# SCTP协议介绍

# ★提问：什么是SCTP协议？

回答：SCTP协议是STREAM CONTROL TRANSMISSION PROTOCOL 即流控制传输协议的缩写。

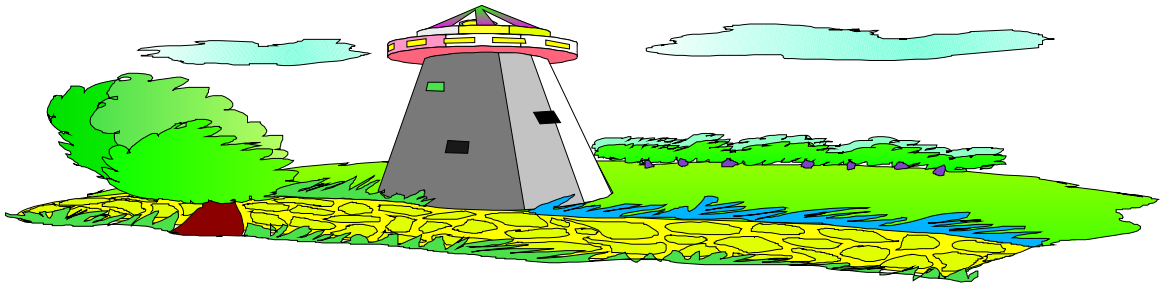
我们可以这样来定义SCTP协议：

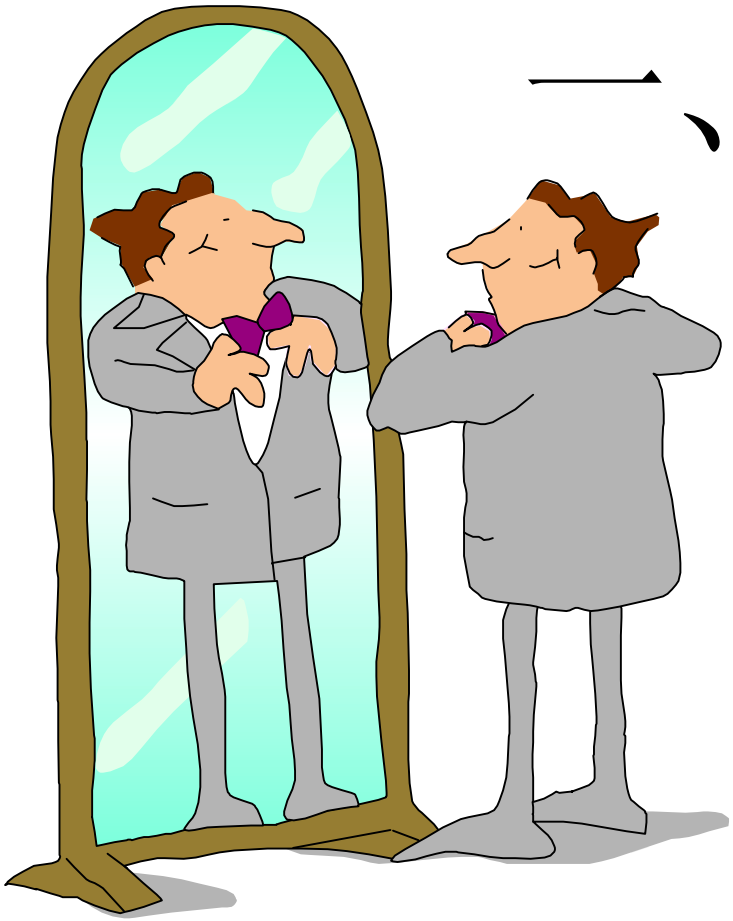
SCTP是基于提供不可靠传输业务的协议（如IP）之上的可靠的数据报传输协议

# ★再问：传输协议？有了TCP和UDP还要SCTP干什么？

回答：因为SCTP最初是设计来在IP网中传送信令的所以：
1、有服务的需求
TCP----提供面向连接的可靠的数据流传输
UDP----提供无连接的不可靠的数据包传输
SCTP--提供面向连接的可靠的数据包传输
2、TCP有固有的缺陷
支持多归属（Multi-homing）比较困难
易受拒绝服务攻击（DoS） 容易出现行头阻塞
仅能支持字节流传输 实时性差

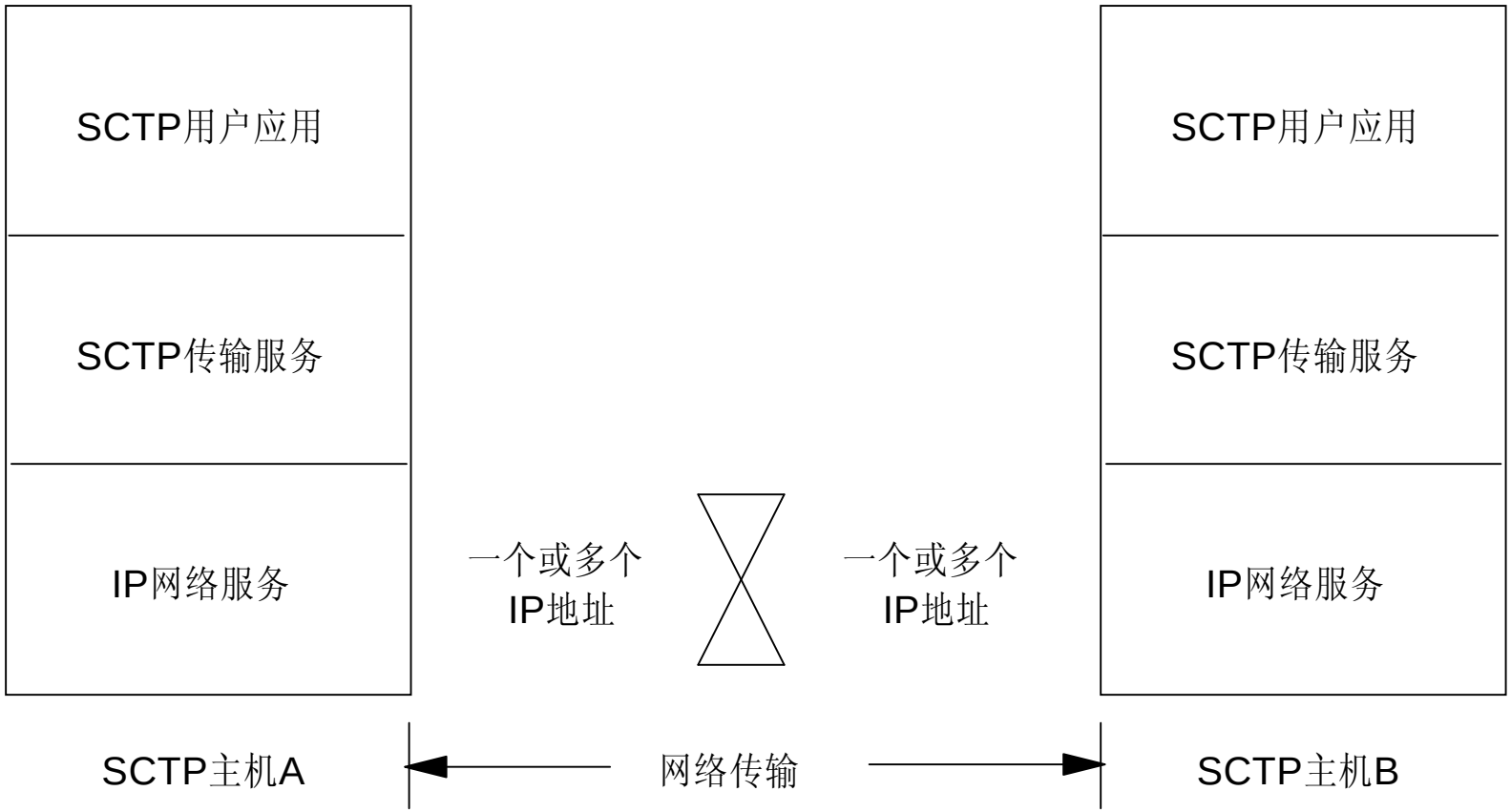
- 正是因为SCTP相对于TCP有了许多优点：
  - 支持数据报传递，无须上层实现数据定界功能
  - 实时性好
  - 安全性好
  - 避免了行头阻塞
  - 支持多归属（MUTI-HOMING）
- SCTP相对于TCP就更适合对实时性，安全性，可靠性要求高的信令传输，并有了更广阔的前景





# 一、SCTP快照

# SCTP快照：SCTP的协议地位



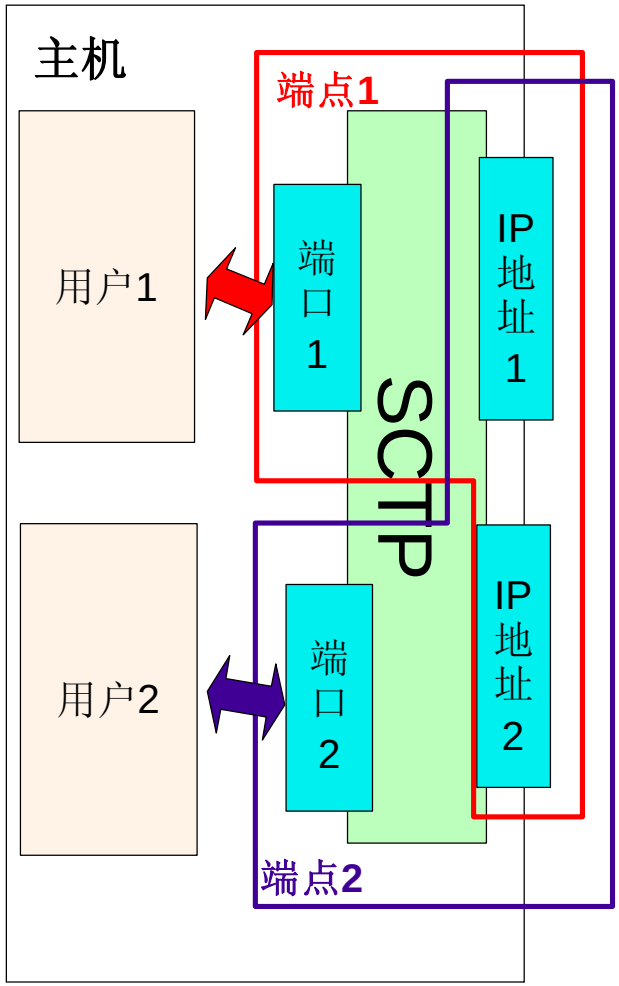

结论：无论基于OSI的7层模型，还是TCP/IP协议族的4层模型（网络接口层，互联层，传输层，应用层），SCTP都处于传输层的地位。

# SCTP快照：SCTP的基本术语（一）

- 概念一：**IP地址和传输地址：**
  - 很简单，SCTP传输地址就是一个IP地址加一个SCTP端口号。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标识了一个传输地址，而10.105.28.92和1023则标识了另外一个传输地址。同样的，10.105.28.93和1023也组成了一个和它们不同的传输地址

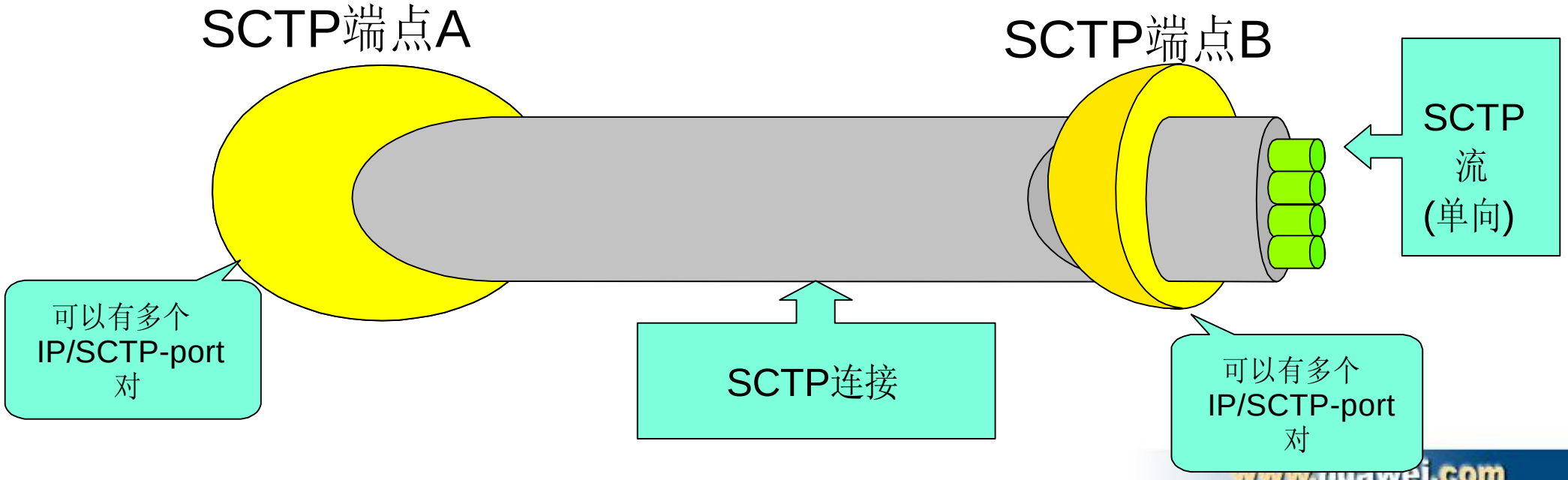
# SCTP快照：SCTP的基本术语（二）

## ■ 概念二：主机和端点：




- **主机**的概念很实在，一个看得见，摸得着的计算机，配有一个或多个IP地址，就构成我们说的主机。是一个典型的物理实体
- **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。一个典型的逻辑实体。
- SCTP端点由一组具有相同SCTP端口号的传输地址标识
- SCTP协议规定两个端点之间能且仅能建立一条连接。
- 所以，SCTP端点可能有多个传输地址，但是这些传输地址有唯一的端口号。

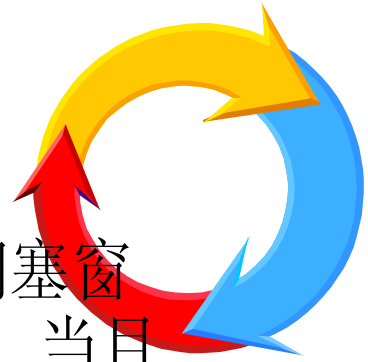
- 概念三：**连接（偶联）和流**
  - 连接就是两个SCTP端点通过SCTP协议规定的4步握手机制建立起来的进行数据传递的逻辑联系或者说通道
  - “流”就是一条SCTP连接中，从一个端点到另一个端点的单向逻辑通道。



- 概念四：**TSN**和**SSN**（传输顺序号和流顺序号）
  - **TSN**（Tranmission Sequence Number），传输顺序号，在SCTP一个连接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（连接建立时随机生成的）的32位顺序号，以便对端收到时进行确认。TSN是基于连接维护的
  - **SSN**（Stream Sequence Number）流顺序号，在SCTP一个连接的每个输出流内，为本端在这个流中发送的每个数据块顺序分配一个16位顺序号，以便保证流内的顺序传递。（TCP没有SSN，于是TCP相当于只有一个输出流的SCTP）。SSN是基于流维护的。

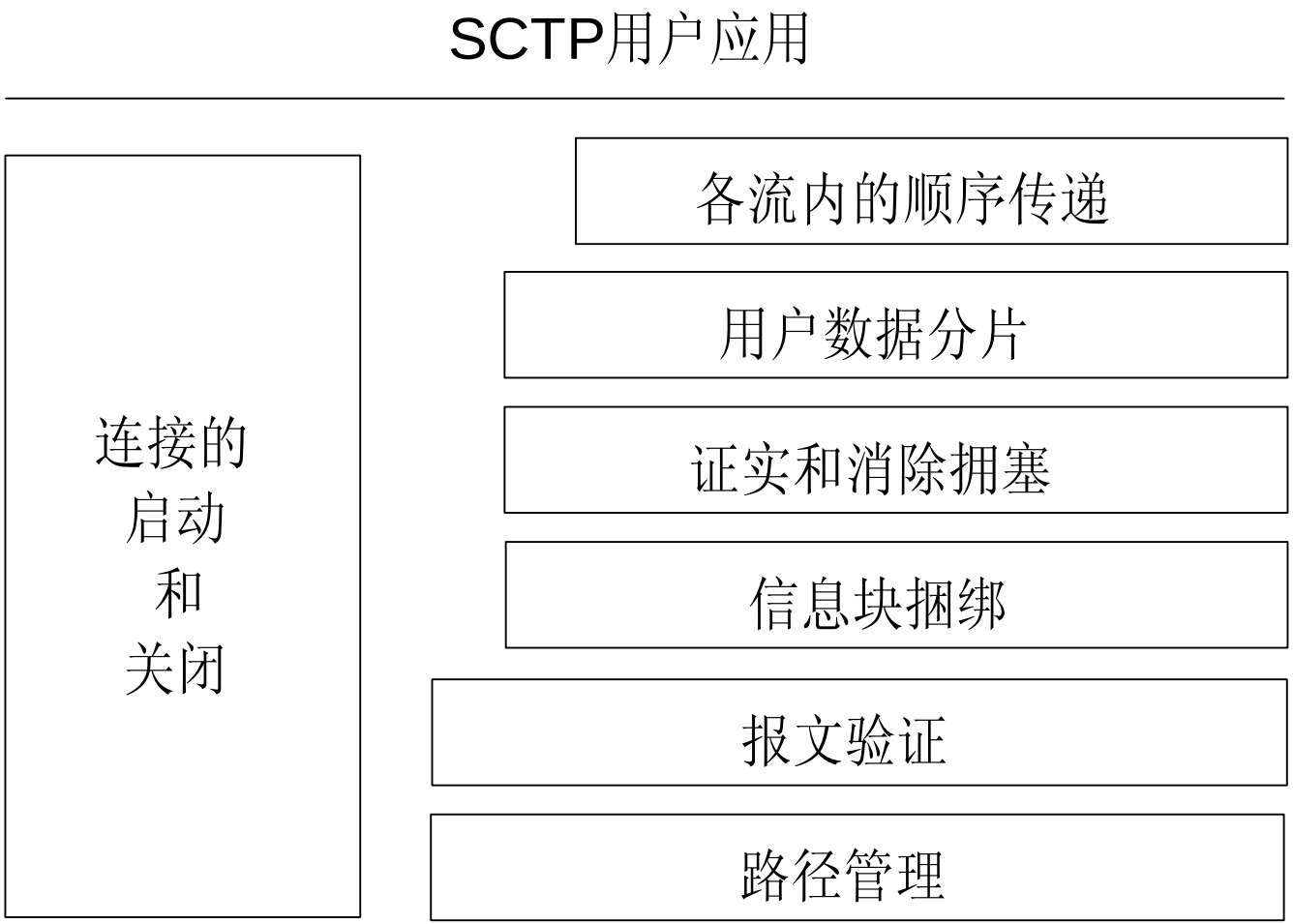
- TSN和SSN的分配是相互独立的，举个例子来说，一个连接的端点A使用两个输出流和端点B相连。有数据块A,B,C,D要发送，发送顺序是这样的：A走流1，B走流2，C走流1，D走流2，而且D太长，被分成了两片（D1，D2）。那么这5个数据块的TSN和SSN分别是：
-

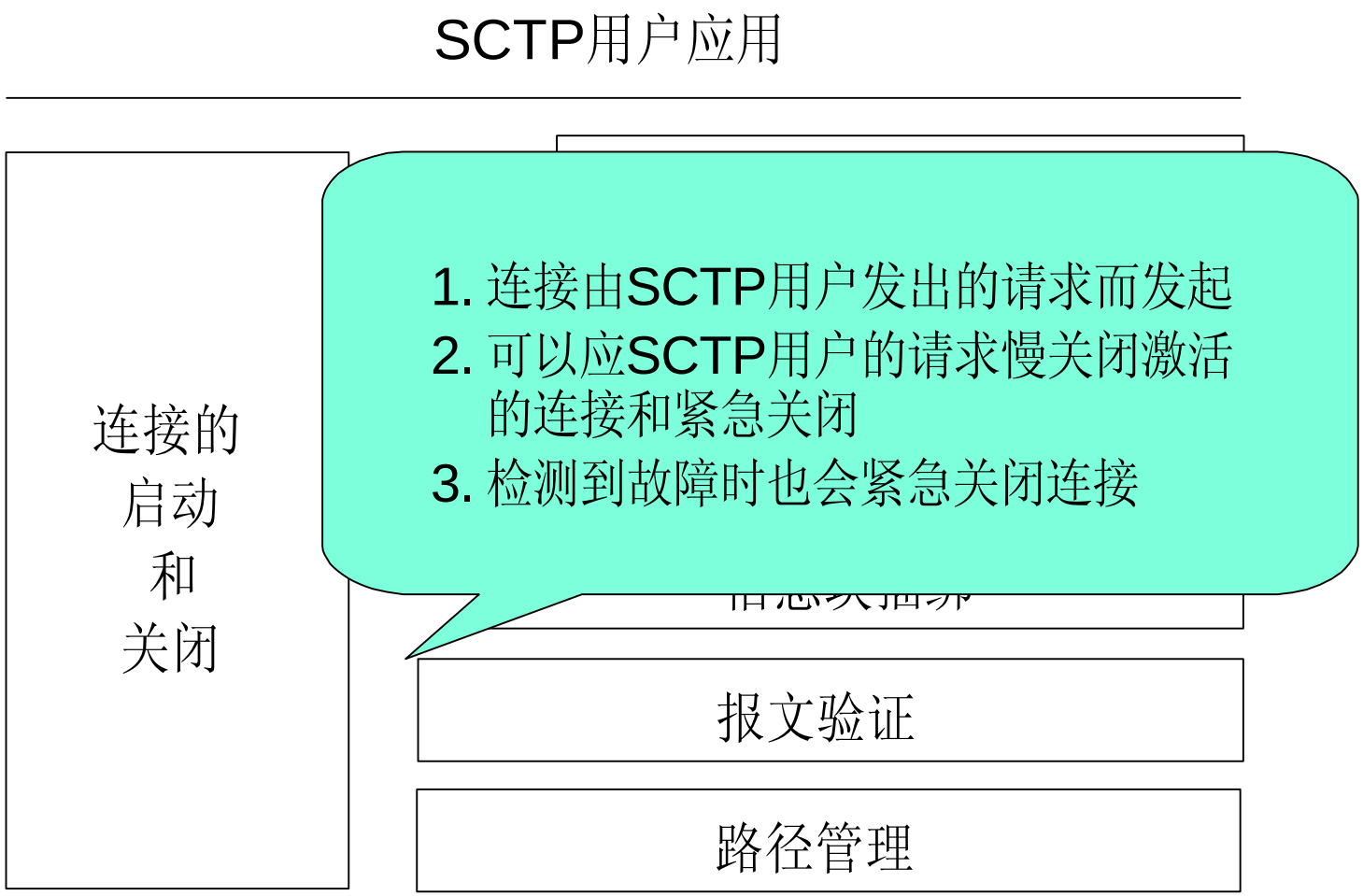
| 数据 | TSN | SSN |
|---|---|---|
| A | 1 | 1 |
| B | 2 | 1 |
| C | 3 | 2 |
| D1 | 4 | 2 |
| D2 | 5 | 2 |



- 概念五：**CWND和RWND**
  - CWND：拥塞窗口。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
  - RWND: 接收窗口。RWND用来描述一个连接对端的接收缓冲区大小。连接建立过程中，双方会交换彼此的初始RWND。RWND会根据数据发送，证实的情况即时的变化。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

- SCTP协议的基本功能如图

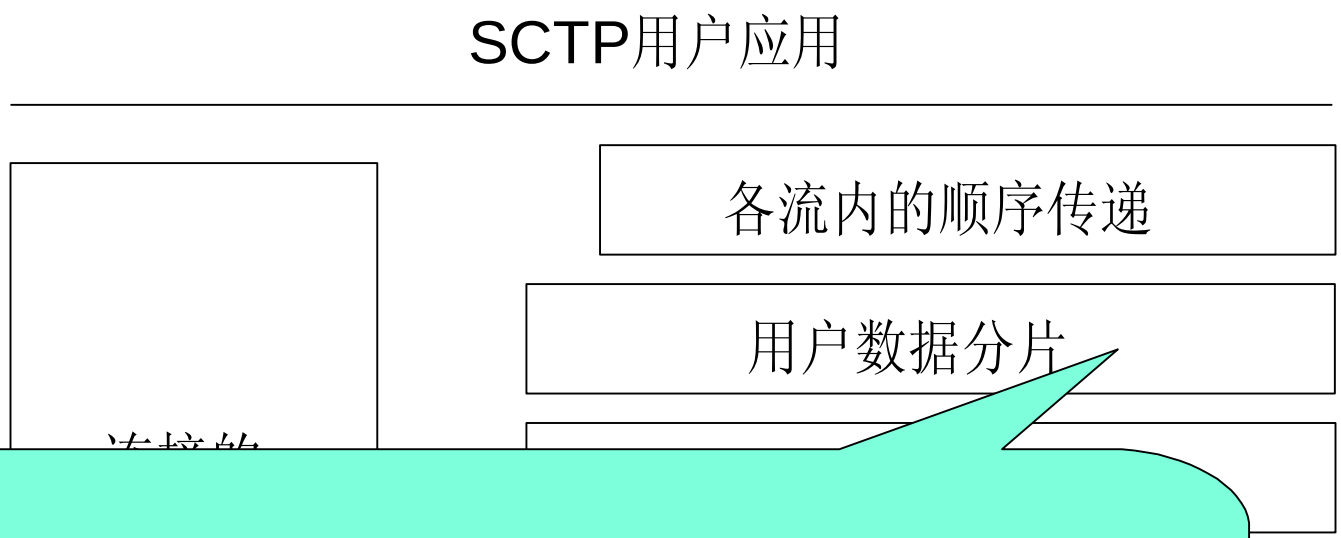




# SCTP快照：SCTP协议基本功能（3）

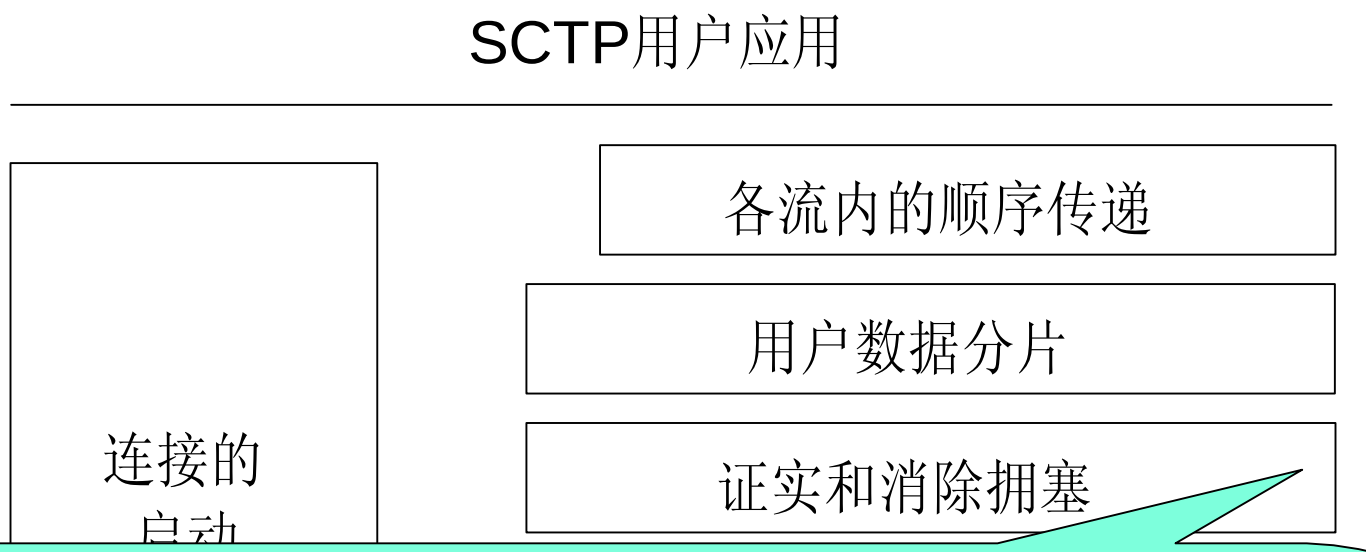

1. SCTP用户在连接建立时可以指定在连接内打开流的数目，这个数目要和远端进行协商
2. 用户数据报具体的流联系在一起（SEND和RECEIVE原语）。在SCTP内部为SCTP用户递交的每个用户数据报分配一个流顺序号，在接收端，SCTP确保在给定的流内把数据报顺序地传递到SCTP用户
3. 当一个流内因乱序或数据报丢失进行数据报等待时，其它流内的数据报可以顺序传递给SCTP用户而不受影响。

# SCTP快照：SCTP协议基本功能（4）



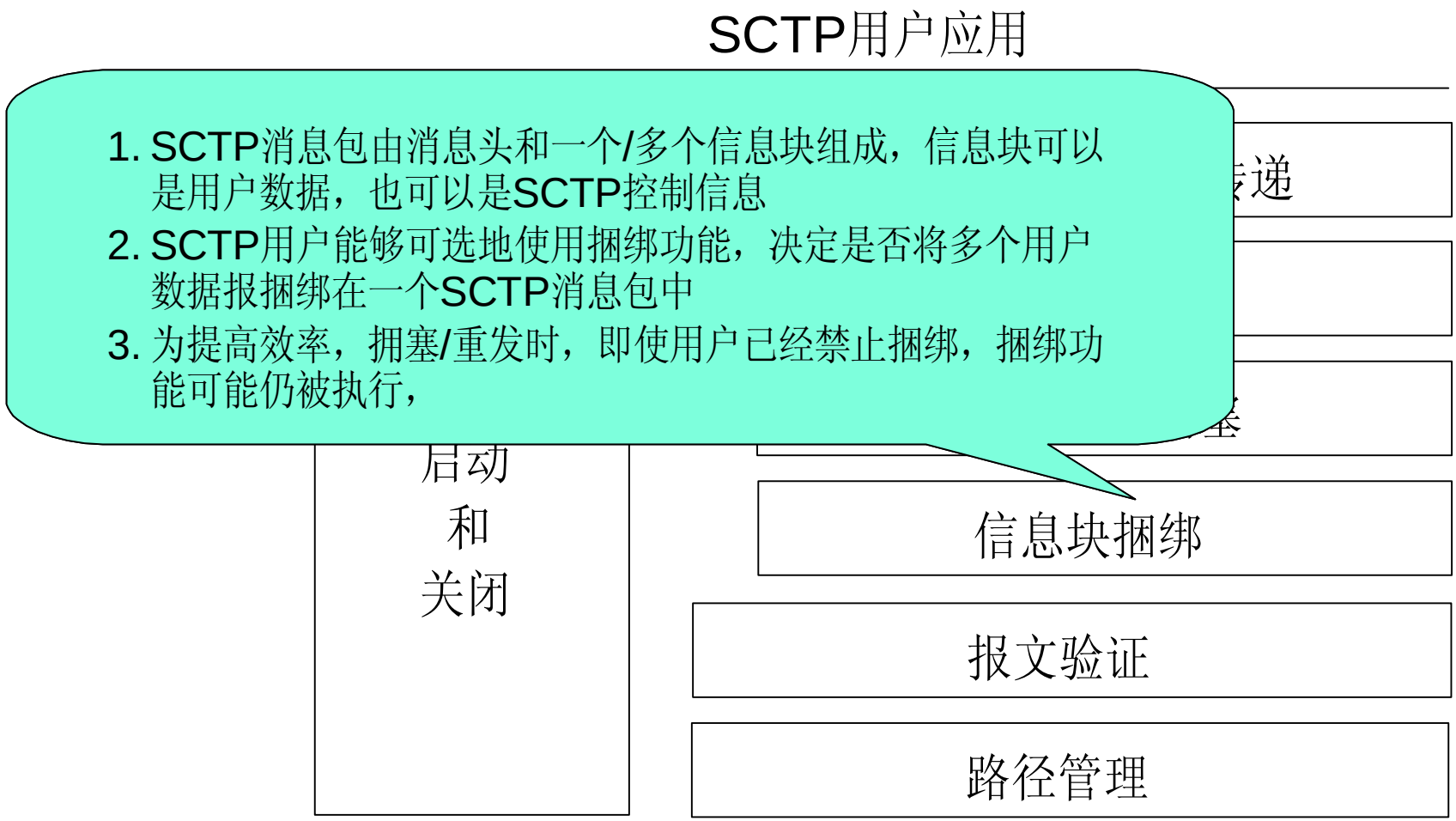

1. 在发送端，SCTP可以对大的用户数据报进行分片以确保SCTP数据报传递到低层时适合路径MTU
2. 在接收端，SCTP将分片重组为完整的用户数据报，然后传递给SCTP用户

# SCTP快照：SCTP协议基本功能（5）

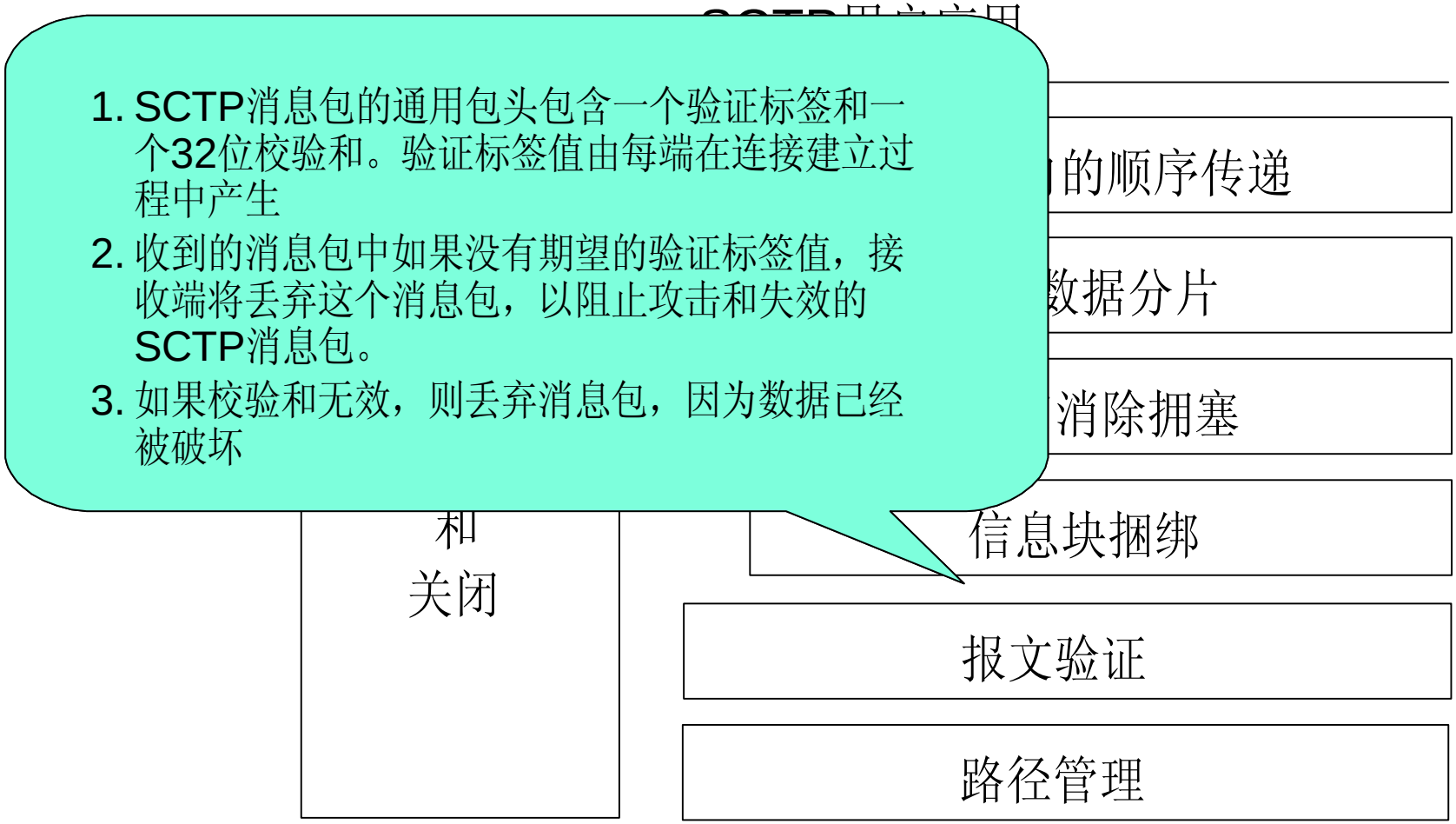



1. SCTP在将数据（数据分片或未分片的用户数据报）发送给底层之前顺序地为之分配一个发送顺序号（TSN）
2. 正常情况下，数据接收端通过延迟证实机制证实数据块，特殊情况下使用立即证实
3. 接收端用选择证实块（SACK）证实所有收到的数据块，即使其中顺序上出现了缝隙
4. SCTP采用极为类似TCP的机制来进行拥塞控制

# SCTP快照：SCTP协议基本功能（6）

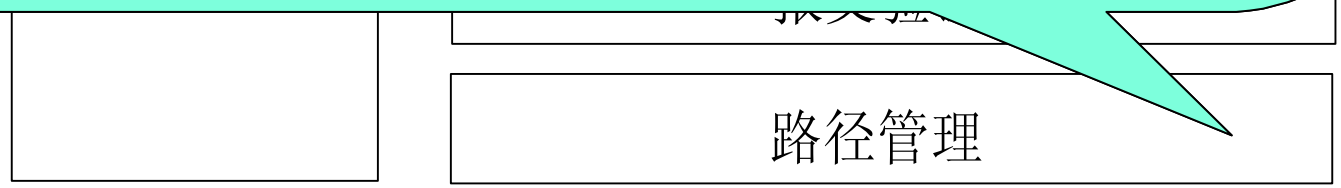


# SCTP快照：SCTP协议基本功能（7）

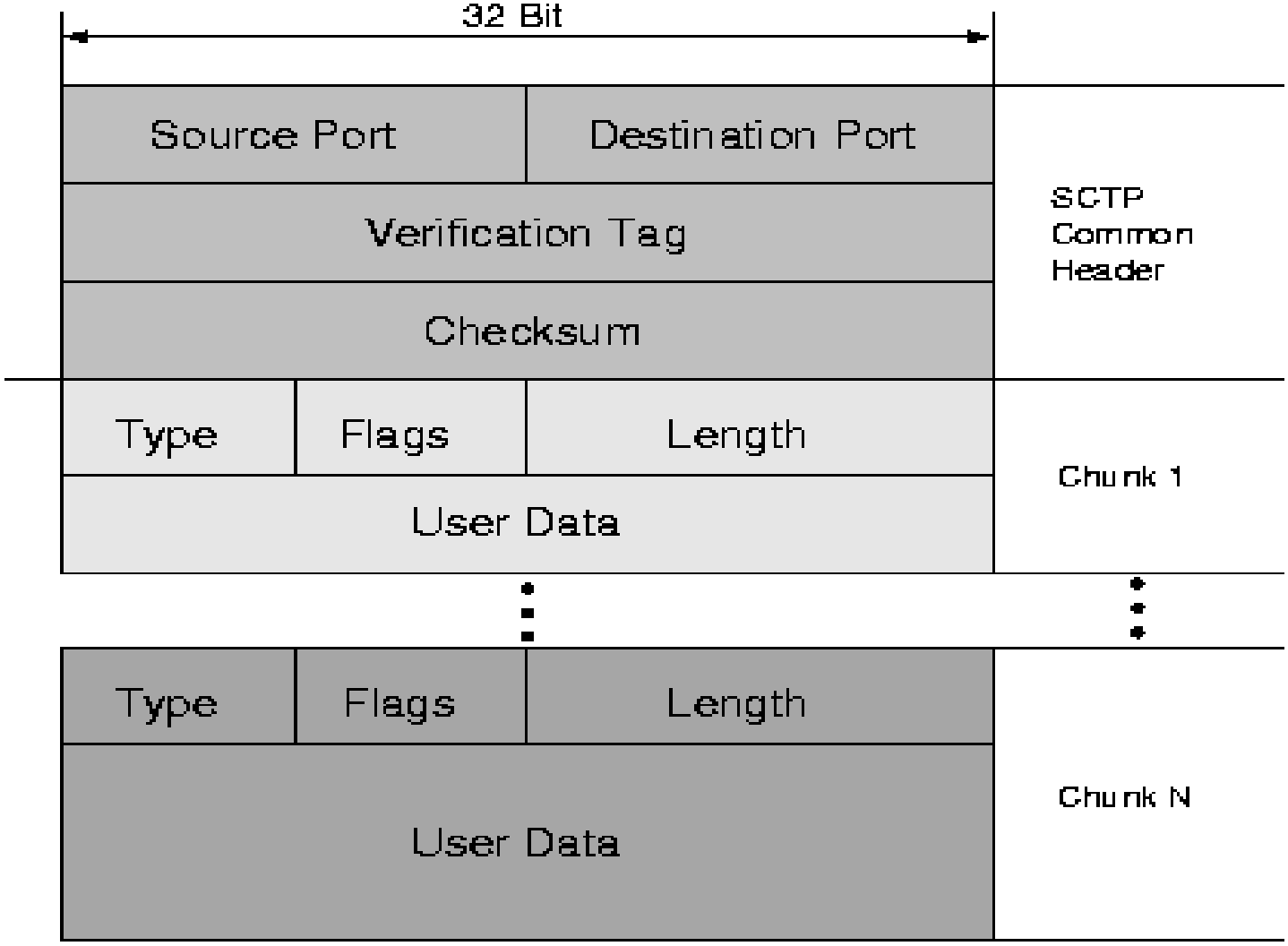


# SCTP快照：SCTP协议基本功能（8）

1. SCTP路径管理功能基于SCTP用户的指定和当前符合条件目的地址集合中各地址可达性的情况来为每个输出的用户数据报选择目的传输地址
2. 路径管理功能在业务流不能充分提供信息时通过心跳功能监视路径的可达性。如果远端传输地址可达性变化，SCTP要向用户进行通报
3. 路径管理功能在连接建立阶段负责向远端报告本端符合条件的传输地址集合，并把回应的远端传输地址报告给SCTP用户
4. 连接建立后，连接的两端都要指定一个首选路径，用于SCTP消息包的正常发送
5. 接收端的路径管理功能负责验证消息包所属的连接是否存在，为消息包后续处理作准备



- SCTP的消息结构就是通用头（COMMON HEADER）加信息块（CHUNK）

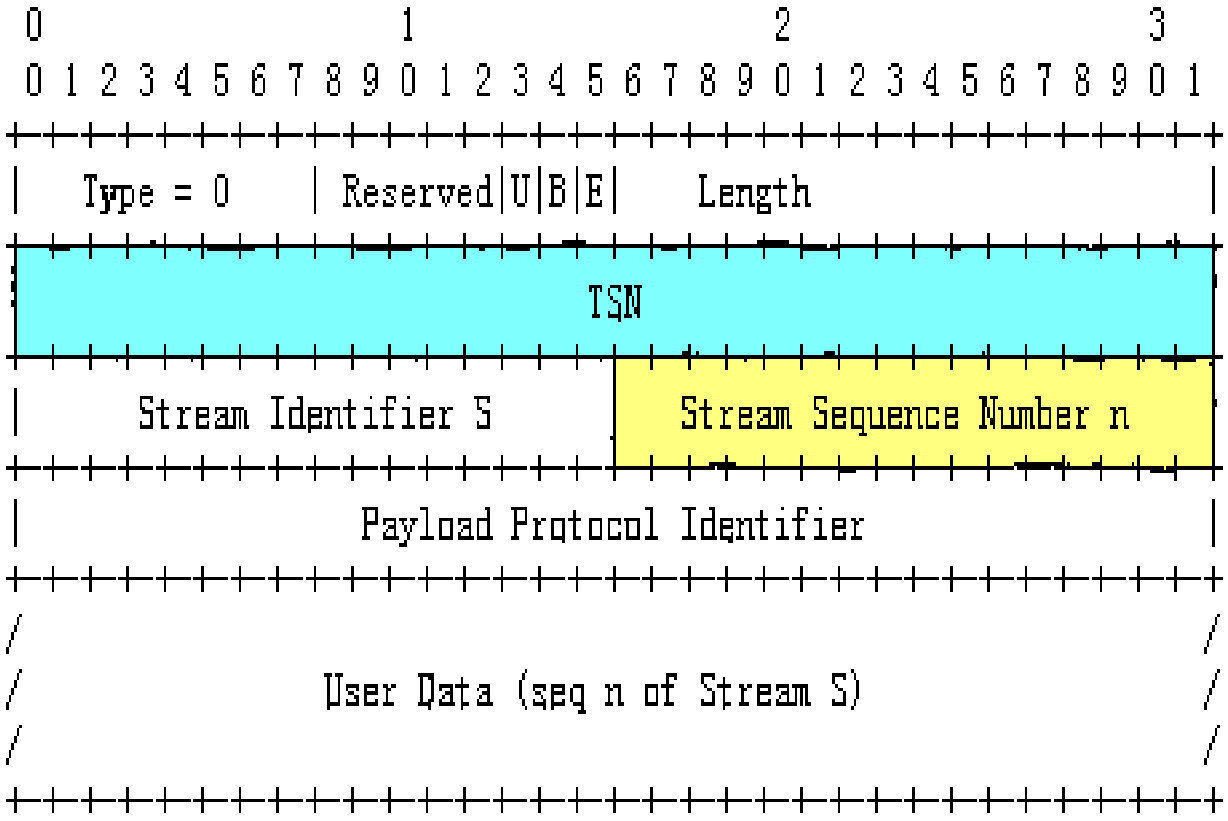


- SCTP信息块类型：

| 信息块标识值 | 信息块类型 | 信息块标识值 | 信息块类型 |
|---|---|---|---|
| 0 | DATA | 12 | ECNE |
| 1 | INIT | 13 | CWR |
| 2 | INIT ACK | 14 | SHUTDOWN COMPLETE |
| 3 | SACK | 15~62 | reserved for IETF |
| 4 | HEARTBEAT | 63 | 块扩展 |
| 5 | HEARTBEAT ACK | 64~126 | reserved for IETF |
| 6 | ABORT | 127 | 块扩展 |
| 7 | SHUTDOWN | 128~190 | reserved for IETF |
| 8 | SHUTDOWN ACK | 191 | 块扩展 |
| 9 | ERROR | 192~254 | reserved for IETF |
| 10 | COOKIE ECHO | 255 | 块扩展 |
| 11 | COOKIE ACK | | |

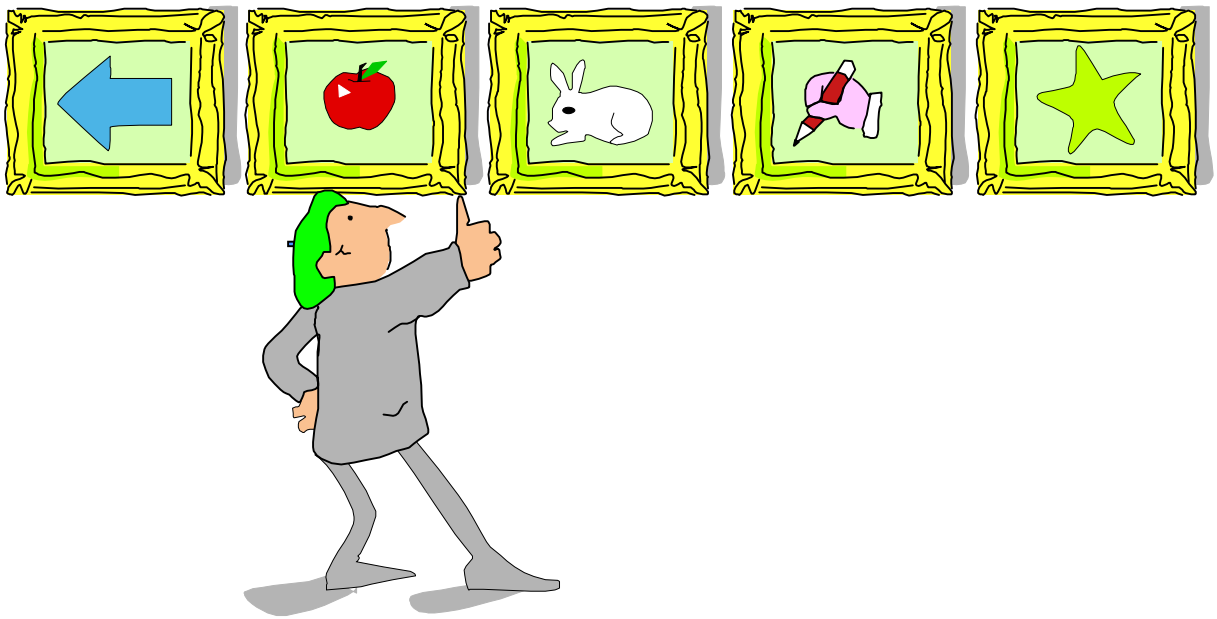
# SCTP快照：SCTP消息结构（3）

## 业务数据块（DATA CHUNK）的结构



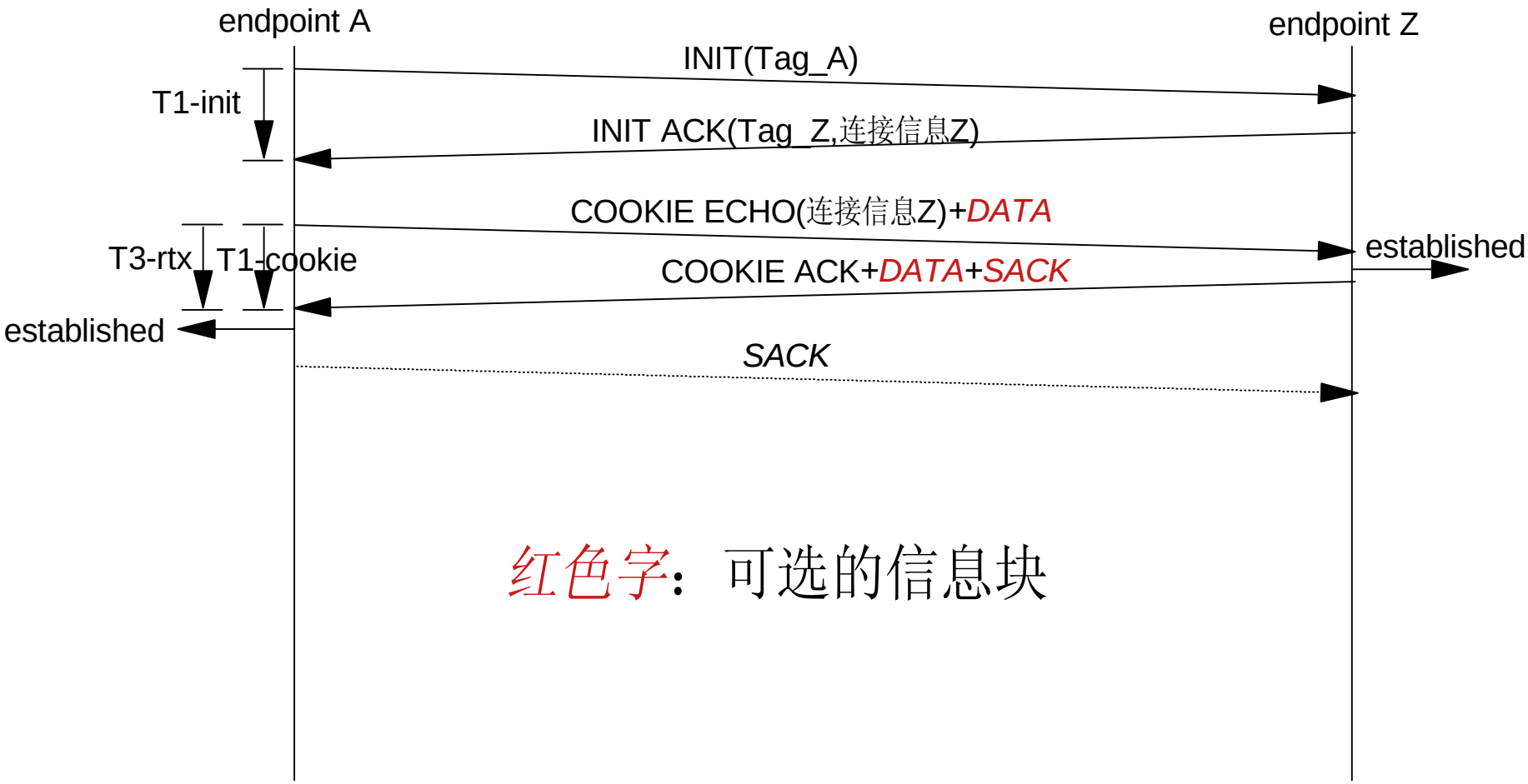

- 注意信息块和数据报的区别和联系：
  - 数据报由通用头和一个或多个信息块组成：
  - 多个信息块放在一个数据报中传输实现了SCTP的捆绑功能。有利于节省带宽。
  - SCTP数据发送，接收，证实等所有的控制都是针对DATA信息块的，也就是说，TSN，SSN编号都是对信息块进行编号，而不是对数据报

# 二、SCTP协议过程



# SCTP协议过程：连接建立过程（1）

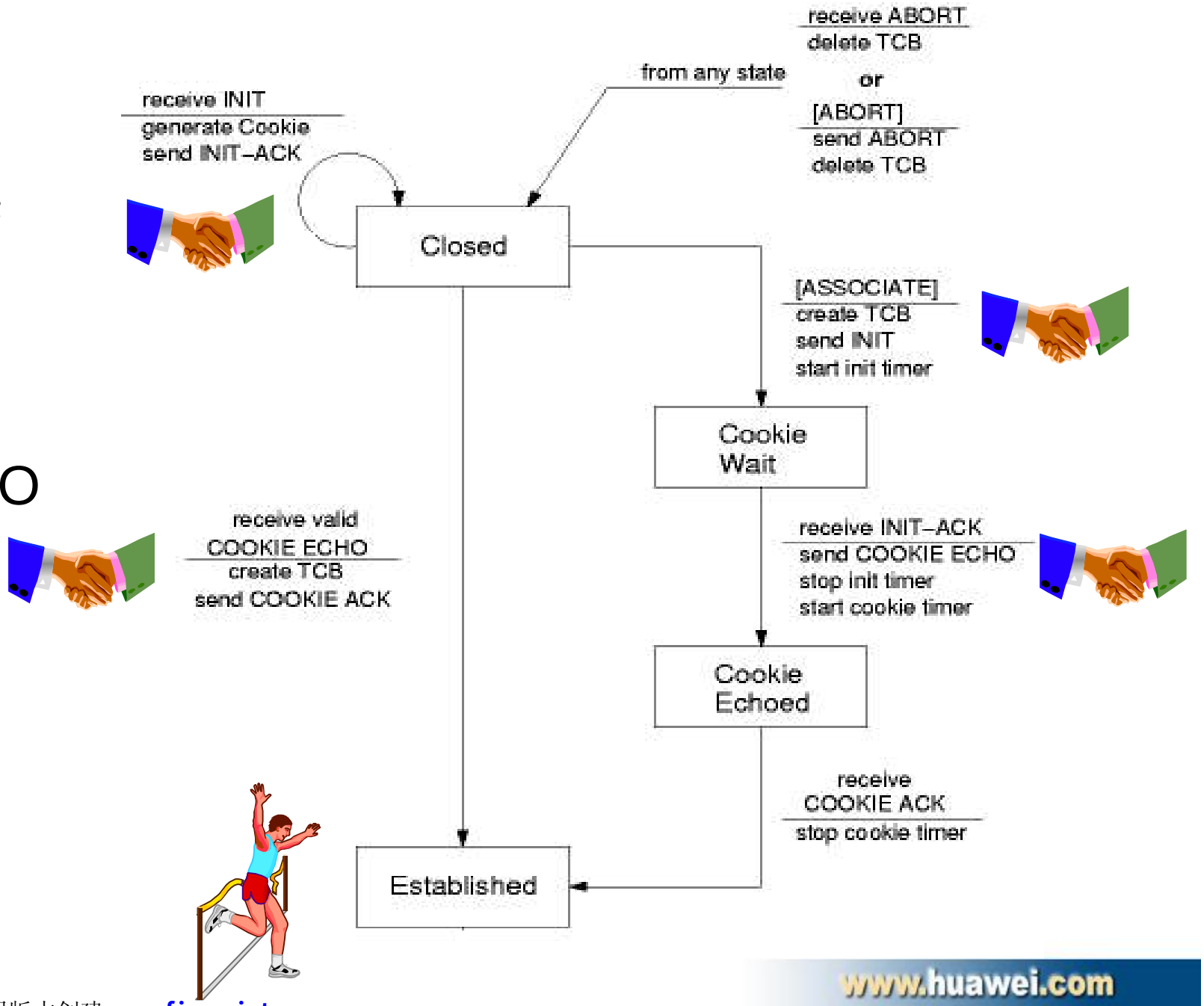
- 连接建立过程消息交互图示：




红色字：可选的信息块

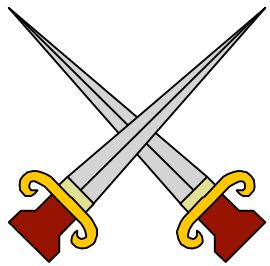
# SCTP协议过程：连接建立过程（2）

- 连接建立过程状态迁移图示：
- 四步握手：
- INIT
- INIT-ACK
- COOKIE-ECHO
- COOKIE-ACK



# SCTP协议过程：连接建立过程（3）

- 连接建立过程中的奥妙（一）：
  - 连接建立的发起端，在连接建立过程中有3次状态迁移，两个中间状态（CLOSED－COOKIE-WAIT－COOKIE-ECHOED－ESTABISHED）；而连接的接受端，只有一次状态迁移（CLOLSED－ESTABLISHED）没有中间状态。
  - 连接发起端要在发起连接前生成一个TCB来放置连接相关的信息。而连接接受端在收到INIT消息后会生成一个临时TCB，并作为COOKIE的一部分放入INIT-ACK中发走，随后删除该TCB，即在连接建立完成前不为连接建立过程保持任何本端资源。
    - 可防止“ 拒绝服务”攻击

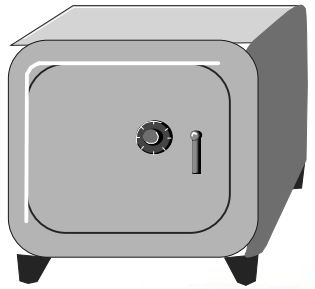


# 拒绝服务攻击

- 所谓拒绝服务攻击，就是利用一些系统TCP协议实现的漏洞，通过在连接建立的握手消息中的标识位设置不规范的值，使得服务器的这个连接的状态机长时间或永久处在中间状态，从而使服务器的监听队列被这样的“连接”占满，而无法为其他正常的连接服务。
- SCTP的连接接受端，在连接建立过程中，没有中间状态，也不分配资源，所以可以防止拒绝服务攻击。

# SCTP协议过程：连接建立过程（4）

- 连接建立过程中的奥妙（二）：
  - COOKIE机制：COOKIE由两部分组成。一部分就是连接接受端生成的临时TCB中的必要信息（包含COOKIE的生命周期，和时间戳），一部份就是上面这些内容和一个本端的密钥按照RFC2401的算法计算生成的一个32位的MAC摘要。
  - COOKIE通过INIT-ACK发到对端，又会通过COOKIE-ECHO送回来，这时会进行COOKIE验证，包括进行MAC摘要的验证和生命期的验证。通过后方认为是有效的COOKIE，其中的连接信息将被作为接受端TCB的主要部分。
    - 可防止IP地址化装攻击



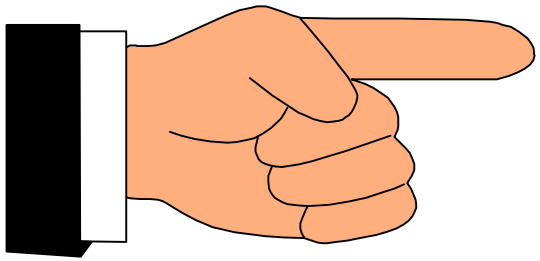
# IP地址化装攻击

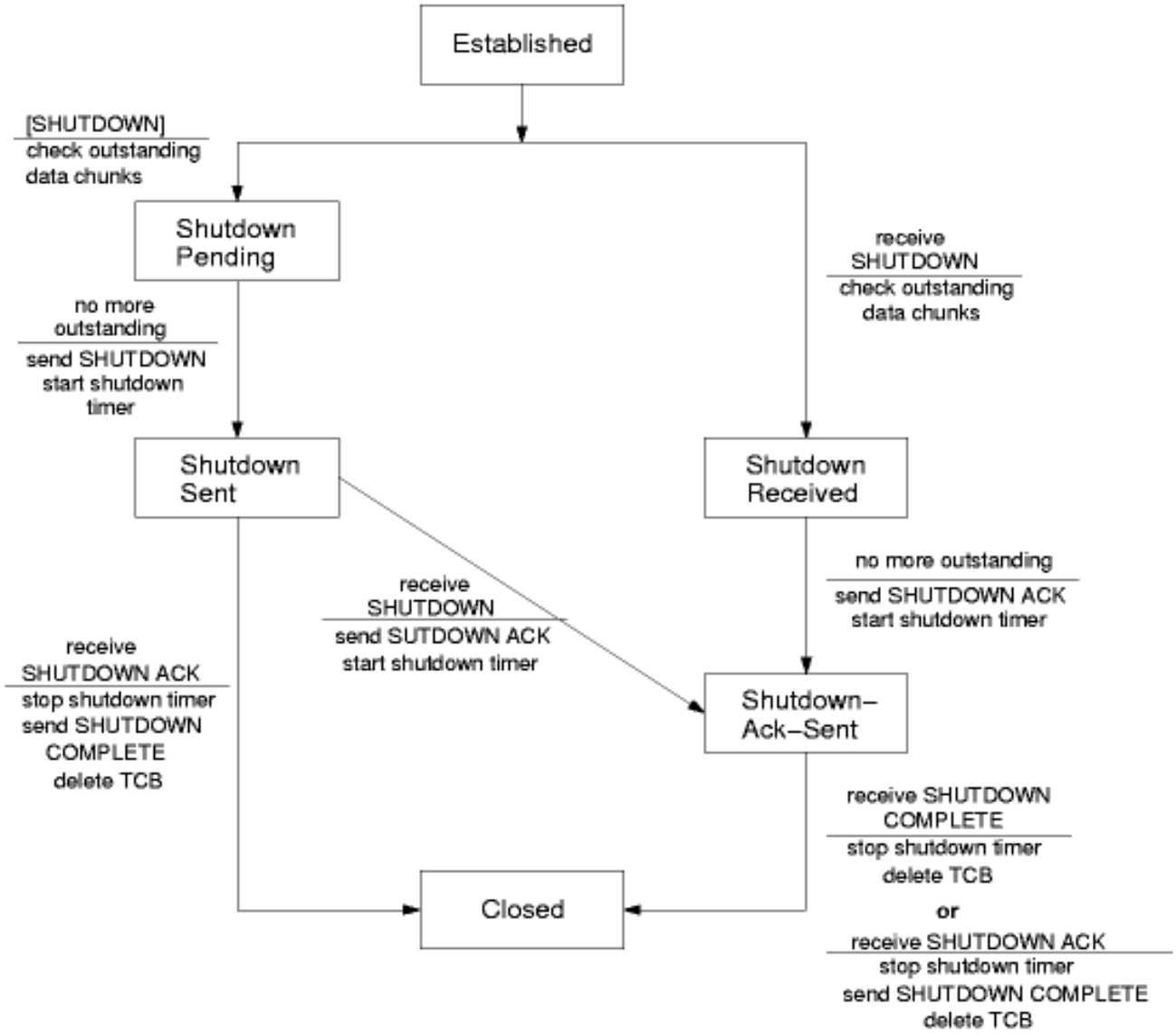
- 所谓IP地址化装攻击，就是一个恶意攻击者A通过冒充一个合法的客户端B的IP地址而和服务器C建立连接。因为TCP的连接过程中没有设置任何的信息加密和校验，所以A无须收到任何C发给B的消息就可以完成和C的连接建立过程。
- SCTP由于使用了COOKIE机制，服务器使用本端密钥和COOKIE一起生成了MAC摘要。有了信息加密机制。假设A冒充B向C发了一个INIT消息，C就会向B回一个INIT-ACK，里面就有COOKIE和摘要。如果A和B不在一个局域网里，则无法截获这个消息。由于A不知道C的本端密钥，则无法生成出一样的MAC摘要。那么A如果试图发COOKIE-ECHO消息去继续这个过程，则将通不过C的校验而失败。因此SCTP可以防止IP地址化装攻击

- 连接终止过程分为两种：
  - GRACEFUL和UNGRACEFUL。前者保证终止时数据的安全，后者不保证
- UNGRACEFUL终止过程非常简单，发起端发送一个ABORT消息到对端后，立即删除连接TCB。对端收到这个ABORT消息后也立即删除连接TCB（何时删除TCB是基于实现的，某些实现可能需要上层的参与，但即使不删，这个连接的状态已经是关闭的了）。由于有Vertification Tag，一个恶意攻击者在不能截获消息的情况下是无法得知其他主机SCTP连接的两端的TAG值，因此也就无法通过发送合法的ABORT消息来试图破坏一个已经建立的连接

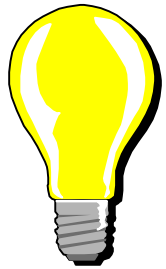
# SCTP协议过程：连接终止过程（2）

- GRACEFUL 连接终止过程：





# SCTP协议过程：连接终止过程（3）



- GRACEFUL终止过程中的注意点：
  - 在上层用户发起连接终止时，SCTP并不是立即向对端发送SHUTDOWN消息，而是进入SHUTDOWN-PENDING状态，等待本端所有发送未证实数据得到对端的证实才发送这个消息。
  - 相似的，对端也是在其所有发送未证实数据都得到证实时才发送SHUTDOWN-ACK消息。
  - 发起端在进入SHUTDOWN-PENDING状态，接受端在进入SHUTDOWN-REVD状态，均拒绝上层用户的数据发送请求。

# SCTP协议过程：数据传输过程（1）

- 数据传输一般发生在连接建立好之后（连接建立的过程中，某些步骤也允许捎带数据）。SCTP数据传输的特点就是：
  - *窗口流控机制*
  - *延迟选择证实*
  - *超时重传*
  - *多归属*
- 实际上除了多归属特性和选择证实，其他特性都和TCP很相似

# SCTP协议过程：数据传输过程（2）

- **窗口发送机制**：如前所述，SCTP发送使用两种窗口。一种是拥塞窗口（CWND），一种是对端接收缓冲区窗口（RWND）。前者对每个目的地址维护，后者对每个连接维护。前者描述的是：对一个传输路径，当前允许传输多大的数据，而不会拥塞。后者描述的是：对连接对端而言，当前可以接收多大的数据而不会丢弃数据。这是描述两个不同的对象，因此需要两个窗口来共同制约发送。
  - 首先，如果RWND显示对端的接收缓冲区不能接收数据（如RWND=0），则不能向对端发送数据。但是如果当前没有发送未证实的数据。则可以发送一个数据（如果CWND允许的话）。这样可防止死锁
  - 在试图向一个地址发送数据时，如果当前在这个地址上的发送未证实数据已经达到或超过CWND的限制，则不能向这个地址发送数据
  - 发送新数据前，必须将标记重传的数据先发送出去。即重发数据优先，这也基本上是可靠传输协议的共性

# SCTP协议过程：数据传输过程（3）

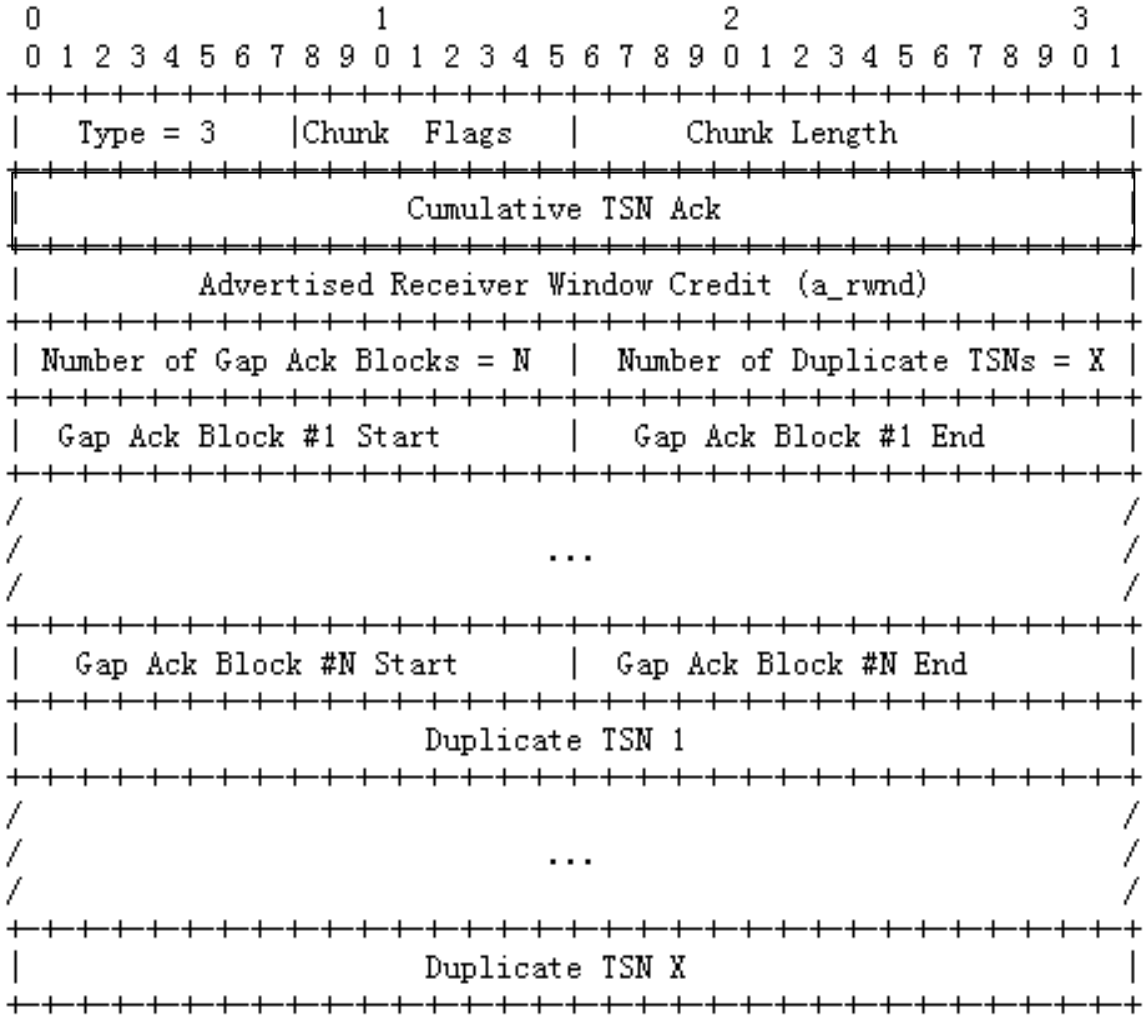
- SCTP传输的实时性：
  - 捆绑不是必须的，正常情况下，数据块会立即被发走，SCTP的捆绑多发生在重传的情况下。这和TCP协议需要PUMP调用来保证实时是不同的。
  - SCTP可以设置数据块是否需要保序，如果不需要，接收端收到这个数据会立即上报。
  - SCTP的顺序传递在流内实现，如果一个流内收到的数据乱序，则该流必须等待数据顺序回复才能上报，但是其他流不受任何影响。即将TCP的行头阻塞限制在一个流里面。
  - 这样SCTP从发送和接收两方面保证了数据的实时传递

# SCTP协议过程：数据传输过程（4）

- **延迟选择证实机制：实际上可以分两方面说，延迟证实和选择证实：**
  - **延迟证实：**是相对于立即证实来说的，也就是说，SCTP连接的一端，收到一个数据报时，不是立即向对端发送一个证实。而是每收到两个数据报（注意是数据报不是数据块，尽管一个数据报可能有好几个数据块），或者一个数据报已经到达200MS而没有证实时向对端发送证实消息。这样可以防止路径上证实消息过多
  - **选择证实：**是相对于顺序证实或者说累积证实而言的。典型的采用累积证实的协议是TCP，如TCP一端收到数据1，2，4，5，7，8，9。它只能在消息的ACK字段里面填2，而不能证实后面的几个数据。相反的，SCTP就可以。

# SCTP协议过程：数据传输过程（5）

- SACK（Selective Acknowledgement）信息块结构：




1、Cumulative TSN Ack：即累积证实的TSN，也就是说没有空隙的最大TSN。如上个例子，就是2。

2、Number of Gap Ack Blocks：即收到的数据序列中存在的缝隙数量。比如上面的例子中，2到4有一个缝隙，5到7有一个缝隙，一共两个。

3、缝隙证实块开始，缝隙证实块结束。继续上面的例子，由于我们有2个缝隙，相应的也有两个这样的证实块。第一个：开始是4，结束是5，第二个：开始是7，结束是9。

4、连缝隙证实块都没有覆盖的数据（也就是掉到缝隙里面的数据），就等于声明没有收到。数据发送端收到这样的SACK以后，要等连续三个后续SACK都证实这个数据没有收到，才重发（也就是说要收到4个这样的SACK）。

# SCTP协议过程：数据传输过程（6）

- **超时重传：**既然有证实，就一定有重传，否则谈不上可靠传输。SCTP对每个目的地址维护一个T3定时器,其规则可简单描述如下：
  - 发送一个数据（包括重传）到一个特定的目的地址时，如果当前没有T3定时器运转，则启动之。如果正在运行则不作操作
  - 收到SACK时，如果证实了所有的数据，则停T3定时器，如果证实了最早的发送未证实数据则重启T3定时器
  - 如果T3定时器超期，就检查当前到这个目的地址的最大路径MTU为多少（比如1500字节），然后将这么多的发送未证实数据块绑定成一个数据报重传到对端，同时启动T3

# SCTP协议过程：数据传输过程（7）

- **多归属：**所谓多归属，就是支持多IP地址，通过实现一个端点有多个IP地址，就可以支持一个端点使用多个物理网口，这样来提高端点的可靠性。支持多归属的基本规则如下：
  - 首先，我们上面看到SCTP的CWND，T3定时器都是针对每个对端传输地址维护的，所以机制上是支持多归属的
  - 第二，SCTP是保守的支持多归属，多归属主要用来保证端点的可靠性，可以具备地址冗余。因此SCTP发送数据时会从对端的多个地址中选一个主用的地址，数据一般都向主用地址发送。
  - 第三，SCTP尽量把证实消息发送到被证实的数据的源地址。但是如果一个证实消息证实多个数据，则无法保证这一点。
  - 第四，当重传时，可能的情况下，将选用一个和发送不同的目的地址进行重传

# SCTP协议过程：路径管理过程（1）

- **端点状态管理：**
  - SCTP的端点状态管理非常简单，就是对对端维护一个计数器，对向这个端点的连续重传次数进行统计（如果对端为多归属，包括所有地址上的连续重传），一旦这个计数器达到了Association.Max.Retrans这个协议参数（可配置）规定的次数，则认为对端不可达。这时，SCTP将连接转入CLOSED状态，并上报。这个计数器在发往对端的一个数据得到证实时清零
- **路径状态管理：**
  - 目前SCTP的路径管理是针对每个对端地址进行的。也就是为每个对端地址维护一个计数器，用以记录在这个地址上发送定时器T3的超时次数，和发送的心跳（下面会讲到）在规定时间内没有收到回应的次数。如果这个次数大于Path.Max.Retrans协议参数，则标识该地址不可达。如果在这个地址上发送的数据收到对端的证实，或者心跳得到心跳证实，则重置这个计数器。

- **心跳机制：**
  - SCTP的心跳和7号里面MTP2的填充信息单元比较类似：SCTP也是在一个目的地址空闲（有一个定时器可以决定是否空闲）的时候向这个地址发送心跳数据块。和MTP2不同的是，SCTP对端在收到这个心跳数据块时要立即回送相应的心跳证实。而发送端如果在一定的时间内收不到这个证实，会象上面描述的，将路径的错误计数器加一。
  - 是否应用心跳机制，心跳发送的间隔都是可以配置的。

# 谢谢！！！