接收方可以使用源端口号、源 IP 地址、目的端口号和目的 IP 地址标识该 SCTP 分组所属的偶联。

- 目的端口号（16 bits）

目的端口号为目的端点的 SCTP 端口号。接收主机可以使用目的端口号将 SCTP 分组解复用到正确的端点或应用中。

- 验证标签（32 bits）

验证标签是偶联建立时，本端端点为这个偶联生成一个随机标识。偶联建立过程中，双方会交换这个 TAG，到了数据传递时，发送端必须在公共分组头中带上对端的这个 TAG，以备校验。

- 校验码（32 bits）

SCTP 通过对用户数据使用 ADLER-32 算法，计算出一个 32 位的校验码，带在数据报中，在接收端进行同样的运算，通过检查校验码是否相等来验证用户数据是否遭到破坏。

数据块字段的格式

数据块包括了块类型（Chunk Type）、块标志位（Chunk　Flags）、块长度（Chunk Length）和块值（Chunk Value　）。

- 块类型（8 bits）

块类型定义在块值（Chunk Value）中消息所属的类型。表 4-3 列出了主要的块类型。

表4-3 SCTP 数据块消息类型

ID	块类型	说明
0	DATA（净数据）	传输的用户数据块。
1	INIT	用于发起两个端点之间的 SCTP 偶联。
2	INIT ACK	用来确认 SCTP 偶联的发起消息（INIT）。
3	SACK	该数据块送至对端，以确认收到 DATA 块，并且通知对端 DATA 的接收顺序间隙。
4	HEARTBEAT	端点发送该数据块至对端，以检测当前偶联中定义的某一目的地址的可达性。

ID	块类型	说明
5	HEARTBEAT ACK	响应 HEARTBEAT 消息。
6	ABORT	关闭偶联。
7	SHUTDOWN	偶联中的一个端点对其偶联发起一个 GRACEFUL 关闭。
8	SHUTDOWN ACK	响应 SHUTDOWN 消息，关闭程序完成时发出。
9	ERROR	通知对端，SCTP 偶联发生某种错误。
10	COOKIE ECHO	仅用于偶联发起过程，它由偶联的发起者发送至对端以完成发起程序。
11	COOKIE ACK	COOKIE 证实，相对于 COOKIE ECHO
12	ECNE	保留，应用于外部环境拥塞发布回声
13	CWR	保留，应用于降低拥塞窗口
14	SHUTDOWN COMPLETE	用于关闭程序完成时对 SHUTDOWN ACK 消息进行确认
15 至 62	-	IETF 保留
63	-	IETF 定义块扩展使用
64 至 126	-	IETF 保留
127	-	IETF 定义块扩展使用
128 至 190	-	IETF 保留
191	-	IETF 定义块扩展使用
192 至 254	-	IETF 保留
255	-	IETF 定义块扩展使用

如果接收端点不能识别块类型时，块类型最高位 2bit 用于标识需要进行的各种操作，比特组合含义如表 4-4 所示。

表4-4 接收端点不能识别块类型时，块类型最高 2bit 含义

Bits（最高两位）	含义
00	停止处理并丢弃此 SCTP 分组，不再处理该 SCTP 分组中的其他消息块。
01	停止处理并丢弃此 SCTP 分组，不再处理该 SCTP 分组中的其他消息块，并且在“ERROR”或“INIT ACK”中向发起端点返回不能识别的参数。

Bits（最高两位）	含义
10	跳过此数据块并继续执行。
11	跳过此数据块并继续执行，并且在“ERROR”或“INIT ACK”中向发起端点返回不能识别的参数。

- 数据块标志位（8bit）

块标志位用法由块类型决定。除非被置为其他值，块标记在传送过程中会被置 0 而且接收端点会忽视块标记。

- 块长度（16bit）

块长度包括块类型（Chunk Type）、块标记（Chunk Flags）、块长度（Chunk Length）和块值（Chunk Value），长度使用二进制表示。

- 块值（可变长度）

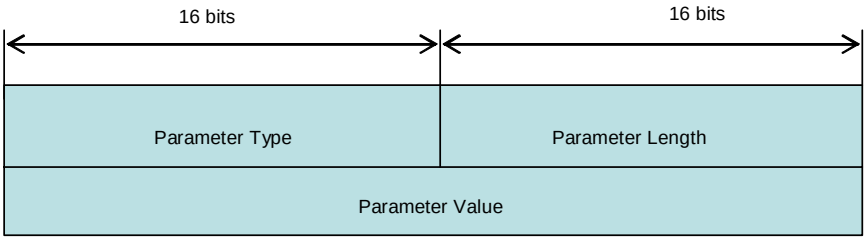
块值的内容在块中传送实际的信息，内容由消息块类型决定。块值的长度为不定长。

说明
数据块的总长度（包括类型、长度和取值长度）必须是 4 字节的整数倍，如果该长度不是 4 字节的整数倍，则发送方应当向数据块中填充全 0 的字节，这些填充的字节不计入数据块长度字段。发送方填充的字节数应不超过 3 个字节，在接收方忽略所有的填充字节。

任选/可变长参数的格式

SCTP 控制数据块（除 DATA 数据块外）的内容取值包含了数据块类型特定的数据块头所要求的字段，随后是一个或多个参数。这些包含在一个数据块中的任选/可变长参数都是按照参数类型、参数长度和参数取值的方式定义的，其格式如图 4-5 所示。

图4-5 任选/可变长参数格式



- 数据块的参数类型（16bit）

参数类型字段用来识别参数的类型，取值范围从 0 到 65534。65535 预留给 IETF 进行扩展。

如果接收端点不能识别参数类型时，参数类型中最高位 2bit 用于标识需要进行的各种操作，比特组合含义如表 4-5 所示。

表4-5 接收端点不能识别块参数类型时，参数类型最高 2bit 含义

Bits（最高两位）	含义
00	停止处理并丢弃此 SCTP 分组，不再处理该 SCTP 分组中的其他消息块。
01	停止处理并丢弃此 SCTP 分组，不再处理该 SCTP 分组中的其他消息块，并且在“ERROR”或“INIT ACK”的“不识别的参数类型”字段中报告不识别的参数类型。
10	跳过此数据块并继续执行。
11	跳过此数据块并继续执行，并且在“ERROR”或“INIT ACK”的“不识别的参数类型”字段中向发起端点返回不能识别的参数类型。

- 数据块的参数长度（16bit）

参数长度字段包含参数类型、参数长度和参数取值字段在内所有字段的字节数。因此一个参数的取值字段为 0，则该长度字段应设置为 4。参数长度字段不计算填充字节。

- 数据块的参数值（可变长度）

参数取值字段包含在该参数中传送的实际信息。

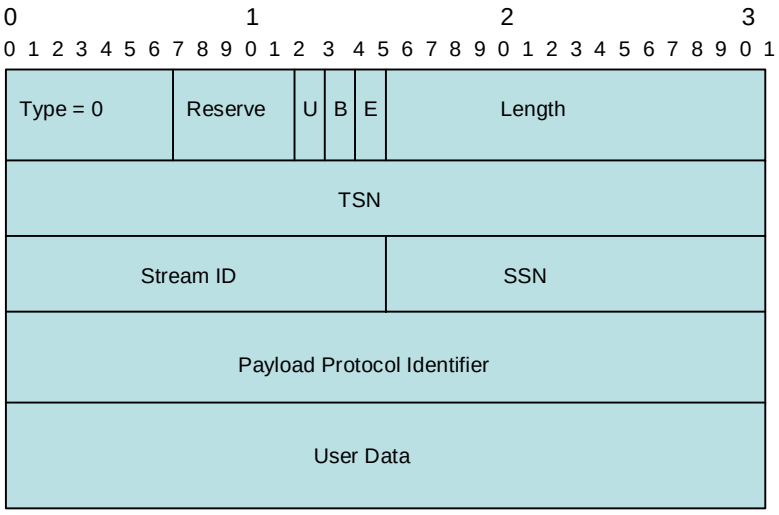
说明
参的总长度（包括类型、长度和取值字段）必须是 4 字节的整数倍。如果该长度不是 4 字节的整数倍，则发送方应当向数据块中填充全 0 的字节，这些填充的字节不计入参数长度字段。发送方填充的字节数应不超过 3 个字节，接收方忽略所有的填充字节。

4.5.2 SCTP 数据块的格式

净荷（DATA）数据块的格式

DATA 数据块格式如图 4-6 所示。

图4-6 DATA 数据块格式



- 数据块类型为 0
- 备用比特（5bit）：设置全为 0，在接收方忽略。
- U 比特（1 比特）

非顺序比特。如果该比特设置为 1，则指示这是一个非顺序的 DATA 数据块，不需要给数据块分配顺序号码。所有接收方必须忽略 SSN。

在重新组装完成后（如果需要），非顺序的数据块不需要尝试任何重新排序的过程，可以由接收方直接递交到 SCTP 用户。

如果一个非顺序的用户消息被分段，则消息的每个分段中的 U 比特必须设置为 1。

- B 比特

分段开始比特。如果该比特被设置，则指示这是用户消息的第一个分段。

- E 比特

分段结束比特。如果该比特被设置，则指示这是用户消息的最后一个分段。

一个没有分段的用户消息应当把所有的 B 和 E 比特设置为 1。

如果 B 和 E 比特都设置为 0，则表明这是一个分段的用户消息的一个中间分段。当用户消息被分段到多个数据块中，接收方需要使用 TSN 对消息进行重组，这意味着给分段的用户消息的每个分段都必须要使用连续的 TSN。BE 比特的取值含义如表 4-6 所示。

表4-6 BE 比特的取值含义

BE	表示的含义
10	用户消息的第一个分段
00	用户消息的中间分段
11	用户消息的最后一个分段

BE	表示的含义
11	未分段的消息

- 长度（16 比特）

指示 DATA 数据块从类型字段开始到用户数据字段结束之间的字节数，但不包含任何填充字节。如果 DATA 数据块的用户数据字段为 0，则长度字段设为 16。

- TSN（32 比特）

表示该数据块的 TSN，TSN 的有效值从 0 到 $2^{32}-1$。TSN 值达到 4294967295 后将转回到 0。

- Stream ID

用来识别用户数据属于的流。由 INIT 和 INIT ACK 数据块的发送者生成。

- SSN（16bit）

表示所在流中的用户数据的顺序号码。该字段的有效值从 0 到 65535。但一个用户消息被 SCTP 分段后，则必须在消息的每个分段中都带相同的流顺序号码。

- Payload Protocol Identifier（净负荷协议标识符）32bit

表示一个应用（或上层协议）特定的协议标识符。这个值由高层协议（SCTP 用户）传递到 SCTP 并发送到对等层。这个标识符不由 SCTP 使用，但可以由特定网络实体或对端的应用来识别在 DATA 数据块中携带的信息类型。甚至在每个分段的 DATA 数据块中也应包含该字段，以确保对网络中间的代理可用。

0 表示高层协议（SCTP 用户）未对该协议净荷规定应用标识符。

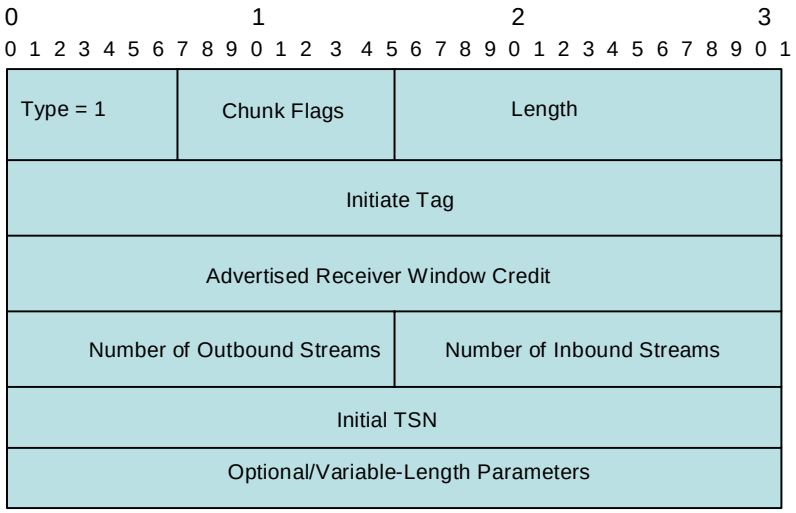
- User Data（用户数据，可变长度）

它用来携带用户数据净荷。该字段必须被填充为 4 字节的整数倍，发送方填充的字节数应不超过 3 个字节，接收方忽略所有的填充字节。

启动（INIT）数据块的格式

该数据块用来启动两个 SCTP 端点间的一个偶联，INIT 数据块的格式如图 4-7 所示。

图4-7 INIT 数据块格式



INIT 数据块应包括如下参数，除非特别支持，否则每个参数只能在 INIT 数据块中出现一次：

必备参数为：启动标签（Initiate Tag）、广播的接收方窗口信用值（Advertised Receiver Window Credit）、最大输出流数量（Number of Outbound Streams，OS）、最大输入流数量（Number of Inbound Streams，MIS)）、初始 TSN（Initial TSN）。

可变长度参数：IPV4 地址、IPV6 地址、Cookie Preservative、ECN 能力、主机名地址、支持的地址类型。

- INIT 数据块标志字段

该字段备用，所有比特应设为 0。INIT 中的参数可以按任何顺序进行处理。

- 启动标签（Initiate Tag）：32bit

INIT 的接收方记录启动标签参数值。这个值必须被放置到 INIT 接收方发送的与该偶联相关的每个 SCTP 分组中的验证标签字段中。

启动标签允许除 0 以外的任何值。如果在收到的 INIT 数据块中的启动标签为 0，则接收方必须作为错误处理，并且发送 ABORT 数据块中止该偶联。

- 广播的接收方窗口信用值（a_rwnd，32 比特）

表示专用的缓冲区的容量，用字节数表示。INIT 发送方为偶联预留的窗口。在偶联存活期间，这个缓冲区的容量不应减少（即不应把该偶联的专用缓冲区取走），但端点可以在发送的 SACK 数据块中修改 a_rwnd 的值。

- 输出流的数量（Number of Outbound Streams，OS）

定义发送 INIT 数据块的一方希望在该偶联中创建的输出流的数量。该值不允许为 0。接收方收到该参数为 0 的 INIT 数据块后会中止该偶联。

- 输入流的数量（Number of Inbound Streams，MIS）

定义了发送这个 INIT 块的一方允许对端在该偶联中所创建的流的数量。该值不允许为 0。接收方收到该参数为 0 的 INIT 数据块后会中止该偶联。

- 初始 TSN（Initial TSN）

定义发送方将使用的初始的 TSN，该值可以设置为启动标签字段的值。

- IP v4 地址

发送方端点的 IP v4 地址，采用二进制编码。INIT 数据块中可以包含多个 IP V4 或 IP V6 地址或地址组合。

- IP V6 地址

发送方端点的 IP V6 地址，采用二进制编码。INIT 数据块中可以包含多个 IP V4 或 IP V6 地址或地址组合。发送方不必把 IP V4 地址映射到 IP V6 地址中，可以直接在 IP V4 地址参数中使用 IP V4 地址。

与 SCTP 公共分组头中的源端口号一起，IP V4 或 IP V6 地址参数中的地址可以用来指示传送地址，并由 INIT 发送方所支持。在偶联存活期间，这个 IP 地址可以出现在 IP 包中起源地址字段中，由 INIT 的发送者来发送，并且可以由 INIT 的接收者作为 IP 包的目的地址。当 INIT 的发送方是一个多归属的情况时，多于一个 IP 地址参数可以包含在一个 INIT 数据块中。此外一个多归属的端点可以接入到不同类型的网络，这样多于一个的地址类型能够在 INTI 数据块中出现，即 IP V4 和 IP V6 的地址允许出现在同一个 INIT 数据块中。

如果 INTI 中包含了至少一个 IP 地址参数，则 IP 数据报中的源 IP 包含在 INIT 数据块中，INIT 中提供的其他附加地址可以被接收 INIT 的端点作为目的地。如果 INIT 中未包含任何 IP 地址参数，在收到 INIT 的端点必须使用收到的 IP 数据报中的源 IP 作为该偶联的目的地址。

- COOKIE Preservative

INIT 的发送方应使用这个参数来建议 INIT 的接收方提供较长的存活跨度的状态 COOKIE。由于失效的 COOKIE 操作差错原因，前一次尝试与对等端建立偶联失败后，又重新尝试偶联建立时，此参数由发送方添加到 INIT 数据块中。接收方出于安全的考虑可以选择忽略建议的 COOKIE 存活跨度增量。

COOKIE Preservative 参数中包含一个 32bit 的建议的 COOKIE 存活跨度（Suggested Cookie Life-span Increment）参数：此参数用来向接收方指示发送方希望接收方为其缺省的 COOKIE 的存活跨度增加的毫秒数。

- 主机名地址（Host Name Address）

INIT 发送方使用此参数把其主机名（在 IP 地址中的位置）传递到对等层。这个对等层负责解释这个主机名，用这个参数可以使偶联工作实现 NAT 穿越。

- 主机名（Host Name）

可变长度，该字段包含了按照 RFC1123 规定的“主机名句法”定义的主机名，主机名地址的解析不在本 SCTP 标准中规定。

该参数中至少有一个空的中止符包含在主机名字符串中，并且应包含长度。

- 支持的地址类型（Supported Address Types）

INIT 的发送方使用该参数列出其所支持的全部地址类型。

- 地址类型（Address Type）

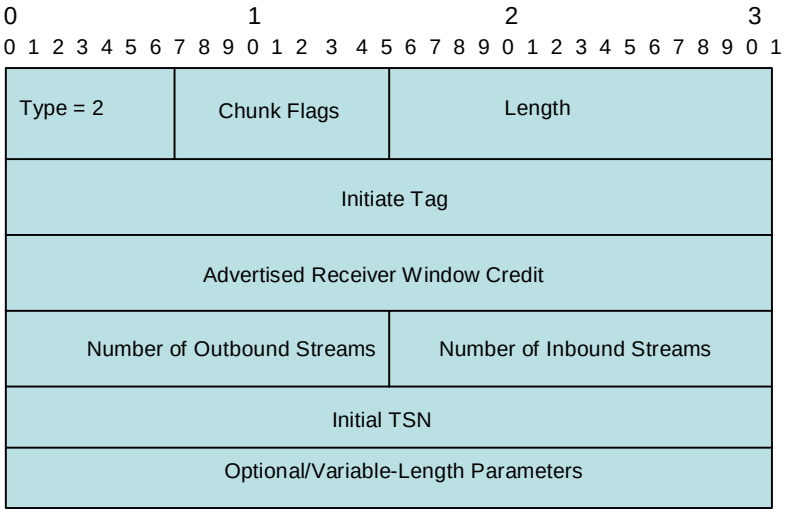
该参数使用对应的地址类型的类型值，如 IPV4＝5，IPV6＝6，主机名（Host Name）＝11。

启动证实（INIT ACK）数据块的格式

INIT ACK 数据块用来确认 SCTP 偶联的启动。

如图 4-8 所示，INIT ACK 的参数部分与 INIT 数据块的参数部分相同，还使用两个可变长度的参数：COOKIE（STATE COOKIE）和未识别的参数。

图4-8 INIT ACK 数据块格式



- 启动标签（Initiate Tag）32bit

INIT ACK 的接收方记录启动标签参数的值，并把该值放到每个 INIT ACK 接收方在相应的偶联上发送的每个 SCTP 分组中的验证标签。

启动标签不允许为 0。如果收到的 INIT ACK 数据块中的启动标签为 0，则接收方当作错误来处理并通过发送 ABORT 来关闭偶联。

- 广播的接收方窗口信用值（Advertised Receiver Window Credit）32bit

表示专用的缓冲区的容量，用字节数表示，INIT ACK 发送方为偶联预留的窗口，在偶联存活期间，这个缓冲区的容量不应减少。

- 输出流的数量（Number of Outbound Streams，OS）16bit

定义发送 INIT ACK 数据块的一方希望在该偶联中创建的输出流的数量。该值不允许为 0。接收方收到该参数为 0 的 INIT ACK 数据块后中止该偶联并舍弃 TCB。

- 输入流的数量（Number of Inbound Streams，MIS）16bit

定义发送 INIT ACK 数据块的一方允许对端端点在偶联中所创建的流的最大数量。该值不允许为 0。接收方收到该参数为 0 的 INIT ACK 数据块后中止该偶联并舍弃 TCB。

- 初始 TSN（ Initial TSN）32bit

定义发送方将使用的 TSN，该值可以设置为启动标签字段的值。

- IPV4 地址和 IP V6 地址

与 SCTP 公共分组头中的源端口号一起，IP V4 或 IP V6 地址参数中的地址可以用来指示传送地址，并由 INIT ACK 发送方所支持。在偶联存活期间，这个 IP 地址可以出现在 IP 包中起源地址字段中，由 INIT ACK 的发送者来发送，并且可以由 INIT ACK 的接收者作为 IP 包的目的地址。当 INIT ACK 的发送方是一个多归属的情况时，多于一个 IP 地址参数可以包含在一个 INIT 数据块中。此外一个多归属的端点可以接入到不同类型的网络，这样多于一个的地址类型能够在 INTI 数据块中出现，即 IP V4 和 IP V6 的地址允许出现在同一个 INIT 数据块中。

如果 INTI ACK 中包含了至少一个 IP 地址参数，则 IP 数据报中的源 IP 包含在 INIT ACK 数据块中，INIT ACK 中提供的其他附加地址可以被接收 INIT ACK 的端点作为目的地。如果 INIT ACK 中未包含任何 IP 地址参数，在收到 INIT ACK 的端点必须使用收到的 IP 数据报中的源 IP 作为该偶联的目的地址。

- 状态 COOKIE（State COOKIE）可变长度

该参数长度取决于 COOKIE 的长度，该参数值的取值必须包含由 INIT ACK 发送方创建该偶联所需的所有状态、参数信息和消息授权码。

- 不识别参数（Unrecognized Parameters）可变长度

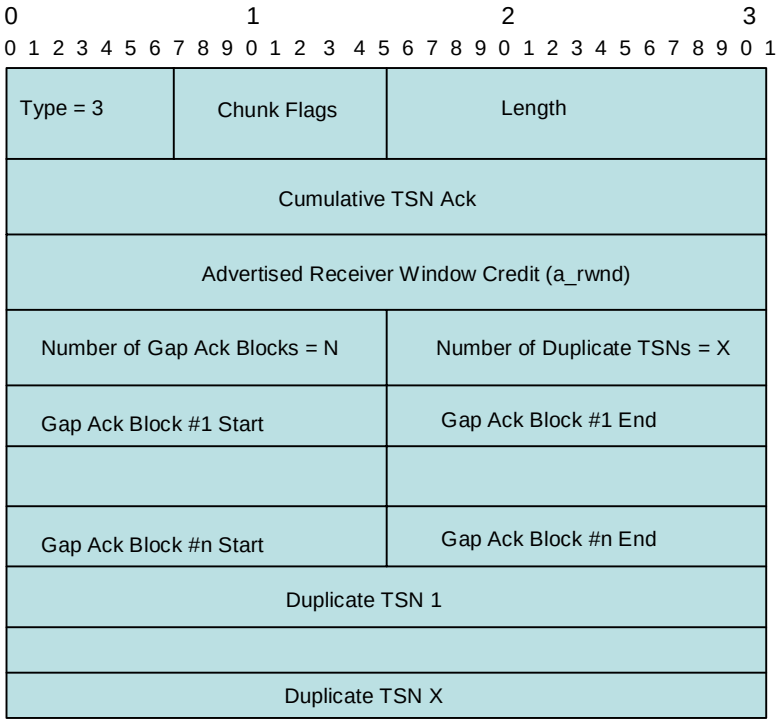
该参数内容是 INIT 数据块中包含的一个不识别的参数，该参数用来返回给 INIT 数据块的产生者一个指示。此参数字段包含了从 INIT 数据块中复制过来的不识别参数的完整的参数类型、长度和参数值。

选择证实（SACK）数据块的格式

SACK 通过使用 DATA 数据块中的 TSN 用来向对端端点确认接收到的 DATA 数据块，并通知对端端点在收到的 DATA 数据块中的间隔。所谓间隔就是指收到的 DATA 数据块的 TSN 不连续的情况。

SACK 数据块格式如图 4-9 所示。

图4-9 SACK 数据块格式



- 数据块类型（Type）

数值为 3。

- 数据块标志

设为全 0 并由接收方忽略。

- 累积 TSN 证实（Cumulative TSN Ack）

指收到的 TSN 顺序断开前的最后一个 TSN 号码，下一个 TSN 则是在发送 SACK 的端点尚未收到 TSN 值。这个参数包含了在收到 TSN 序列的间隔前的最后一个 TSN 值。此参数确认已经收到了小于或等于该值的所有 TSN。

- 广播的接收方窗口信用值（Advertised Receiver Window Credit (a_rwnd)）

指示修改了 SACK 的发送方的接收缓冲容量的字节数。

- 间隔证实块的数目（Number of Gap Ack Blocks = N ）

指示 SACK 数据块中包含的间隔证实块的数目。每个间隔证实块确认了在一个不连续 TSN 后所收到的 TSN 序列，所有通过间隔证实块确认的 TSN 都比累积 TSN 证实的值大。

- 重复的 TSN 数目（Number of Duplicate TSNs = X）

包含了该端点收到的重复的 TSN 的数目。每个重复的 TSN 都列在间隔证实块列表后。

- 间隔证实块（Gap Ack Blocks）

此字段包含了间隔证实块，根据间隔证实块数量字段给出的值，间隔证实块重复若干次。所有 TSN 大于或等于累积 TSN 证实＋间隔证实块开始的 DATA 数据块，或者是小于或等于每个间隔证实块的累积 TSN 证实＋间隔证实块结束的 DATA 数据块都可以认为是被正确接收了。

- 间隔证实块开始（Gap Ack Block Start）

该字段用来指示这个间隔整数块的开始 TSN 偏移。为了计算实际的 TSN 号码必须用累积 TSN 证实加上偏移号码。计算出来的 TSN 标识用于识别第一个在这个间隔证实块中被收到的 TSN。

- 间隔证实块开始（Gap Ack Block End）

该字段用来指示这个间隔整数块的结束 TSN 偏移。为了计算实际的 TSN 号码必须用累积 TSN 证实加上偏移号码。计算出来的 TSN 标识用于识别在这个间隔证实块中最后收到的 TSN。

- 重复的 TSN（Duplicate TSN）

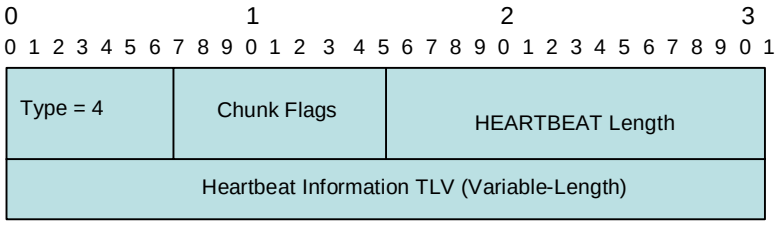
用来指示一个在上一个 SACK 发送后收到 TSN 重复个数。每次一个接收者收到一个重复的 TSN(在发送 SACK 前)，则把这个 TSN 加到重复的 TSN 列表中。每发送一次 SACK 后，则把统计重复 TSN 的计数器重新置 0。

Heart Beat 请求（HEARTBEAT）数据块的格式

SCTP 端点通过向对端端点发送这个数据块，从而检测定义在该偶联上到特定目的地传送地址的可达性。

HEARTBEAT 数据块的格式如图 4-10 所示。

图4-10 HEARTBEAT 数据块格式



- 数据块类型（Type）8bit

此值为 4。

- 数据块标志（Chunk Flags）8bit

在发送方设置为全 0，并在发送方忽略。

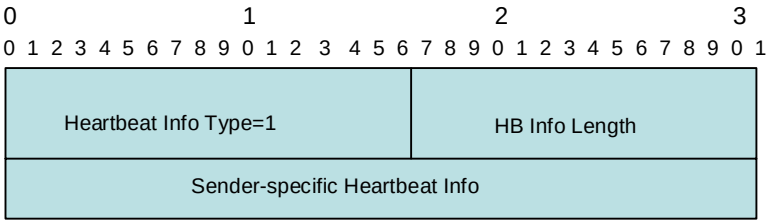
- HEARTBEAT 长度（HEARTBEAT Length）

设置为数据块长度的字节数，包括数据块头和 HEARTBEAT 信息参数的长度。

- HEARTBEAT 信息参数（HERATBEAT Information TLV）

HEARTBEAT 参数字段包含 HEARTBEAT 信息（Heartbeat Information TLV），HEARTBEAT 信息是一个可变长度的非透明数据结构，其信息通常只需要发送方明白即可。HEARTBEAT 信息参数格式如图 4-11 所示。

图4-11 HEARTBEAT 信息参数格式



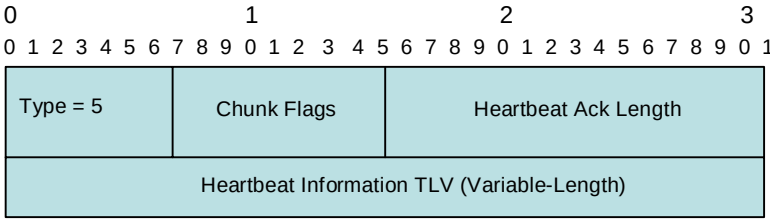
当该 HEARTBEAT 数据块发送到目的地传送地址时，发送方特定的 HEARTBEAT 信息字段（Sender-specific Heartbeat Info）通常包括关于发送方当前的时间信息。

Heart Beat 证实（HEARTBEAT ACK）数据块的格式

SCTP 端点在收到对端端点发来的 HEARTBEAT 数据块后，则发送该数据块作为响应。HeartBeat 证实数据块总是向包含 HEARTBEAT 数据块的 IP 数据包中的起源 IP 地址发送，来作为对该 HEARTBEAT 数据块的响应。

HEARTBEAT ACK 数据块格式如图 4-12 所示。

图4-12 HEARTBEAT ACK 数据块格式



- 数据块类型（Type）8bit

此值为 5。

- 数据块标志位（Chunk Flags）8bit

在发送方设置为全 0，并在接收方忽略。

- HEARTBEAT 证实长度（HEARTBEAT ACK Length）

设置为数据块长度的字节数，包括数据块头和 HEARTBEAT 信息参数的长度。

- HEARTBEAT 信息参数（HEARTBEAT Information TLV）

可变长度，该字段的内容把 HEARBEAT 请求数据块中的 HEARTBEAT 信息参数作为回送的响应，该参数字段包含一个可变长度的非透明的数据结构。

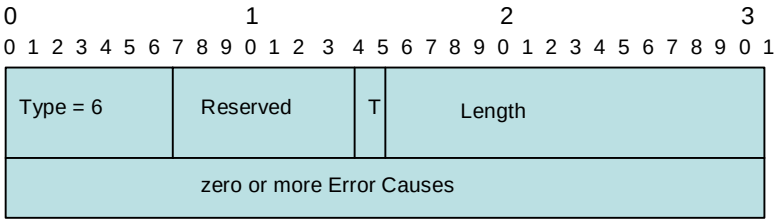
中止（ABORT）数据块的格式

SCTP 端点发送 ABORT 数据块来中止到对端端点的偶联，ABORT 数据块中可以包含原因参数来通知接收 ABORT 数据块的一方中止该偶联的原因。DATA 数据块不能与 ABORT 数据块捆绑在一个 SCTP 分组中。SCTP 控制数据块中，除 INIT、INIT ACK、SHUTDOWN COMPLETE 数据块外的数据块都可以与 ABORT 捆绑在一个 SCTP 分组中，但这些捆绑的控制数据块都应放在 SCTP 分组中的 ABORT 数据块之前，否则这些控制数据块会被接收方忽略。

如果一个端点收到了格式错误或与不存在的偶联相关的 ABORT 消息，则应当舍弃该消息。此外，在任何情况下，端点收到一个 ABORT 消息后，都不能通过发送 ABORT 消息作为响应。

ABORT 数据块的格式如图 4-13 所示。

图4-13 ABORT 数据块格式



- 数据块类型（8bit）

该值为 6。

- 数据块标志位（Chunk Flags）8bit

其中高 7 比特备用，在发送方设置为全 0，并在接收方忽略。当发送方由一个 TCB（Transmission Control Block）被破坏时，则 T 比特设置为 0；如果发送方没有 TCB，则 T 比特设置为 1。

- 长度（Length）16bit

设置为该数据块的长度，包括数据块头和所有包含的差错原因字段。

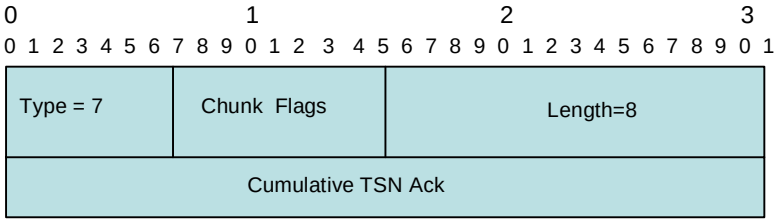
- 0 或多个差错原因（zero or more Error Causes）

ABORT 数据块的信息内容。

关闭偶联（SHUTDOWN）数据块的格式

偶联的端点可以使用这个数据块启动对该偶联的正常关闭程序。SHUTDOWN 数据块的格式如图 4-14 所示。

图4-14 SHUTDOWN 数据块格式



- 数据块类型（TYPE）

该值为 7。

- 数据块标志（Chunk Flags）8bit

在发送方设置为全 0，接收方忽略。

- 长度（Length）

指示 SHUTDOWN 数据块的长度，该字段设置为 8。

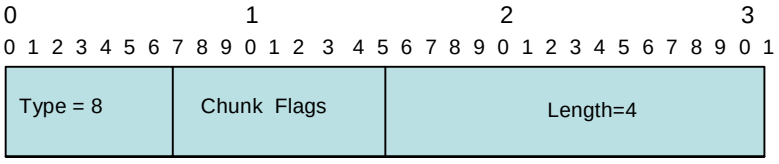
- 累积的 TSN 证实（Cumulative TSN Ack）

包含了在任何间隔前收到的最后一个数据块的 TSN。由于 SHUTDOWN 消息不包含间隔证实块，因此，不能用来对收到的非连续 TSN 进行证实。

关闭证实（SHUTDOWN ACK）数据块的格式

在完成了偶联关闭后，必须使用 SHUTDOWN ACK 数据块确认收到的 SHUTDOWN 数据块。SHUTDOWN ACK 数据块的格式如图 4-15 所示。

图4-15 SHUTDOWN ACK 数据块格式



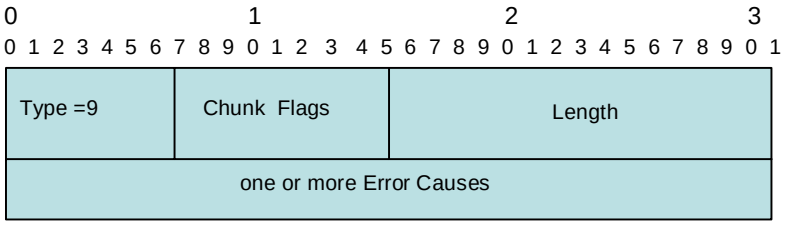
数据块标志位（Chunk Flags）：在发送方设置为全 0，并在接收方忽略。

SHUTDOWN ACK 中不再包含其他参数，因此长度设为 4。

操作差错（ERROR）数据块的格式

SCTP 端点发送 ERROR 数据块向其他对端端点通知一些特定的差错情况。该数据块中可以包含一个或多个差错原因。一般操作差错不一定是致命的。致命差错情况的报告一般使用 ABORT 数据块。ERROR 数据块格式如图 4-16 所示。

图4-16 ERROR 数据块格式



- 数据块格式（Type）8bit

该值为 9。

- 数据块标志位（Chunk Flags）8bit

在发送方设置为全 0，并在接收方忽略。

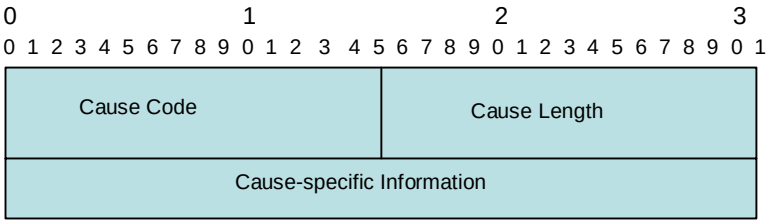
- 可变长度（Length）16 位

设置为该数据块的字节数，包括数据块头和所有包含的差错原因字段的长度。

- 差错原因参数（Error causes）

差错原因参数包括原因编码、原因长度和原因特定的信息，格式如图 4-17 所示。

图4-17 差错原因参数的格式



原因特定信息由原因编码确定，其对应关系如表 4-7 所示。

表4-7 原因特定信息与原因编码对应关系

原因编码	含义	参数信息
1	无效的流标识符：指示收到了一个关于不存在的流的 DATA 数据块	流 ID（16bit）：包含了接收的差错的 DATA 数据块的流 ID。
		备用字段（16bit）：由发送方设为全 0，在接收方忽略。
		原因长度＝8

原因编码	含义	参数信息
2	丢失必备参数：指示一个或多个必备的参数在收到的 INIT 或 INIT ACK 数据块中丢失。	丢失的参数个数（32bit）：指示丢失的参数个数。
		丢失的参数类型（16bit）：丢失的必备参数号。
		原因长度＝8＋N×2
3	过期的 COOKIE 差错：指示收到的有效的 State Cookie 已经过期了。	过期测量（32bit）：包含了当前时间和 State Cookie 过期时的时间差值（用微妙表示）。该差错原因的发送方可以通过在该字段中包含一个非 0 的值来报告 State Cookie 过期了多长时间。如果发送方不希望提供此信息，则该字段设置为 0。
		原因长度＝8
4	资源耗尽：指示发送方的资源已经耗尽，通常情况下，该差错原因与 ABORT 数据块一起发送。	原因长度＝4
5	不可解析的地址：指示发送方不能解析特定的地址参数（即发送方不支持该类地址类型），通常情况下，该差错原因与 ABORT 数据块一起发送。	不可解析的地址（可变长度）：不能解析的完整的地址参数或主机名参数（类型、长度和地址值）。
		原因长度为可变长度。
6	不识别的数据块类型：如果接收方不能识别数据块类型而且数据块类型比特中的高位比特设为 1，则将不识别的数据块类型错误返回给数据块的发送方。	不识别的数据块（可变长度）：该字段包含 SCTP 分组中不识别数据块的类型、数据块标志和数据块长度。
		原因长度为可变长度。
7	无效的必备参数：当一个必备参数被设置成无效值时，则向 INIT 或 INIT ACK 的生成者返回无效的必备参数差错原因。	原因长度＝4
8	不识别的参数：如果接收方不能识别 INIT ACK 数据块中一个或多个任选参数时，则向 INIT ACK 数据块的发送方返回该参数。	不识别参数（可变长度）：包含了从 INIT ACK 数据块中复制的完整的不识别参数。当 COOKIE ECHO 数据块的发送者希望报告不识别的参数时，此参数通常是包含在 ERROR 数据块中与 COOKIE ECHO 数据块捆绑在一起发送作为对 INIT ACK 的响应。
		原因长度为可变长度。

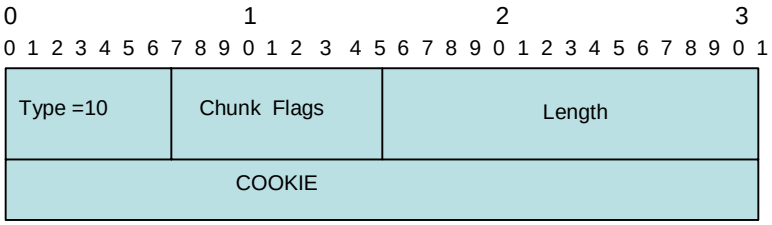
原因编码	含义	参数信息
9	无用户数据：如果收到的 DATA 数据块中未包含用户数据，则把此差错原因返回给发送方。	TSN：接收到的没有用户数据的 DATA 数据块 TSN 值。
		原因长度＝8
10	关闭阶段收到 COOKIE：当端点处于 SHUTDOWN-ACK-SENT 状态，收到 COOKIE ECHO 时，则发送此差错原因。	原因长度＝4

状态 COOKIE（COOKIE ECHO）数据块的格式

COOKIE ECHO 数据块只在启动偶联时使用，它由偶联的发起者发送到对端端点，以完成启动的过程。COOKIE ECHO 必须在该偶联上发送的 DATA 数据块前发送，但可以与其他的 DATA 数据块捆绑到同一个 SCTP 分组中。

COOKIE ECHO 数据块的格式如图 4-18 所示。

图4-18 COOKIE EHCO 数据块的格式



- 数据块类型（Type）8bit

该值为 10。

- 数据块标志位（Chunk Flags）8bit

在发送方设置为全 0，在接收方忽略。

- 长度（Length）16bit

该数据块长度的字节数，包括 4 字节的数据块头和 COOKIE 的长度。

- COOKIE（可变长度）

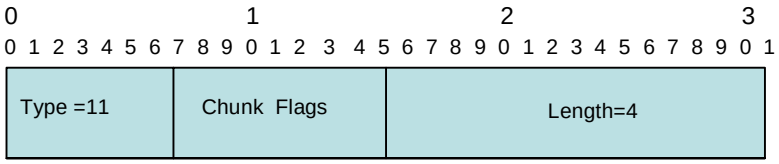
该字段必须包含从前一个 INIT ACK 数据块的状态 COOKIE（State COOKIE）参数中收到准确的 COOKIE，使用 COOKIE 时应尽可能的从小从而保证互操作性。

COOKIE 证实（COOKIE ACK）数据块的格式

COOKIE ACK 数据块只在启动偶联时使用，它用来证实收到 COOKIE ECHO 数据块。这个数据块必须在该偶联上发送任何 DATA 或 SACK 数据块前发送，但这个数据块可以与一个或多个 DATA 或 SACK 数据块捆绑在一个 SCTP 分组中发送。

如图 4-19 所示，COOKIE ACK 数据块中没有任何其他参数。

图4-19 COOKIE ACK 数据块格式



- 数据块标志（Chunk Flags）8bit

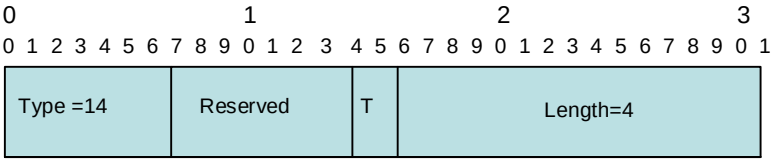
在发送方设置为全 0，并在接收方忽略。

关闭完成（SHUTDOWN COMPLETE）数据块的格式

此数据块在完成关闭程序后用来确认收到的 SHUTDOWN ACK 数据块。

如图 4-20 所示，SHUTDOWN COPLIETE 数据块中不含其他参数。

图4-20 SHUTDOWN COMPLETE 数据块的格式



- 数据块类型（Type）8bit

其中高 7 比特备用。备用比特在发送方设置全为 0，在接收方忽略。

- T 比特（1 比特）

当发送方有一个 TCB 被破坏时，则该 T 比特设置为 0。如果发送方没有 TCB，则把该比特设置为 1。

4.5.3 SCTP 端点维护的参数和建议值

对应每个 SCTP 实例所需的参数

对应每个 SCTP 实例所需的参数如表 4-8 所示。

表4-8 对应每个 SCTP 实例所需的参数

参数	含义
偶联	当前偶联列表，对应每个偶联的数据用户。
密钥	端点使用密钥来计算 MAC，这应当是有足够长度的加密质量随机数，RFC1750 对密钥的选择有一些较为有益的说明。
地址列表	一组该实例绑定的 IP 地址的列表，这个参数在 INIT 或 INIT ACK 数据块中被传递到对端端点。
SCTP 端口	端点绑定的本地 SCTP 端口。

对应每个偶联 SCTP 端点所需的参数

对应每个偶联 SCTP 端点所需参数如表 4-9 所示。

表4-9 对应每个偶联 SCTP 端点所需的参数

参数	含义
对端验证标签	指收到的 INIT 或 INIT ACK 数据块中启动标签（Initiate Tag）字段中的值。
本地验证标签	指发出的 INIT 或 INIT ACK 数据块中启动标签（Initiate Tag）字段中的值。
对端传送地址类别	一组该实例绑定的 IP 地址的列表，这个参数在 INIT 或 INIT ACK 数据块中被传递到对端端点。
首选通路	端点绑定的本地 SCTP 端口。
全局差错计数	整个偶联的所有差错计数。
全局差错门限	这个门限用来控制偶联，当全局差错计数达到了这个门限，将导致偶联的关闭或中止。
对端的 RWND	对端的 Rwnd 的当前计算值。
下一个 TSN	下一个 TSN 号码被分配给一个的 DATA 数据块，它可以在 INIT 或 INIT ACK 数据块中发送到对端，并且这个号码每分配给 DATA 数据块（通常的情况是在发送前或者是分段时）后加 1。
最后收到的 TSN	这是最后一个按顺序收到的 TSN，这个值最初是使用对端的初始 TSN 来设定的，并在收到的 INIT 或 INIT ACK 数据块中携带，并把该值减 1 得到。
映射数组	这是一个以比特或字节定义的数组，它用来指示哪个收到的 TSN 是非连续的（相对于最后收到的 TSN）。如果不存在着不连续的情况，即没有收到失序的分组，这个数组将被设置为全 0。

参数	含义
ACK 状态	这个标志位用来指示下一个收到的分组是否应当响应 SACK，其初始值为 0。
入局流	一个用来跟踪入局流的数组结构，通常包含下一个希望收到的流顺序号码的可能的流号码。
出局流	一个用来跟踪出局流的数组结构，通常包含下一个希望在某个流上发送的流顺序号码。
Reship Queue	一个重装队列。
本地传送地址列表	该偶联绑定的本地 IP 地址
偶联的 PMTU	对所有对端端点传送地址发现的最小的 PMTU（Path MTU）

说明
对于一个给定的偶联，两个端点间使用的验证标签值在偶联的存活期间不需要改变。但无论何时端点在清除偶联后，再重新建立到对端的偶联时则必须重新使用一个验证标签值。

对应每个传送地址所需的参数

对应于从 INIT 或 INIT ACK 中收到的对端端点地址列表中每个目的地传送地址，端点都需要维护如表 4-10 所示的参数。

表4-10 对应每个传送地址所需的参数

参数	含义
差错计数	对该目的地的当前差错计数。
差错门限	对该目的地的当前差错门限，当差错计数到达该值时，则标记到该目的传送地址的偶联停止。
CWND	当前的拥塞窗口。
RTO（Retransmission Timeout Value）	当前的重发超时取值。
SRTT（Smoothed Round Trip Time）	当前的平滑双向时延值。
RTTVAR（RTT Variation）	当前双向传播时间变化。
部分字节证实	在拥塞避免模式下，CWND 增加的跟踪方法。
状态	目的地的当前状态，包括 DOWN、UP、ALLOW-HEARTBEAT、NO-HEARTBEAT。
PMTU	当前已通知的通路 MTU

参数	含义
每个目的地的定时器	针对每个目的地使用的定时器
最后使用时间	指示最后向该目的地发送分组的时间，用来确定是否需要发送 HEART BEAT。

需要的通用参数

- 出局队列：出局 DATA 数据块的队列。
- 入局队列：入局 DATA 数据块的队列。

SCTP 参数的建议值

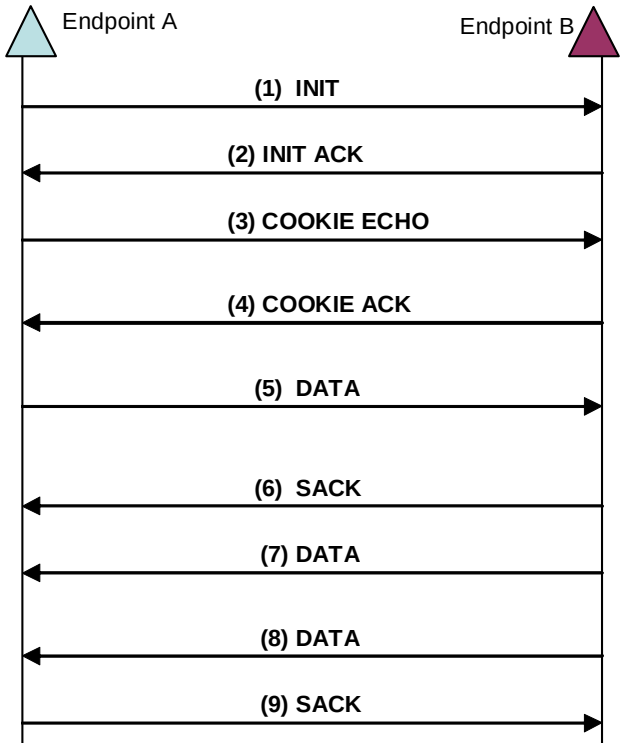
- RTO 的初始值：3s
- RTO 的最小值：1s
- RTO 的最大值：60s
- RTO Alpha：1/8
- RTO Beta：1/4
- 有效的 COOKIE 寿命：60s
- 偶联的最大重传次数：10 次
- 通路的最大重传次数：5 次
- IINIT 的最大重传次数：8 次
- HEARTBEAT 测试周期：30s

4.6 SCTP 基本信令流程

4.6.1 偶联的建立和发送流程

SCTP 端点 A 启动建立偶联，并向端点 B 发送一个用户消息，随后端点 B 向 A 发送两个用户消息。（假定这些消息没有捆绑和分段）。信令流程如图 4-21 所示。

图4-21 偶联建立过程消息交互图



1. 端点 A 创建一个数据结构 TCB（传输控制块）来描述即将发起的这个偶联（包含偶联的基本信息），然后向端点 B 发送 INIT 数据块。INIT 数据块中主要包括如下参数：

- 启动标签（Initiate Tag）：对端验证标签，如设为 Tag_A。Tag_A 是从 1 到 4294967295 中的一个随机数。
- 输出流数量（OS）：本端点期望的最大出局流的数量。
- 输入流数量（MIS）：本端点允许入局流的最大数量。

说明

对端点 A、端点 B 而言，当收到对端端点的流信息后，都需要进行相关的检查。如果对端的最大入局流数量比本端端点最大出局的流数量小，意味着对端端点不能支持本端端点期望的出局流的数量，此时，本端端点可以使用对端端点最大入局流的数量作为本端端点出局流的数量，也可以中止偶联并向 SCTP 用户报告对端端点资源短缺。

端点 A 发送 INIT 后启动一个 INIT 定时器，并进入 COOKIE-WAIT 状态。

说明

INIT 定时器作用是等待对端端点返回 INIT ACK 消息块。如果定时器超期仍收不到 INTI ACK 消息块，本端端点则重发 INIT 数据块，直达到最大重发的次数。

2. 端点 B 收到 INIT 消息后，立即用 INIT ACK 数据块响应。INIT ACK 数据块中必须带有如下参数：

- 目的地 IP 地址：设置成 INIT 数据块的起源 IP 地址。
- 启动标签（Initiate Tag）：设置成 Tag_B。

- 状态 COOKIE（STATE COOKIE）：根据偶联的基本信息生成一个 TCB，不过这个 TCB 是一个临时 TCB。这个 TCB 生成以后，将其中的必要信息（包含一个 COOKIE 生成的时间戳、COOKIE 的生命期）和一个本端的密钥通过 RFC2401 描述的算法计算成一个 32 位的摘要 MAC（这种计算是不可逆的）。必要信息和 MAC 组合成 STATE COOKIE 参数。
- 本端点传送地址。
- 最大入局流的数量。
- 最大出局流的数量。

3. 端点 A 收到 INIT ACK 后，首先停止 INIT 定时器离开 COOKIE-WAIT 状态，然后发送 COOKIE ECHO 数据块，将收到 INIT ACK 数据块中的 STATE COOKIE 参数原封带回。最后端点 A 启动 COOKIE 定时器并进入 COOKIE-ECHOED 状态。

注意

COOKIE ECHO 数据块能够与 DATA 数据块捆绑在一个 SCTP 分组中发送，但 COOKIE ECHO 必须是分组里的第一个数据块。除非收到返回的 COOKIE ACK 数据块，否则发送端点不能给对端端点发送其他分组。

4. 端点 B 收到 COOKIE ECHO 数据块后，进行 COOKIE 验证。将 STATE COOKIE 中的 TCB 部分和本端密钥根据 RFC2401 的 MAC 算法进行计算，得出的 MAC 和 STATE COOKIE 中携带的 MAC 进行比较。如果不同则丢弃这个消息；如果相同，则取出 TCB 部分的时间戳，和当前时间比较，看时间是否已经超过了 COOKIE 的生命期。如果是，同样丢弃。否则根据 TCB 中的信息建立一个和端 A 的偶联。端点 B 将状态迁入 ESTABLISHED，并发出 COOKIE ACK 数据块。端点 B 向 SCTP 用户发送 SCOMMUNCIATION UP 通知。

注意

COOKIE ACK 数据块能够与 DATA、SACK 数据块捆绑在一个 SCTP 分组中发送，但 COOKIE ACK 必须是分组里的第一个数据块。

端点 A 收到 COOKIE ACK 数据块后，从 COOKIE-ECHOED 状态迁移到 ESTAABLISHED 状态，并停止 COOKIE 定时器。端点 A 使用 COMMUNICATION UP 通知 SCTP 用户偶联建立成功。

说明

一个偶联的建立包括 4 次握手过程：INIT、INIT ACK、COOKIE ECHO 和 COOKIE ACK。

5. 端点 A 向端点 B 发送一个 DATA 数据块，启动 T3-RTS 定时器。DATA 数据块中必须带有如下参数：

- TSN：DATA 数据块的初始 TSN。
- 流标识符（Stream Identifier）：用户数据属于的流，假设流标识符为 0。
- 流顺序码（Stream Sequence Number）：所在流中的用户数据的顺序号码。该字段从 0 到 65535。

- 用户数据（User Data）：携带用户数据净荷。

6. 端点 B 收到 DATA 数据块后，返回 SACK 数据块。SACK 数据块中必须带有如下参数：

- 累积证实 TSN 标签（Cumulative TSN Ack）：端点 A 的初始 TSN。
- 间隔块（Gap Ack Block）：此值为 0。

端点 A 收到 SACK 数据块后，停止 T3-RTX 定时器。

7. 端点 B 向端点 A 发送第一个 DATA 数据块。DATA 数据块中必须带有如下参数：

- TSN：端点 B 发出 DATA 数据块的初始 TSN。
- 流标识符（Stream Identifier）：用户数据属于的流，假设流标识符为 0。
- 流顺序码（Stream Sequence Number）：所在流中的用户数据的顺序号码。假设流顺序码为 0。
- 用户数据（User Data）：携带用户数据净荷。

8. 端点 B 向端点 A 发送第二个 DATA 数据块。DATA 数据块中必须带有如下参数：

- TSN：端点 B 发出 DATA 数据块的初始 TSN＋1。
- 流标识符（Stream Identifier）：用户数据属于的流，假设流标识符为 0。
- 流顺序码（Stream Sequence Number）：所在流中的用户数据的顺序号码。此时流顺序码为 1。
- 用户数据（User Data）：携带用户数据净荷。

9. 端点 A 收到 DATA 数据块后，返回 SACK 数据块。SACK 数据块中必须带有如下参数：

- 累积证实 TSN 标签（Cumulative TSN Ack）：端点 B 的初始 TSN。
- 间隔块（Gap Ack Block）：此值为 0。

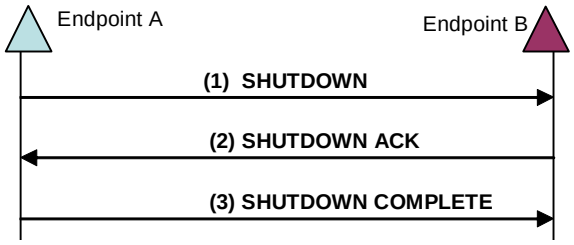
4.6.2 偶联关闭流程

一个端点退出服务时，需要停止它的偶联。偶联的停止使用两种流程：偶联的中止流程（非正常关闭）和偶联的正常关闭流程。

偶联的中止（非正常关闭）可以在任何未完成期间进行，偶联的两端都舍弃数据并且不提交到对端。此种方法不考虑数据的安全。偶联的中止步骤比较简单：发起端点向对端端点发送 ABORT 数据块，发送的 SCTP 分组中必须填上对端端点的验证标签，而且不在 ABORT 数据块中捆绑任何 DATA 数据；接收端点收到 ABORT 数据块后，进行验证标签的检查。如果验证标签与本端验证标签相同，接收端点从记录上清除该偶联，并向 SCTP 用户报告偶联的停止。

偶联的正常关闭：任何一个端点执行正常关闭程序时，偶联的两端将停止接受从其 SCTP 用户发来的新数据，并且在发送或接收到 SHUTDOWN 数据块时，把分组中的数据递交给 SCTP 用户。偶联的关闭可以保证所有两端的未发送、发送未证实数据得到发送和证实后再终止偶联。

图4-22 偶联正常关闭消息交互图



偶联的正常关闭步骤如下：

1. 偶联关闭发起端点 A 的 SCTP 用户向 SCTP 发送请求 SHUTDOWN 原因。SCTP 偶联从 ESTABLISHED 状态迁入 SHUTDOWN-PENDING 状态。在这个状态，SCTP 不接受 SCTP 用户在这个偶联上的任何数据发送请求。同时等待端点 A 所有发送未证实的数据得到端点 B 的证实。当所有端点 A 发送未证实数据得到证实，则向端点 B 发送 SHUTDOWN 数据块。端点 A 启动 T2-shutdown 定时器进入 SHUTDOWN-SENT 状态。启动 T2-shutdown 定时器的目的是等待端点 B 发回的 SHUTDOWN-ACK 数据块，如果定时器超时，则端点 A 必须重新发送 SHUTDOWN 数据块。
2. 端点 B 收到 SHUTDOWN 消息后，进入 SHOUTDOWN－RECEIVED 状态，不再接收从 SCTP 用户发来的新数据，并且检查数据块的累积 TSN ACK 字段，验证所有未完成的 DATA 数据块已经被 SHUTDOWN 的发送方接收。当端点 B 所有未发送数据和发送未证实数据得到发送和证实后，发送 SHUTDOWN ACK 数据块并启动本端 T2-SHUTDOWN 定时器，并且进入 SHUTDOWN-ACK-SENT 状态。如果定时器超时了，端点 B 则重新发送 SHUTDOWN ACK 数据块。
3. 端点 A 收到 SHUTDOWN ACK 消息后，停止 T2-shutdown 定时器，并且向端点 B 发送 SHUTDOWN COMPLETE 数据块，并清除偶联的所有记录。端点 B 收到 SHUTDOWN COMPLETE 数据块后，验证是否处于 SHUTDOWN-ACK-SENT 状态。如果不是处于该状态，则丢弃该数据块；如果端点处于 SHUTDOWN-ACK-SENT 状态，端点 B 则停止 T2-shutdown 定时器并清除偶联的所有记录，进入 CLOSED 状态。