




SCTP: An Overview

Part 2: Protocol Details

Randall Stewart, Cisco Systems
Phill Conrad, University of Delaware

Outline



10h00-11h00	intro	Randy
	overview of SCTP What is SCTP? What are the major features?	Phill
11h15-12h15	SCTP details	Randy
13h15-14h15	details of sockets API (Randy)	Phill or Randy
	open Q and A	Both

Now, Randy with some details...

We are now engaging the hyperdrive...

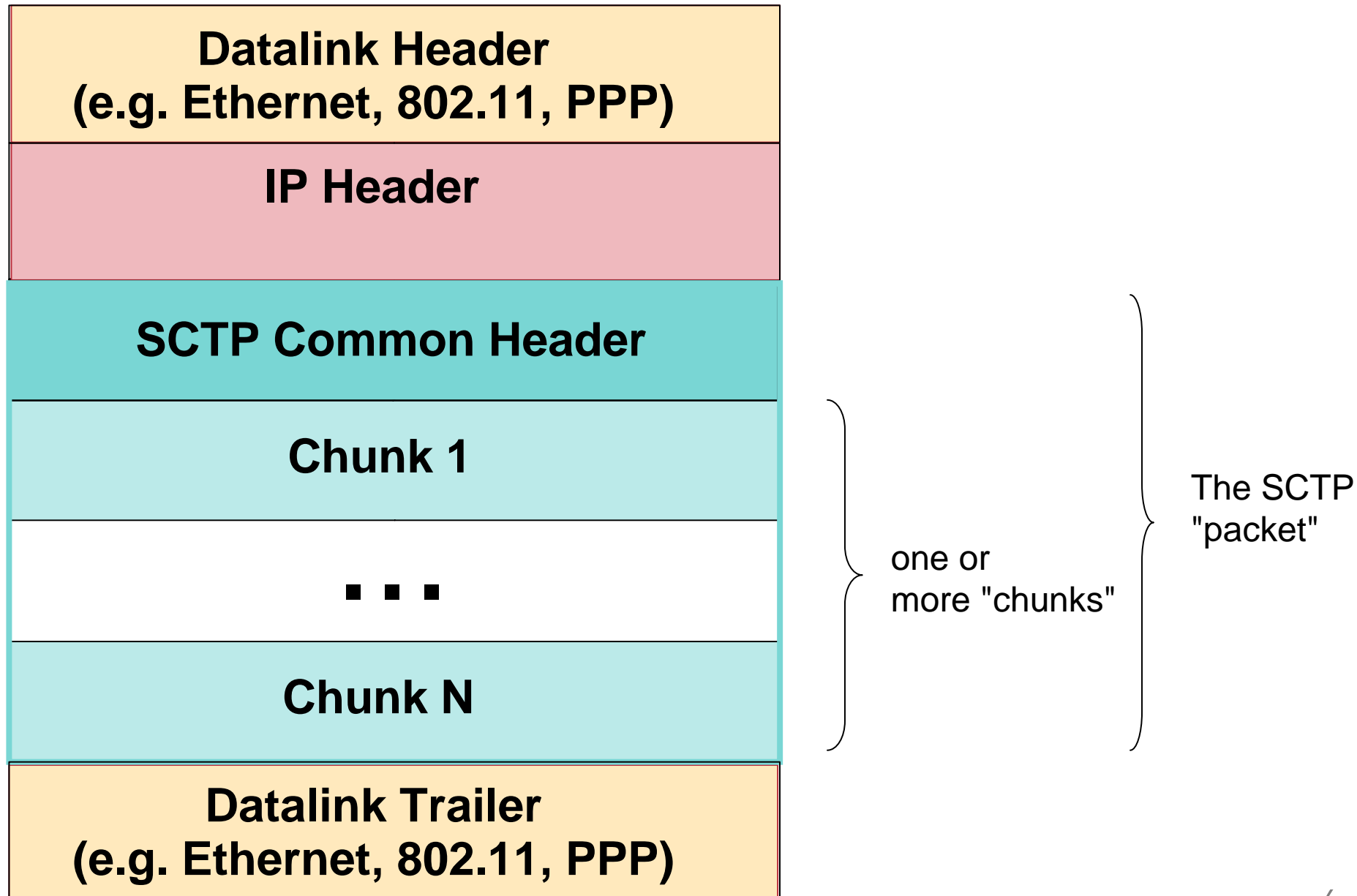
- **packet format: common header, chunks, etc.**
- **connection establishment
(in SCTP-speak, *association* establishment)**
 - Four way handshake, State diagram**
- **data exchange**
 - Streams, Ordered and Unordered Data**
 - Reliable and Unreliable data, Timed Reliability (PR-SCTP)**
- **failure detection and recovery**
 - multi-homing**
 - heartbeats**

Bits and Bytes: SCTP "on the wire"

Bits, Bytes, and Chunks

- We will now turn our attention to some of the on-the-wire bits and bytes of SCTP
- An SCTP packet has a **common header** that appears in each packet, followed by one or more **chunks**
- SCTP chunks use a self-describing Tag-Length-Value (TLV) format
- Note: all figures used are always 32-bits wide

Whole SCTP packet "on the wire"



SCTP Common Header

Source Port	Destination Port
Verification Tag	
CRC-32c Checksum	

- **Source and Destination Port:** 16-bit port values
- **Verification Tag:** 32-bit random value selected by each endpoint in an association during setup

Discriminates between two successive associations

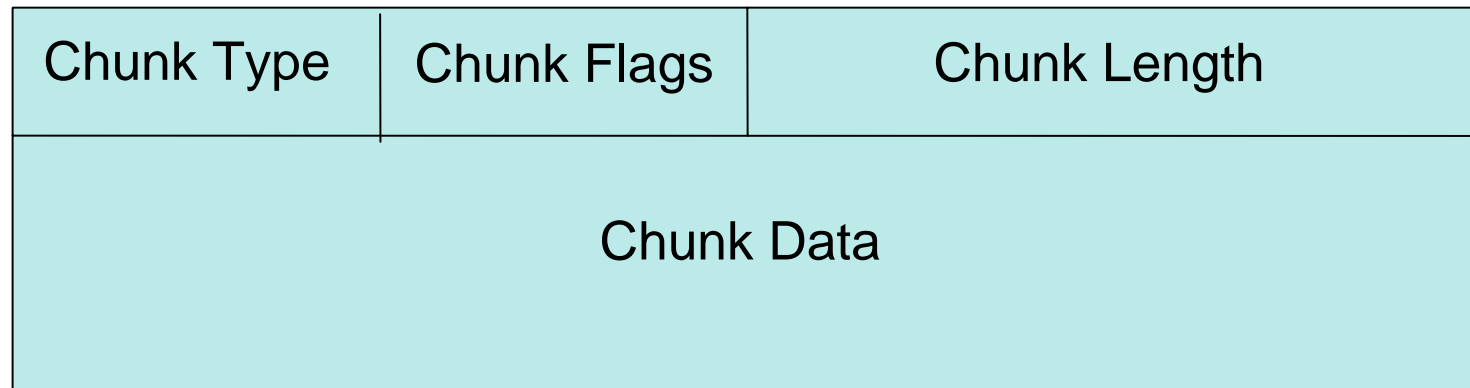
Protection mechanism against blind attackers

- **CRC32c Checksum:** 32-bit CRC covering the entire SCTP packet (SCTP common header and all chunks)

Note that RFC 3309 (CRC32c) supercedes the Adler-32 checksum defined in RFC 2960 (SCTP)

SCTP chunk header

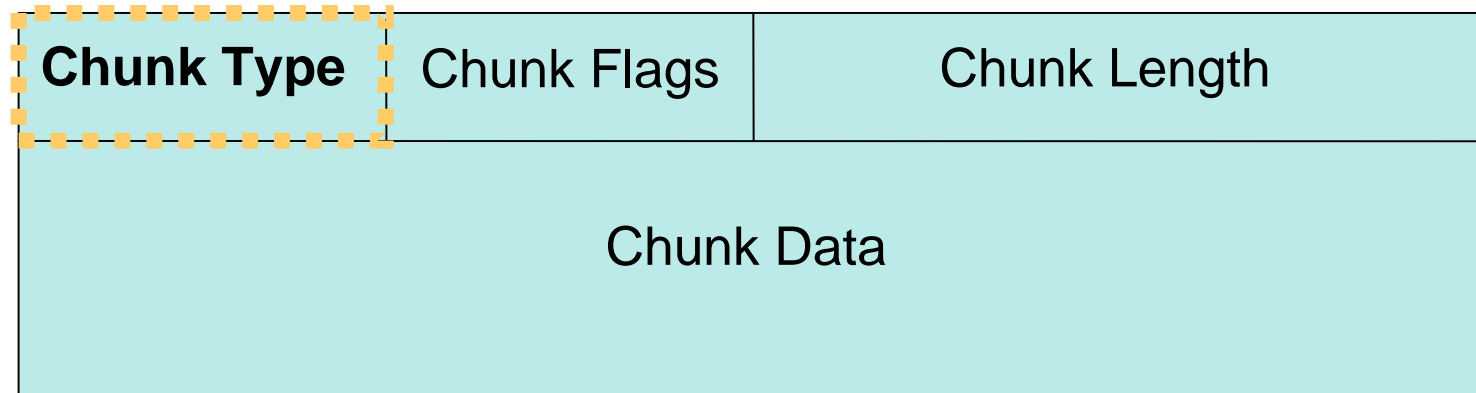
Every chunk has a TLV form: Type (with flags), Length, then Value
The "generic" format of each chunk:



- **Chunk Type:** 8-bit value indicating the type of chunk
- **Chunk Flags:** 8-bit flags, defined on per chunk type basis
- **Chunk Length:** 16-bit length in bytes, including the chunk type, chunk flags, and chunk length fields.

Note that chunks are padded to 32-bit boundaries within an SCTP packet. Any padding bytes (0x00) used are NOT included in the chunk length

Chunk Types



- There are **20** chunk types currently defined in SCTP (including both extensions in RFCs and those still in Internet Drafts):
 - (1) DATA (0x00)
 - (2) INITIATION [INIT] (0x01)
 - (3) INITIATION-ACKNOWLEDGMENT [INIT-ACK] (0x02)
 - (4) SELECTIVE-ACKNOWLEDGMENT [SACK] (0x03)
 - (5) HEARTBEAT (0x04)... etc...

Complete List of Chunk Types

RFC2960

- (1) **DATA** (0x00)
- (2) **INIT** (0x01)
- (3) **INIT-ACK** (0x02)
- (4) **SACK** [SELECTIVE-ACKNOWLEDGMENT] (0x03)
- (5) **HEARTBEAT** (0x04)
- (6) **HEARTBEAT-ACK** (0x05)
- (7) **ABORT** (0x06)
- (8) **SHUTDOWN** (0x07)
- (9) **SHUTDOWN-ACK** (0x08)
- (10) **ERROR** [OPERATIONAL-ERROR] (0x09)
- (11) **COOKIE-ECHO** (0x0A)
- (12) **COOKIE-ACK** (0x0B)
- (13) **ECNE** [EXPLICIT CONGESTION NOTIFICATION ECHO] (0x0C)
- (14) **CWR** [CONGESTION WINDOW REDUCE] (0x0D)
- (15) **SHUTDOWN-COMPLETE** (0x0E)

PR-SCTP - RFC 3758

- (16) **FORWARD-TSN** (0xC0)

ADD-IP draft

- (17) **ASCONF** (0xC1)
[ADDRESS-CONFIGURATION]
- (18) **ASCONF-ACK** (0x80)

Packet-Drop draft

- (19) **PKT-DROP** (0x81)
[SCTP-PACKET-DROP-REPORT]

Authentication draft

- (20) **AUTH** [AUTHENTICATION] (0x82) –
about to undergo drastic changes;
may add 2-3 chunks.

General Chunk Processing

- In any SCTP packet, **control** chunks **always** come before **DATA** chunks
- Some chunks **must** be singletons: INIT or INIT-ACK
- Chunk type number assignments are not linear, because...
- Chunk type upper two bits have specific meanings used for processing unrecognized chunks
 - 00xxxxxx => silently drop
 - 01xxxxxx => send an ERROR chunk in reply
 - In both cases, remainder of the packet is ignored
 - 10xxxxxx => Skip to next chunk
 - 11xxxxxx => Skip to next chunk and send an ERROR chunk

Pop Quiz

- **To see if you are paying attention:**

Assume you have an SCTP implementation that understands **NONE** of the extensions mentioned earlier.

- **What will the implementation do with:**

- FORWARD-TSN (0xC0)
- ASCONF (0xC1)
- ASCONF-ACK (0x80)
- PKT-DROP (0x81)
- AUTHENTICATION (0x82)

Connection Establishment

(actually, "association" establishment...)

SCTP is connection-oriented

- **Like TCP, SCTP is connection-oriented**
 - i.e. three phases: setup, communicate, teardown
 - requires a setup procedure to establish the communication relationship between two parties,
 - maintains state at the endpoints
- **Note: connection-oriented DOES NOT NECESSARILY imply reliable,**
 - SCTP is always connection oriented, but ...
 - SCTP can be configured to be reliable, unreliable or partially reliable
- **To establish this state, both sides go through a specific set of exchanges**
 - TCP uses a 3-way handshake (SYN, SYN/ACK, ACK)**
 - SCTP uses a 4-way handshake (INIT, INIT-ACK, COOKIE-ECHO, COOKIE-ACK)**

The a-word: *association*

- In TCP, the communication relationship between two endpoints is called a *connection*

Socket pair: { <Local IP addr, port>, <Remote IP addr, port> }

e.g. { <10.1.61.11, 2223>, <161.10.8.221, 80> }

- In SCTP, we would call this an *association*
- An SCTP association can be represented as a pair of SCTP endpoints:

assoc = { [10.1.61.11 : 2223],
 [161.10.8.221, 120.1.1.5 : 80] }

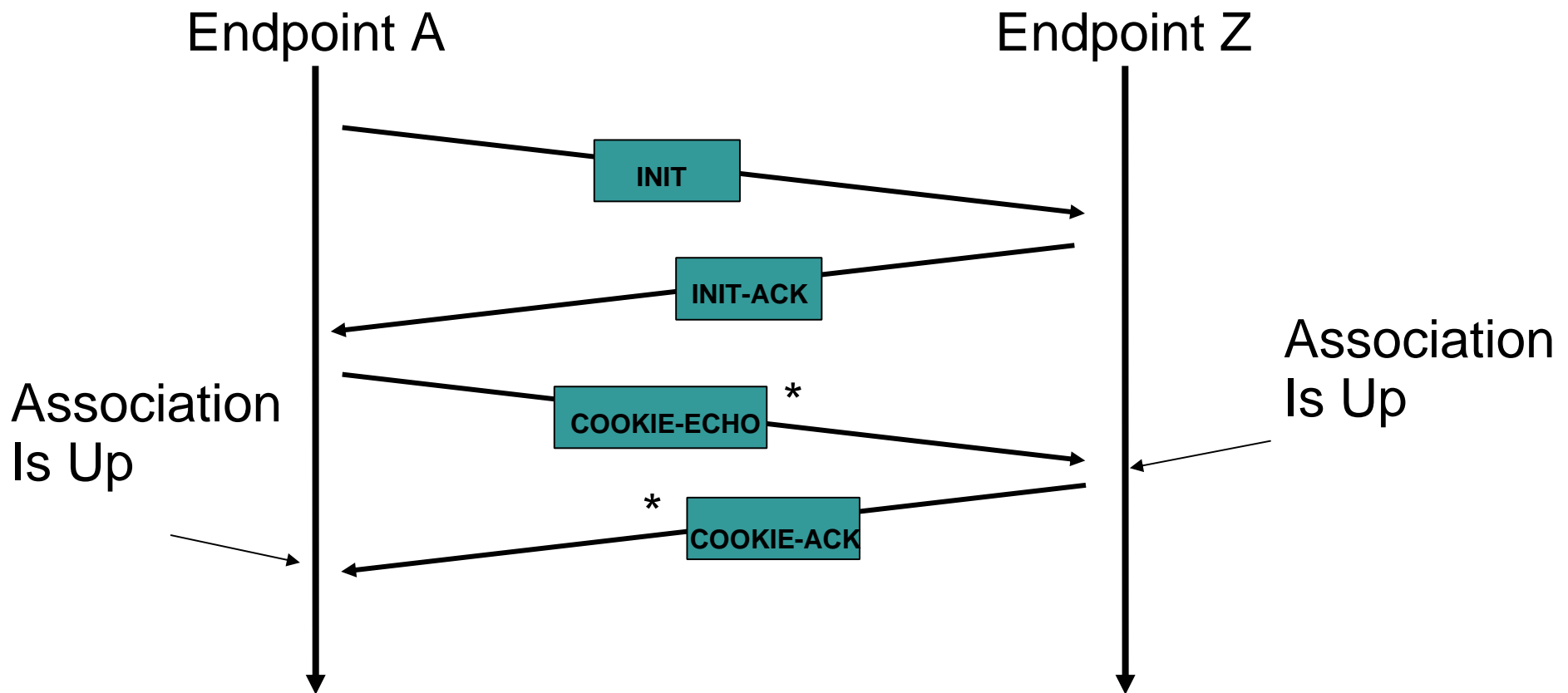
Note: second endpoint has two IP addresses

word "association" emphasizes that the two *endpoints* are "associated" rather than that two *IP-addresses (interfaces)* are "connected"

Associations and Endpoints

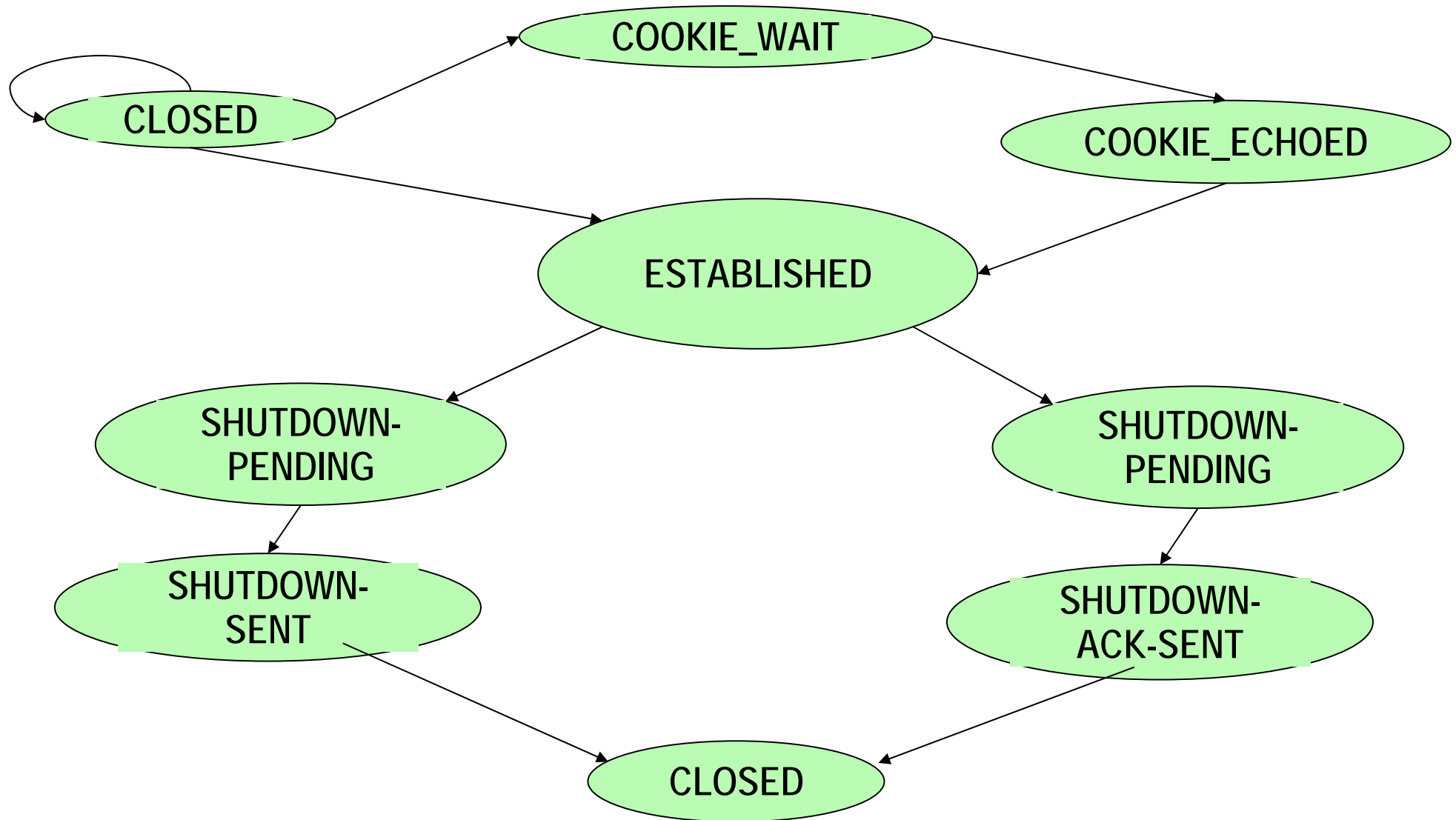
- **An SCTP endpoint is a port number on a specific host**
- **An SCTP endpoint may have multiple associations**
- **Only one association may be established between any *two* SCTP endpoints**

Setting Up an Association



* -- User data can be attached

SCTP state diagram



Data Transfer

Ordinary transfer

Streams

Unordered

PR-SCTP

Data Transfer Basics

- **We now shift our attention to normal data transfer.**
- **Data transfer happens in the ESTABLISHED, SHUTDOWN-PENDING, SHUTDOWN-SENT and SHUTDOWN-RECEIVED states.**
- **Note that even though the COOKIE-ECHO and COOKIE-ACK can optionally bundle DATA, we are in the ESTABLISHED state by the time the DATA is processed.**

DATA Chunk

Type=0x00	Flags=UBE	Length=variable
TSN Value		
Stream Identifier	Stream Sequence Num	
Payload Protocol Identifier		
Variable Length User Data		

- **Flag Bits:** U – Unordered Data B – Begin E-End (for fragmentation)
- **TSN:** transmission sequence num for ordering, reassembly, retransmission
- **Stream Identifier:** the stream number for this DATA
- **Stream Sequence Number:** orders this DATA chunk within the stream
- **Payload Protocol Identifier:** opaque value used by the endpoints
- **User Data:** the user message (or portion of)

Byte-stream vs. Messages

- **When data is transferred in TCP, the user gets a stream of bytes (not to be confused with SCTP streams).**
- **Users must “frame” their own messages if they are not transferring a stream of bytes (ftp might be considered an application that sends a stream of bytes).**
- **An SCTP user will send and receive messages. All message boundaries are preserved.**
- **A user will always read either ALL of a message or in some cases part of a message.**

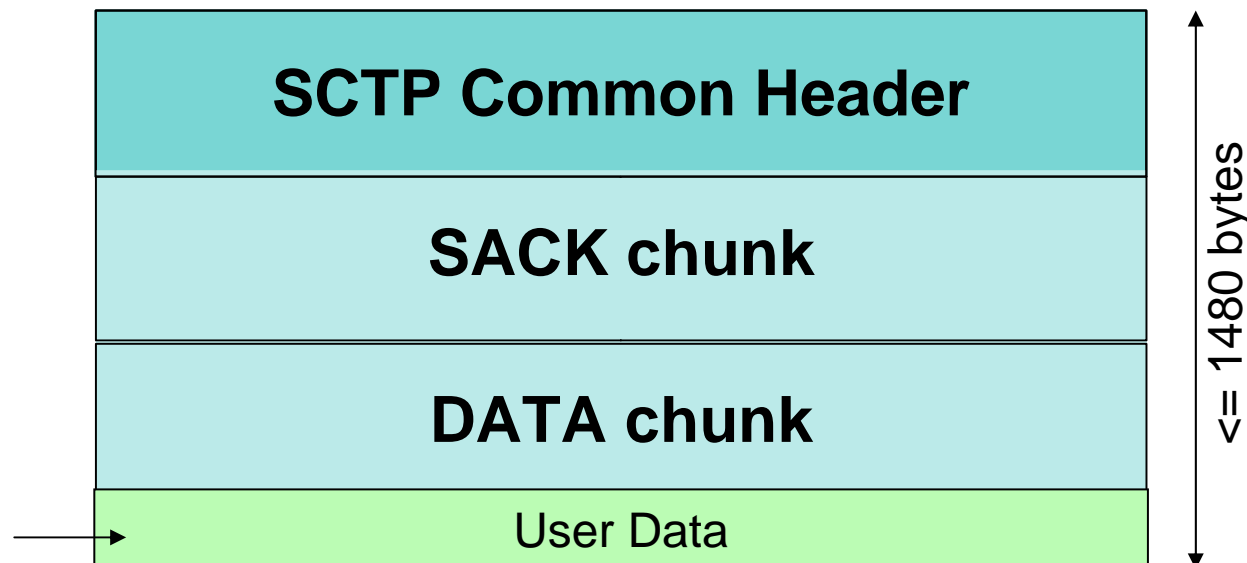
Receiving and Sending Messages

- **To send a message, the SCTP user...**
 - passes a message to either `sndmsg()` or `sctp_sndmsg()`**
 - (more on these two calls later)**
 - (could also just be `write()`, or any of its cousins...)**
- **The SCTP user at the other side...**
 - calls `rcvmsg()` to read the data (or `read()`, etc.)**
 - the SCTP user will NEVER see two different messages in a buffer returned from a single `rcvmsg()` call**
- **In between, the user message takes one of two paths through the SCTP stack:**
 - Singleton: Whole message fits in a single chunk**
 - or–**
 - Fragmentation: Message split up over multiple chunks**

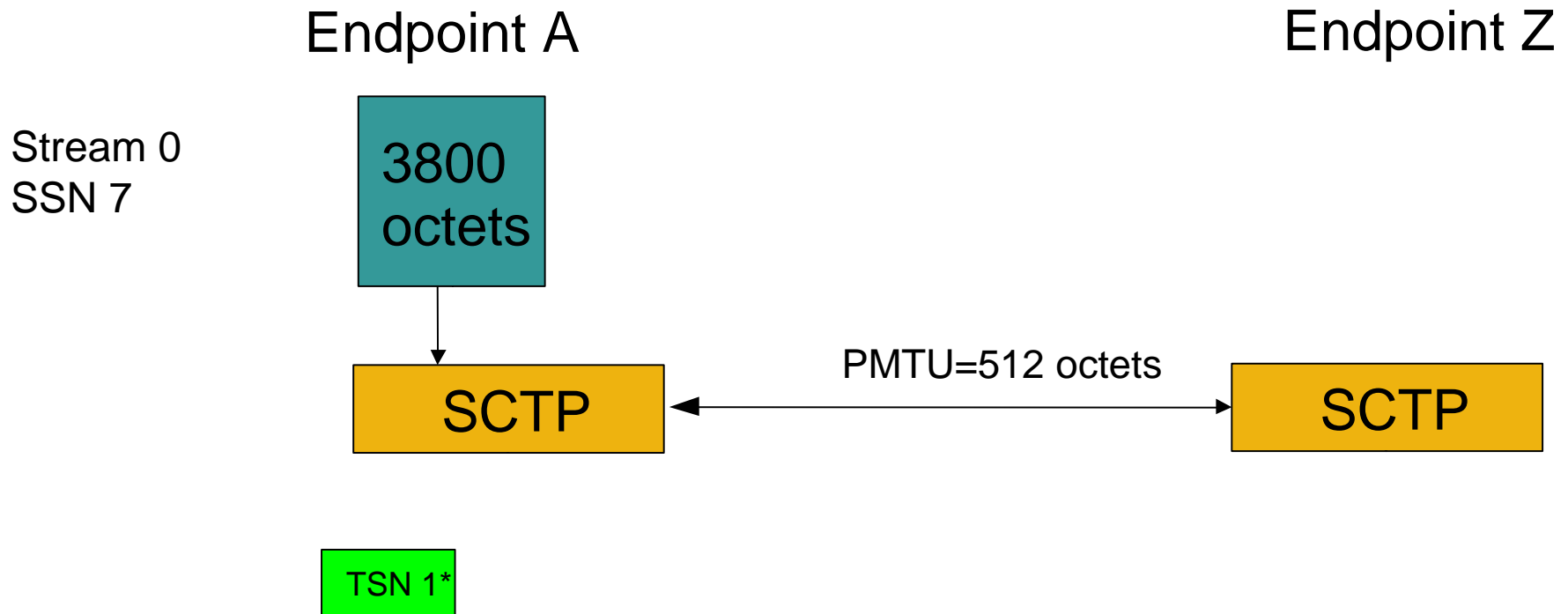
SCTP Singleton vs. Fragmentation

- **Singleton: message fits entirely in one SCTP chunk.**
- **maximum chunk size:**
 - smallest MTU of all of the peer's destination addresses**
- **Path MTU discovery is a required part of RFC2960**
- **But when it doesn't all fit, we fragment...**

*Singleton Example
Everything fits in one MTU...*



A Large Message Transfer

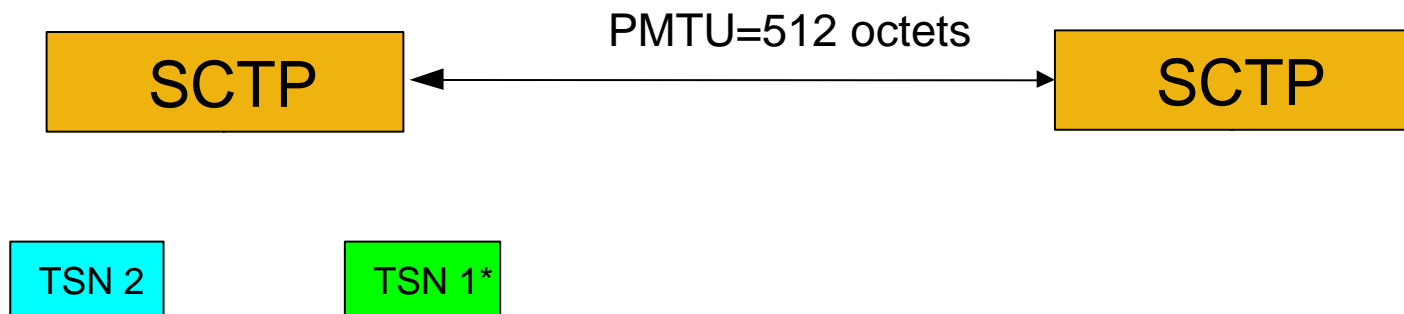


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

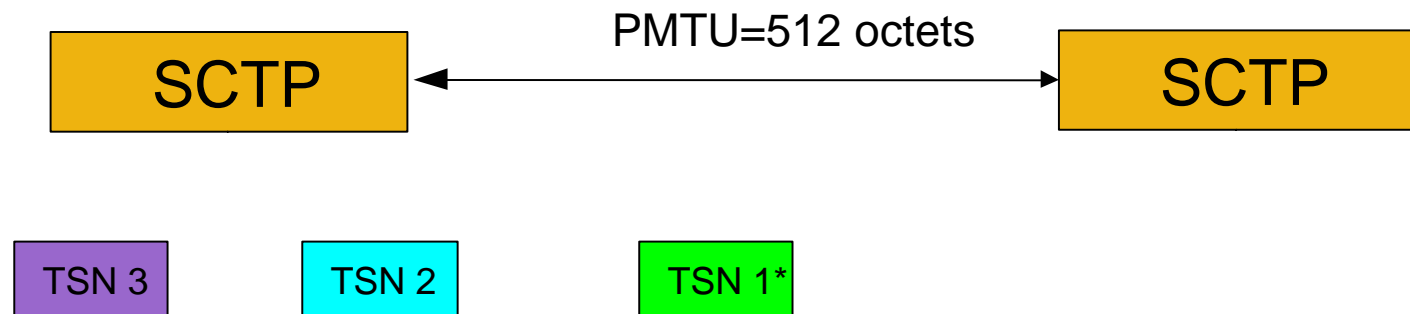


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

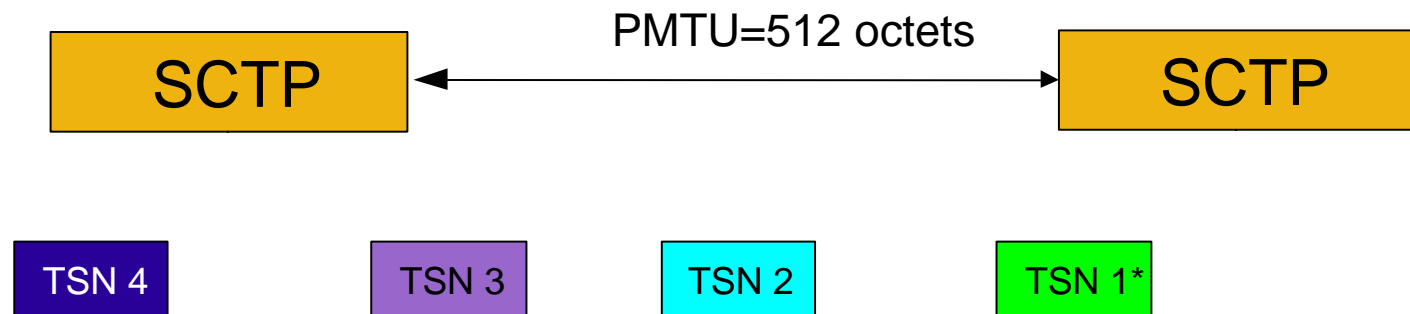


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

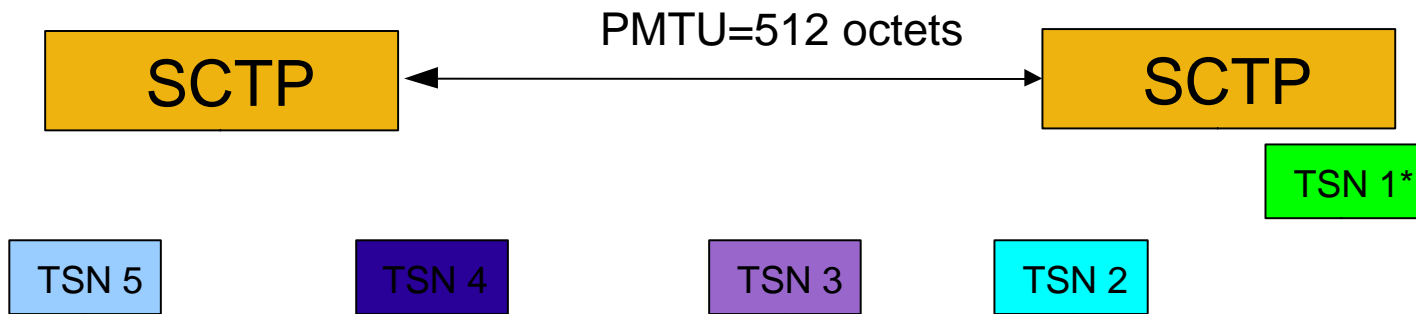


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

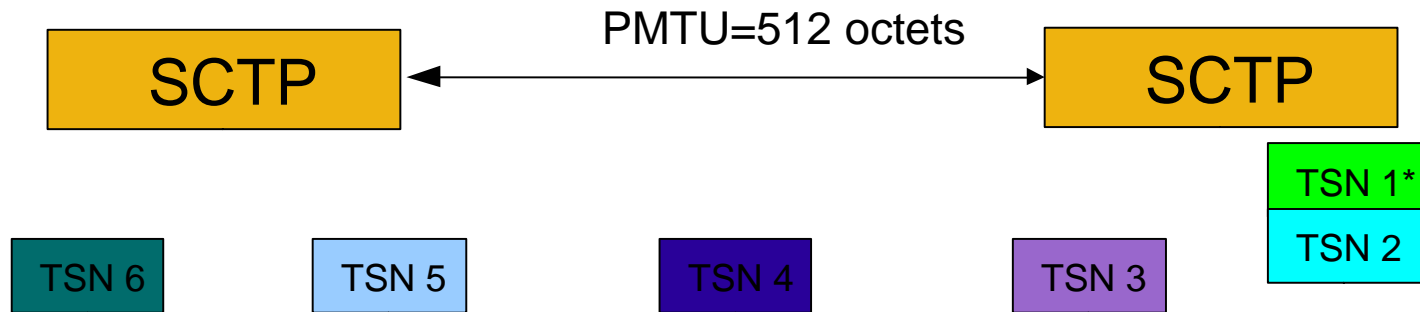


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

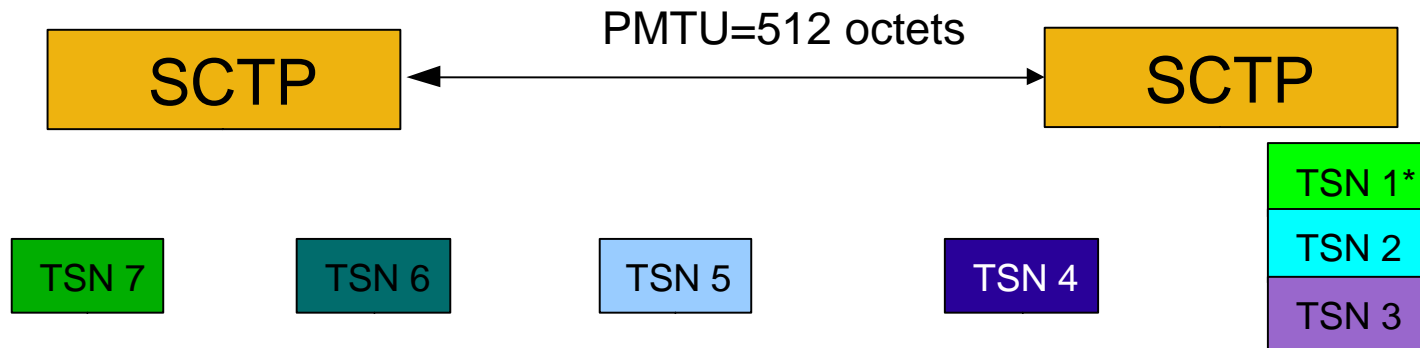


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

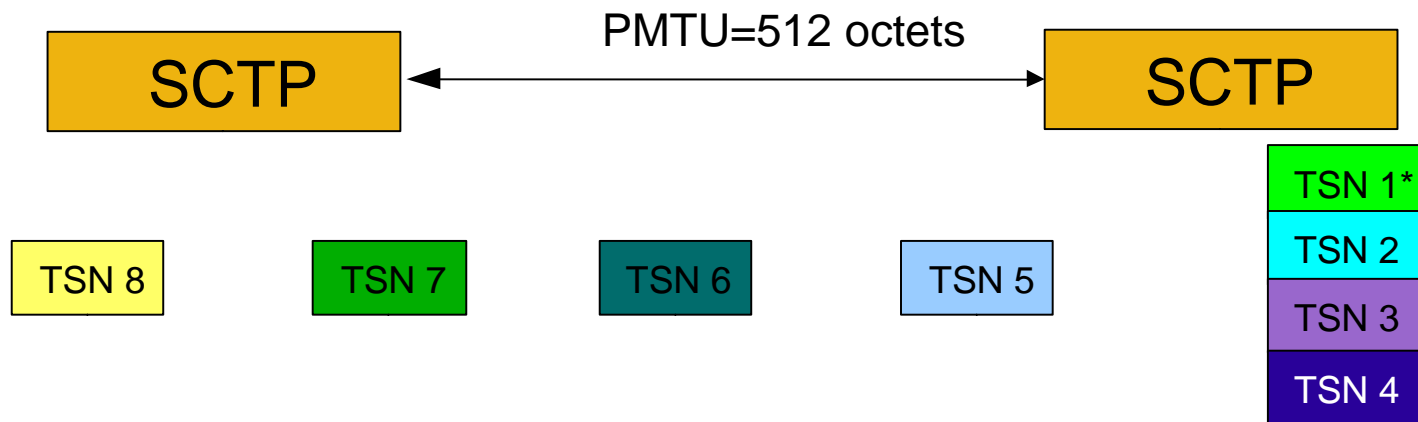


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z

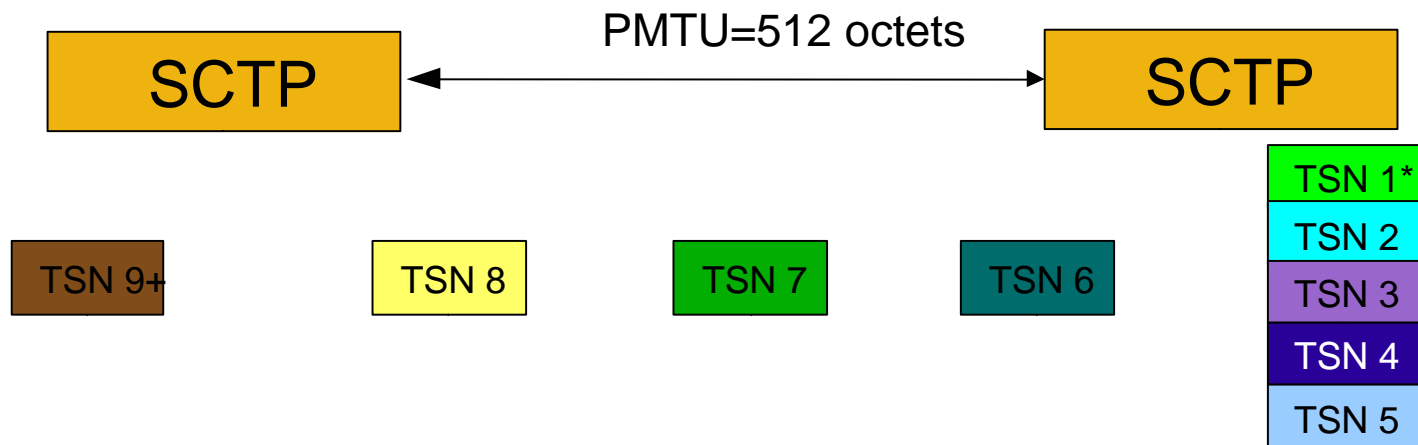


* - B bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z



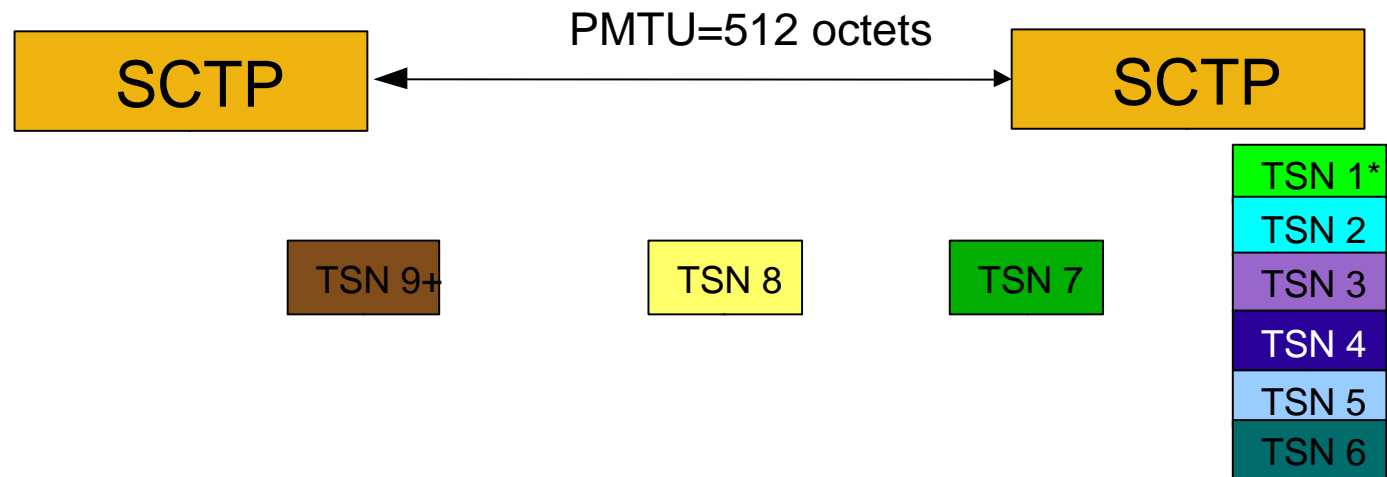
* - B bit set to 1

+ - E bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z



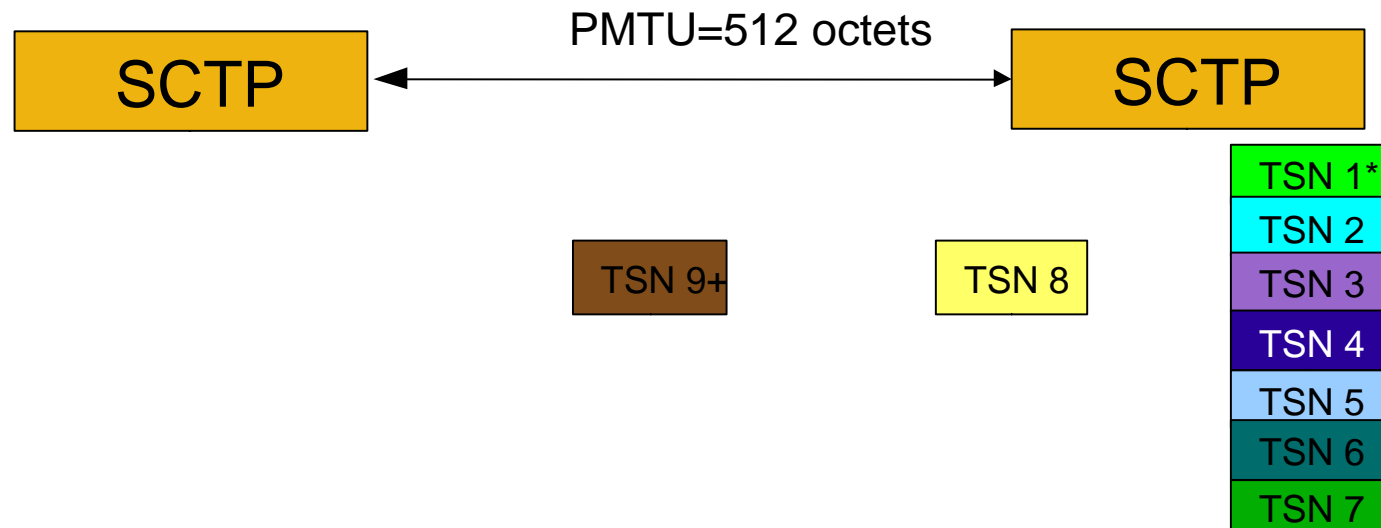
* - B bit set to 1

+ - E bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z



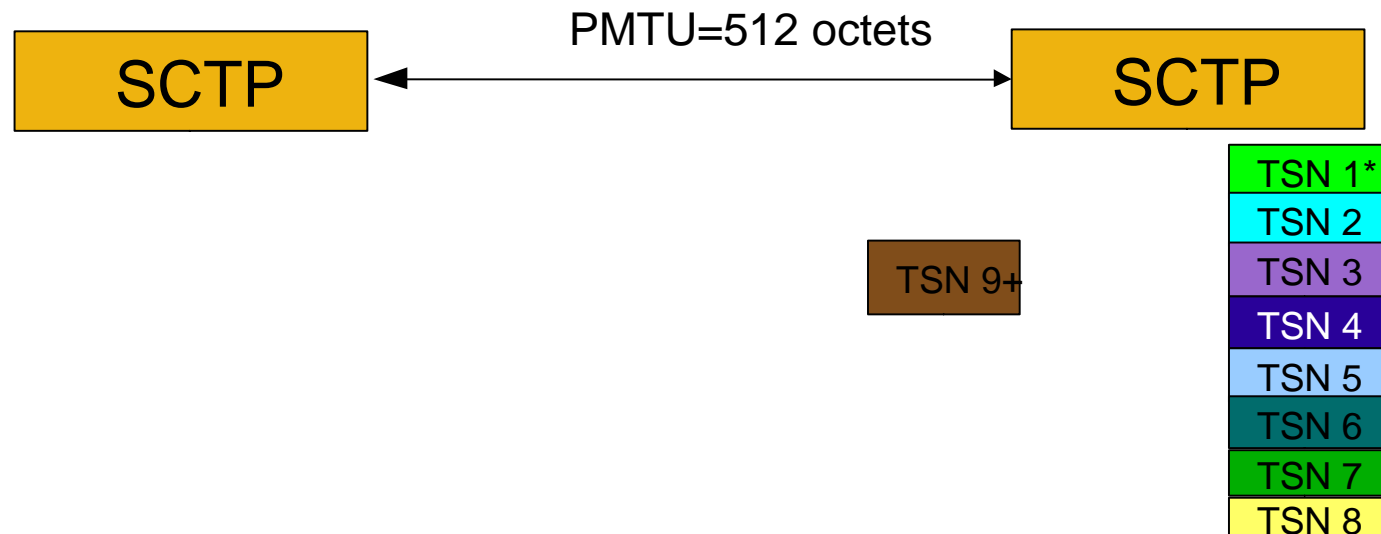
* - B bit set to 1

+ - E bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z



* - B bit set to 1

+ - E bit set to 1

A Large Message Transfer

Endpoint A

Endpoint Z



Using Streams

- **Streams are a powerful mechanism that allows multiple ordered flows of messages within a single association.**
- **Messages are sent in their respective streams and if a message in one stream is lost, it will not hold up delivery of a message in the other streams**
- **The application specifies the stream number to send a message on using its API interface**

For sockets, this is generally `sctp_sendmsg()` or `sctp_send()`

Streams and Ordering

- **A sender tells the `sndmsg()`, `sctp_sndmsg()`, or `sctp_send()` function which stream to send data on.**
- **Both ordered and un-ordered data can be sent within a stream.**

For un-ordered data, delivery to the upper layer is immediate upon receipt.

For ordered data, delivery may be delayed due to reassembly from network reordering.

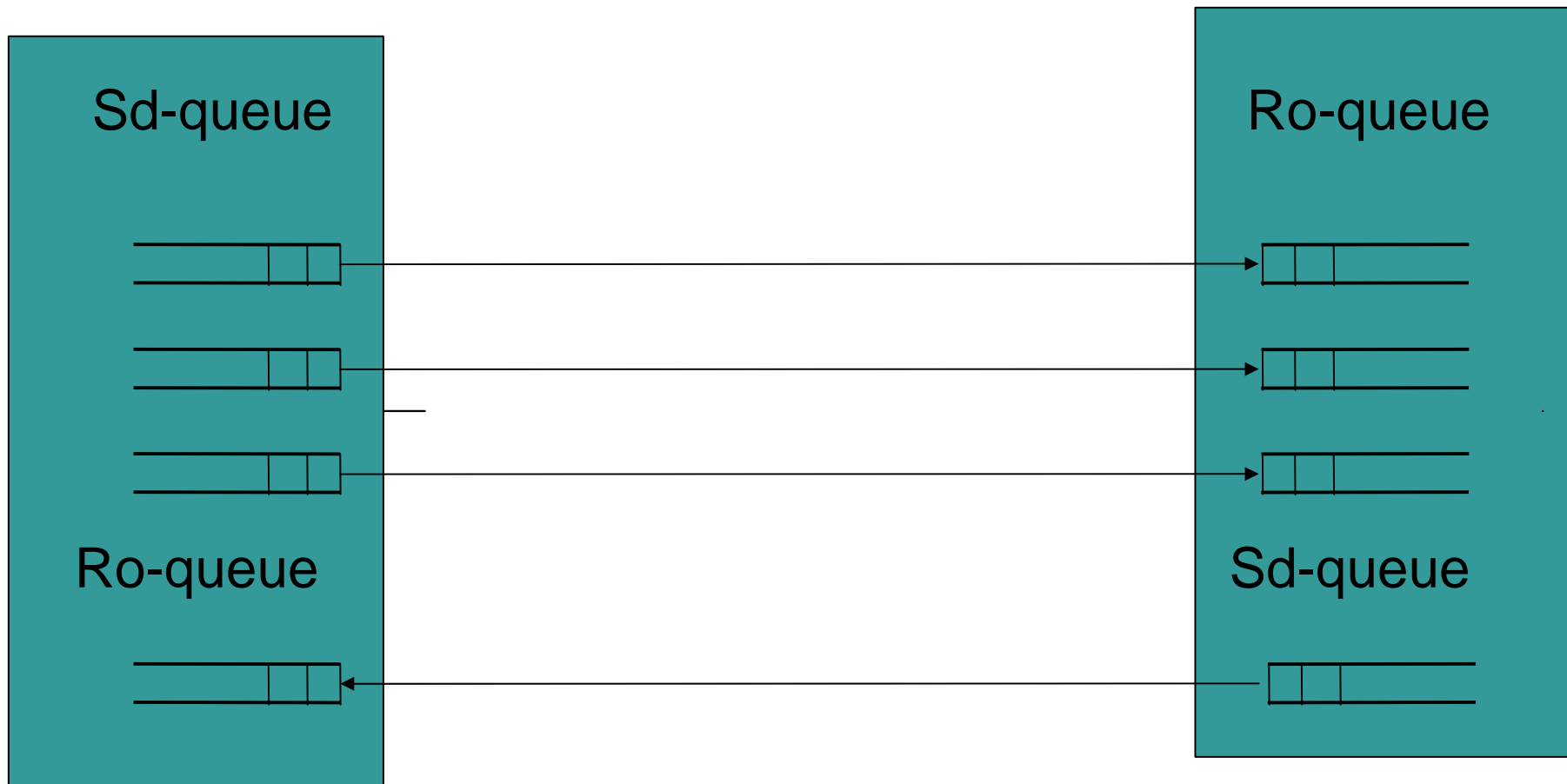
More on Streams

- **A stream is uni-directional**
SCTP makes NO correlation between an inbound and outbound stream
- **An association may have more streams traveling in one direction than the other.**
Valid stream number ranges for each direction are set during association setup
- **Generally an application will want to tie two streams together.**

Stream Queues

- **Usually, each side of an association maintains a send queue per stream and a receive queue per stream for reordering purposes.**
- **Stream Sequence Numbers (SSN) are used for reordering messages in each stream.**
- **TSN's are used for retransmitting lost DATA chunks.**

SCTP Streams



PR-SCTP I

- **Partial Reliability SCTP allows a sender to “skip” unacknowledged messages.**
- **Both endpoints must support the extension. A parameter is passed during setup to show that support is present on each side of the association.**
- **Normally, an application will put a “time limit” on the life of any given message.**
- **When this time limit expires and the message has not been acknowledged, a “skip message” is sent (FORWARD-TSN chunk)**

PR-SCTP II

- **The FORWARD-TSN chunk specifies the new cumulative TSN point for the remote end.**
- **It also specifies any stream and sequences that are being skipped by.**
- **The stream information aids a receiving endpoint in finding held messages for reordering on stream queues.**

PR-SCTP III

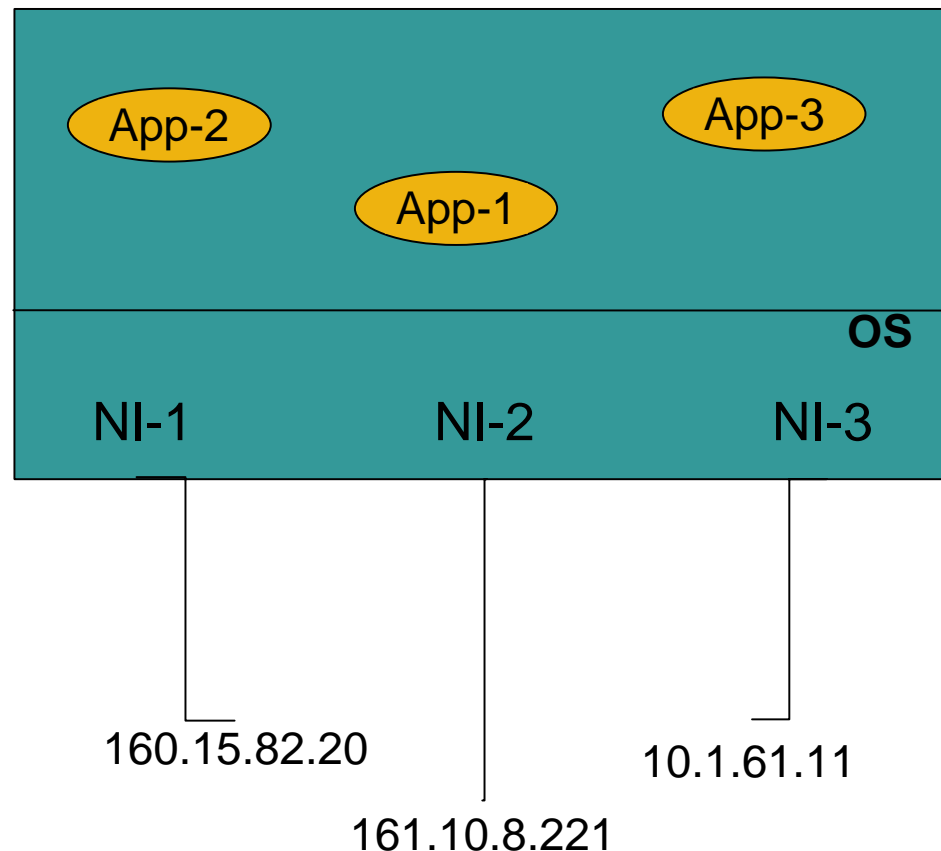
- **When a FWD-TSN is received, the receiver must update its cumulative ack point and respond with a SACK.**
- **The FWD-TSN mechanism is separated in the PR-SCTP document from the decision process for skipping a TSN.**
- **The document details an extension of the lifetime mechanism but other API interfaces are possible.**
- **A receiver does not need to be aware of the sender side policy for skipping TSN's.**

Failure Detection and Recovery

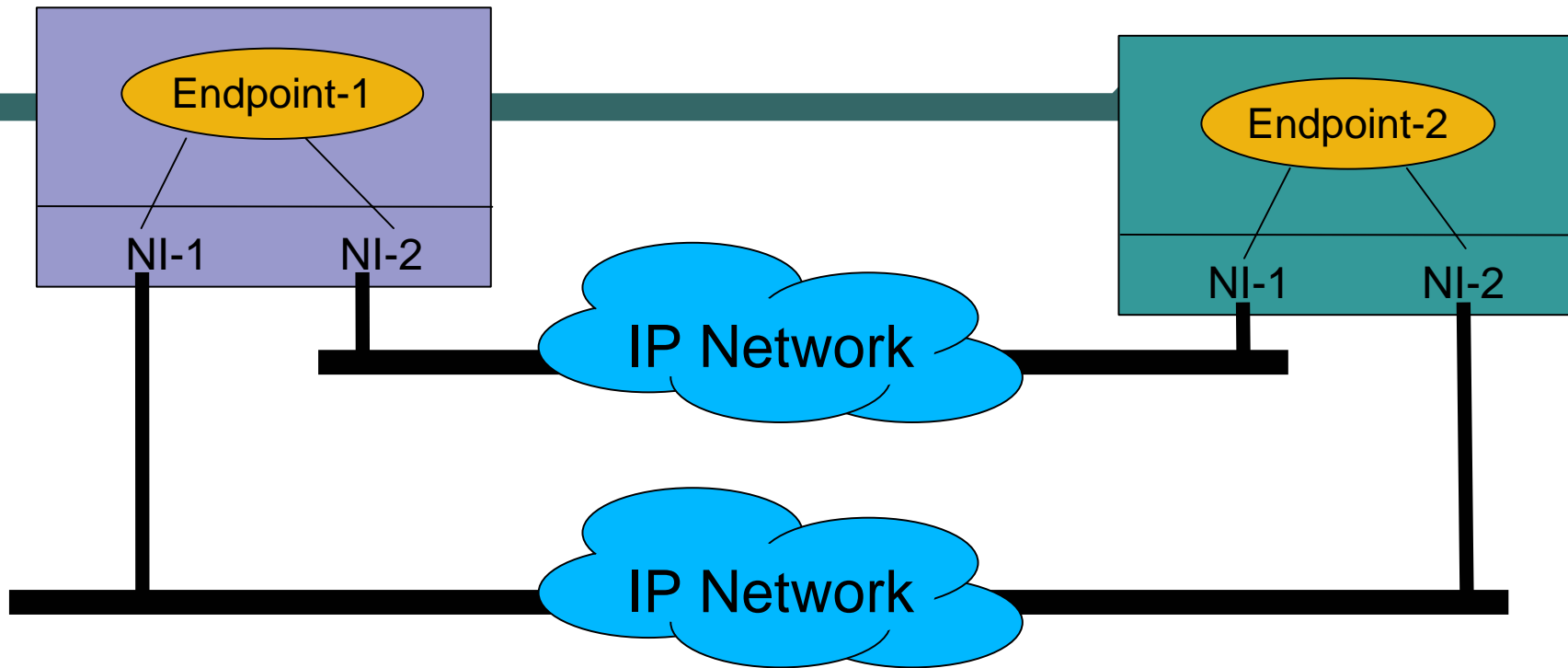
Heartbeats
Multihoming

IP Multi-homing

- The following figure depicts a typical multi-homed host. Keep this picture in mind when we discuss multi-homing.



Multi-homed Considerations

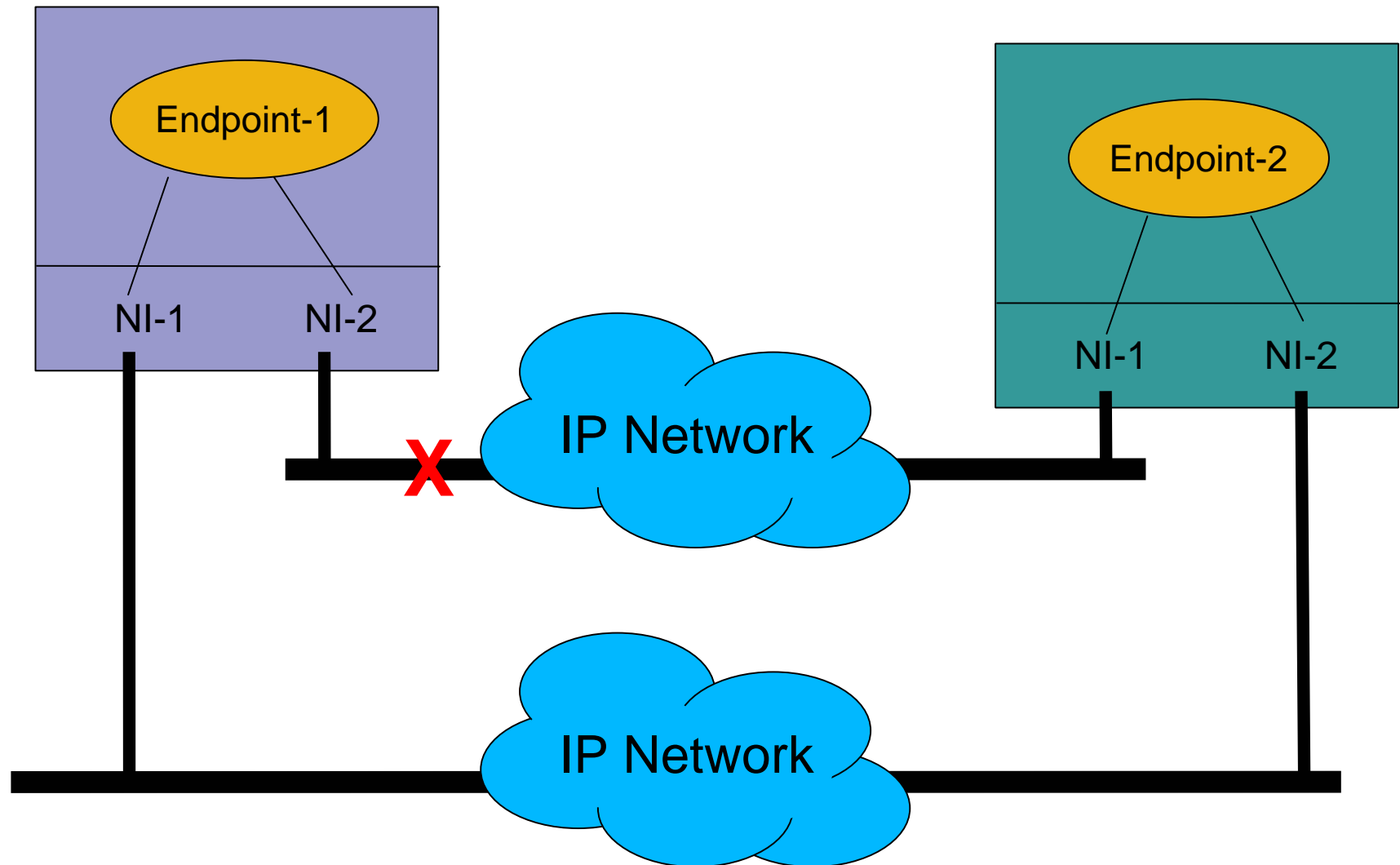


- When a peer is multi-homed, a “primary destination address” is selected by the SCTP endpoint.
- By default, all data is sent to this primary address.
- When the primary address fails, the sender selects an alternate primary address until it is restored or the user changes the primary address.

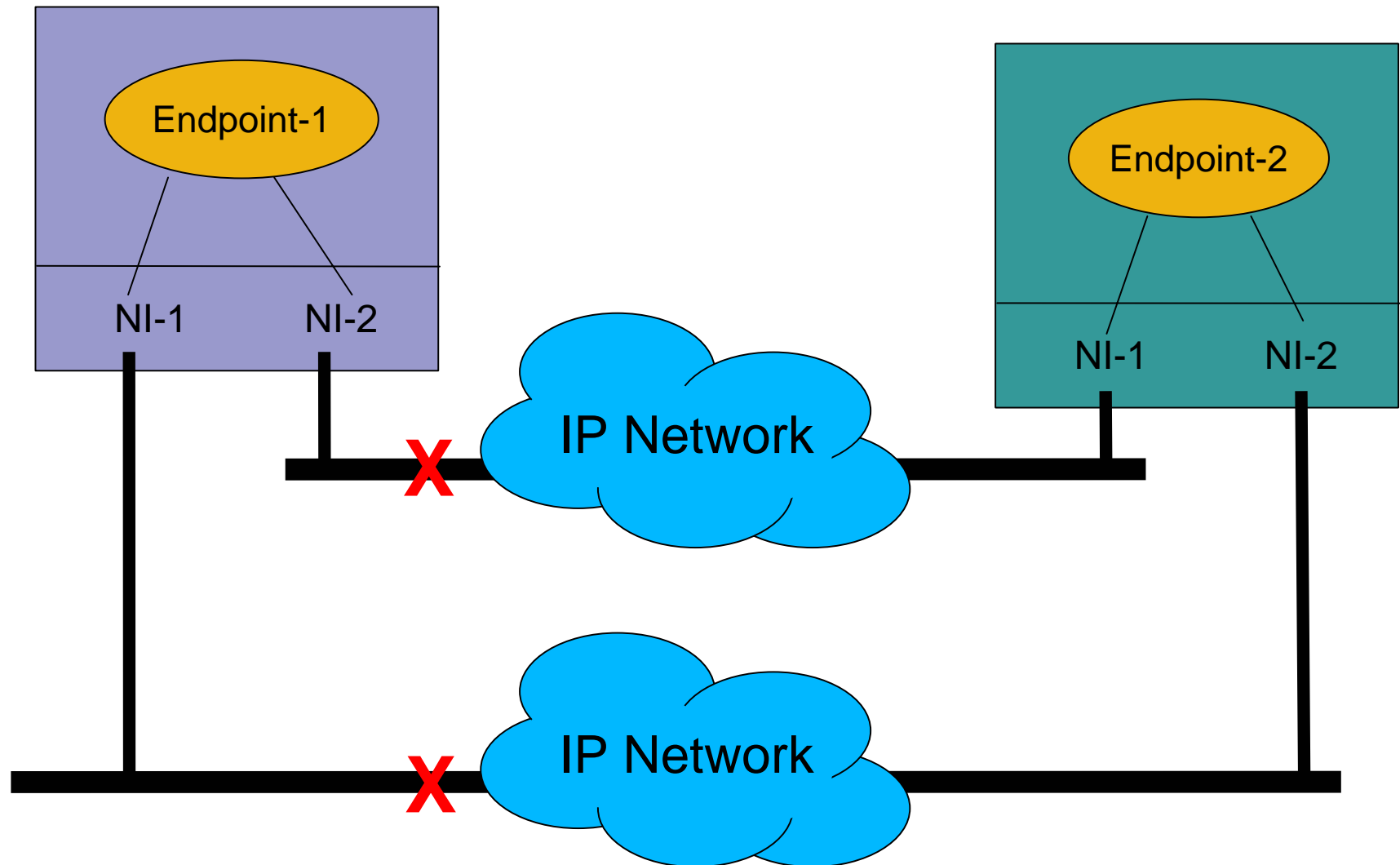
Failure Detection and Recovery

- **SCTP has two methods of detecting fault:**
 - Heartbeats**
 - Data retransmission thresholds**
- **Two types of faults can be discovered:**
 - An unreachable address**
 - An unreachable peer**
- **A destination address may be unreachable due to**
 - interface failure**
 - network failures**
 - endpoint failure**

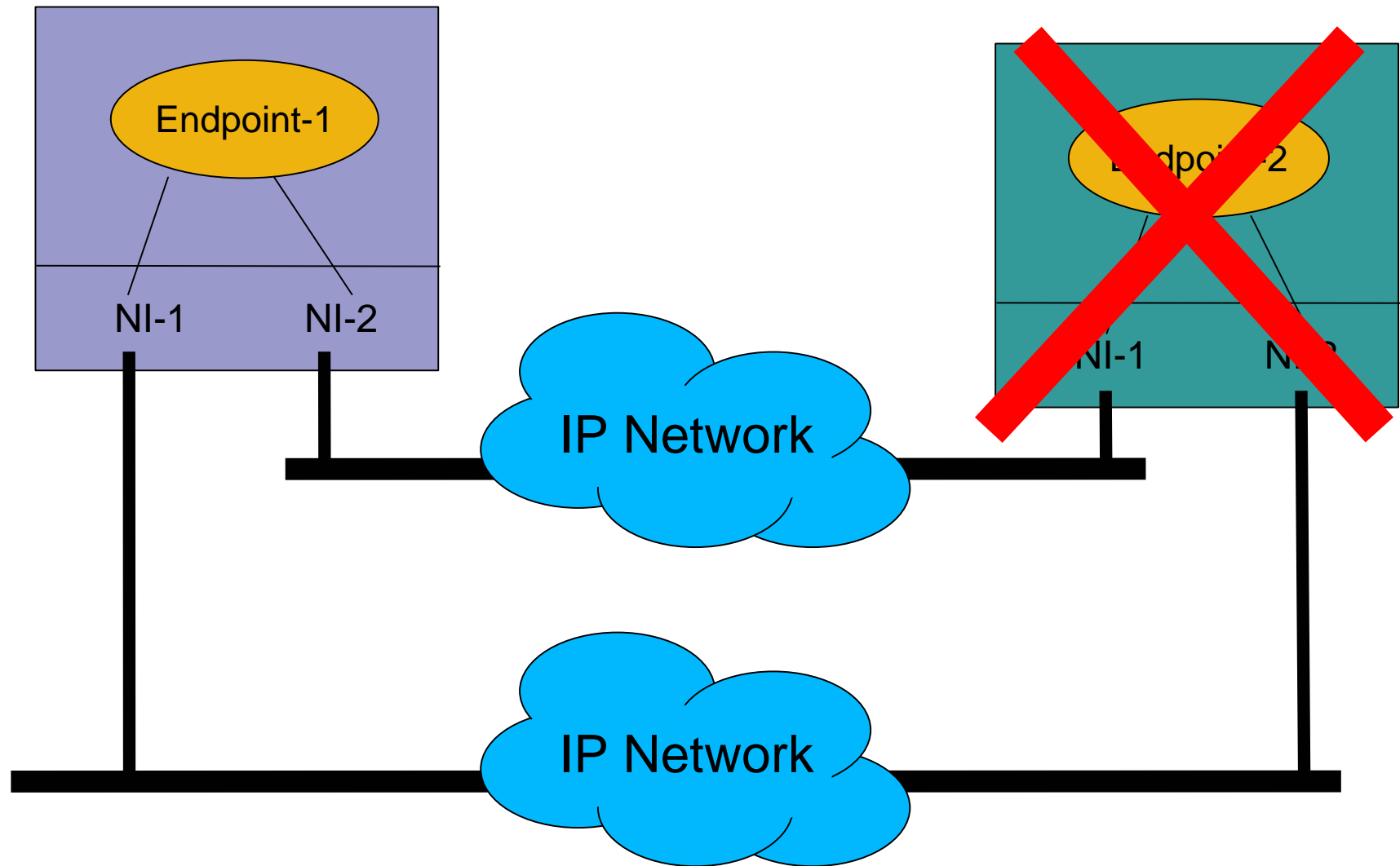
Unreachable Destination Address



Unreachable Peer: Network Failure



Unreachable Peer: Endpoint Failure



Heartbeat Monitoring Mechanism

- A HEARTBEAT is sent to any destination address that has been **idle** for longer than the **heartbeat period**
- A destination address is **idle** if no chunks that can be used for RTT updates have been sent to it
e.g. usually DATA and HEARTBEAT
- The heartbeat period timer is reset any time a DATA or HEARTBEAT are sent
- The peer responds with a HEARTBEAT-ACK

Other Uses for Heartbeats

- Heartbeat is also used to calculate RTT estimates
- The standard Van Jacobson SRTT calculation is done on both DATA RTTs or Heartbeat RTTs
- Just after association setup, Heartbeats will occur at a faster rate to “confirm” addresses
- **Address Confirmation** is a new concept added in Version 10 of the I-G

Heartbeat Controls

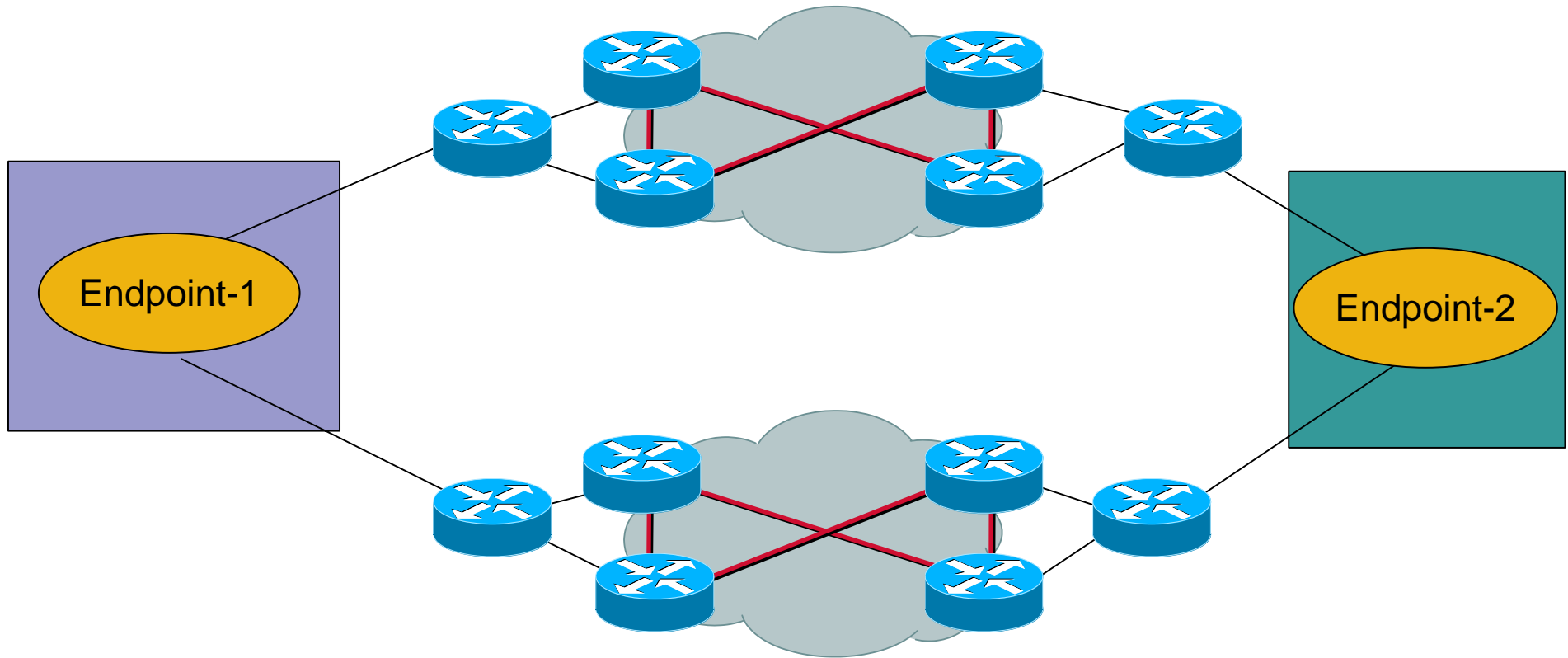
- **Heartbeats can be turned on and off.**
- **Heartbeats have a default interval of 30 seconds. This can also be adjusted.**
- **The Error thresholds can be adjusted:**
 - Each Destination's Error threshold**
 - Overall Association Error threshold**
- **Care must be taken in making any adjustments as false failure detections may occur.**

Network Diversity and Multi-homing

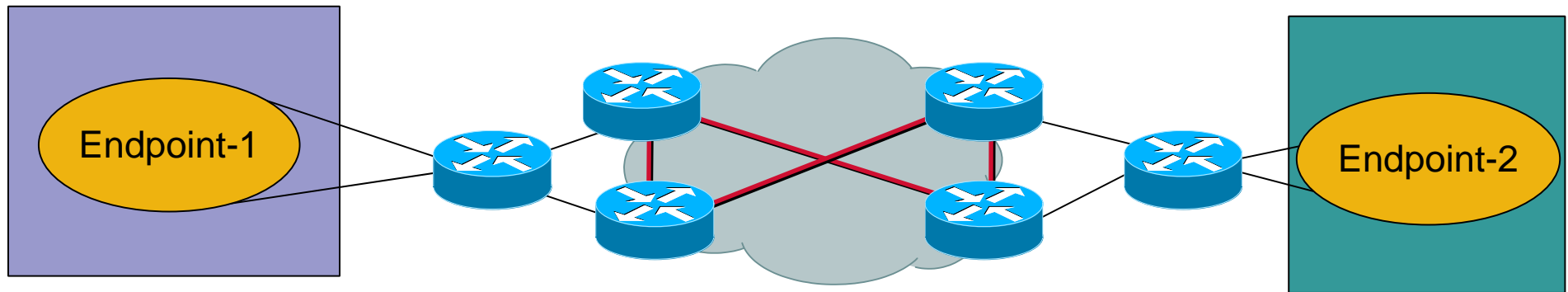
- **Multi-homing can assist greatly in preventing single points of failure**
- **Path diversity is also needed to prevent a single point of failure**
- **Consider the following two networks with maximum path diversity and minimal path diversity:**

Both hosts are multi-homed, but which network is more desirable?

Maximum Path Diversity



Minimum Path Diversity



ADD-IP Extensions

- **The ADD-IP draft allows dynamic changes to an address set of an endpoint without restart of the association.**

Outline

10h00-11h00	intro	Randy
	overview of SCTP What is SCTP? What are the major features?	Phill
11h15-12h15	SCTP details	Randy
13h15-14h15	details of sockets API (Randy)	Phill or Randy
	open Q and A	Both





Questions?