SCTP

A detailed overview of the protocol and a examination of the socket API

© 2003 Cisco Systems, Inc. All rights reserved.

Course Objectives: What You Should Get

- To come away with an understanding of the nuts and bolts of SCTP
- To know where in the course materials (the SCTP book and the RFC's) you can find information you may need when looking at an SCTP implementation
- To be able to understand the user interface to SCTP stacks (e.g. the SCTP sockets API)
- To know where the updates to the specification (and book) are (e.g. the I-G)

© 2003 Cisco Systems, Inc. All rights reserved.

Prerequisites

- A basic understanding of IP and transport protocols
- Some knowledge of TCP may be helpful, but is not strictly required.
- Willingness to put up with engineers that are attempting to teach a tutorial:-D

© 2003 Cisco Systems, Inc. All rights reserved.

Course Strategy

- **We will first do a detailed look at the actual protocol mechanisms**
- **We will point out reference material along the way as appropriate (i.e. RFC's and Internet-Drafts etc.)**
- **We expect YOU to ask questions if you get lost.**
- **We will cover a lot of ground in a limited time so hold on to your seats :-D**

© 2003 Cisco Systems, Inc. All rights reserved.

Reference Materials

- [SCTP reference book] Stream Control Transmission Protocol (SCTP): A Reference Guide, R. Stewart and Q. Xie, Addison-Wesley, 2002, ISBN 0-201-72186-4
- RFC 2960: Stream Control Transmission Protocol, October 2000
- RFC 3309: SCTP Checksum Change, September 2002
- [I-G] draft-ietf-tsvwg-sctpimpguide-10: SCTP Implementer's Guide

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Programming References

Cisco.com

- [sockets API] draft-ietf-tsvwg-sctpsocket-07: Sockets API Extensions for SCTP
- UNIX Network Programming, Volume 1, Third Edition, Stevens-Fenner-Rudoff, Addison-Wesley, 2004, ISBN 0-13-141155-1

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Extensions Drafts

- [PR-SCTP] RFC 3758
- [Add-IP] draft-ietf-tsvwg-addip-sctp-08: SCTP Dynamic Address Reconfiguration
- [Pkt-Drop] draft-stewart-sctp-pktdrprep-00: SCTP Packet Drop Reporting
- [Auth] draft-tuexen-sctp-auth-chunk-00: Authenticated Chunks for SCTP

© 2003 Cisco Systems, Inc. All rights reserved.

Online References

Cisco.com

- http://www.sctp.org

  Also reachable with HTTP over SCTP!

- http://www.ietf.org/html.charters/tsvwg-charter.html

  All current work on SCTP is done in the IETF TSVWG

- sctp-impl on mailer.cisco.com

© 2003 Cisco Systems, Inc. All rights reserved.

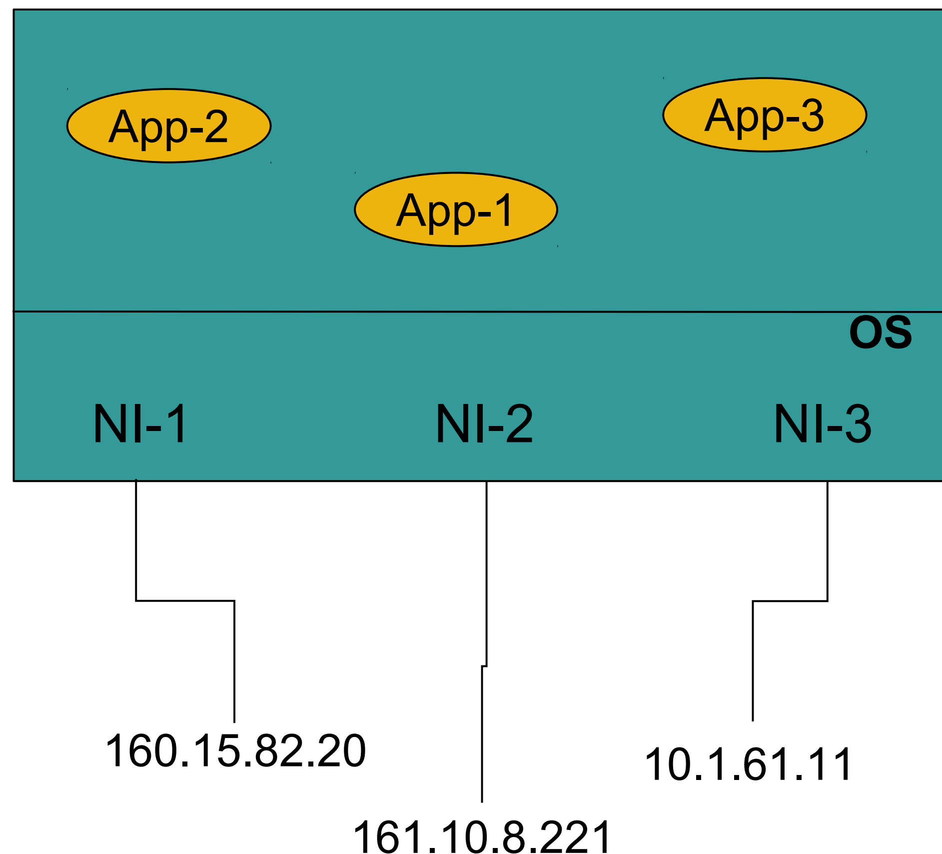
Features of SCTP

- Reliable data transfer w/SACK
- Congestion control and avoidance
- Message boundary preservation
- PMTU discovery and message fragmentation
- Message bundling
- Multi-homing support
- Multi-stream support
- Unordered data delivery option
- Security cookie against connection flood attack (SYN flood)
- Built-in heartbeat (reachability check)
- Extensibility

© 2003 Cisco Systems, Inc. All rights reserved.

IP Multi-homing

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



© 2003 Cisco Systems, Inc. All rights reserved.

Of Endpoints and Associations

- Two fundamental concepts in SCTP
  - Endpoints (communicating parties)
  - Associations (communication relationships)
- These two concepts are key to understanding the protocol and its basic operation
- We start by defining an “SCTP Transport Address”

© 2003 Cisco Systems, Inc. All rights reserved.

An SCTP Transport Address

- Each transport protocol defines a transport level header
- The transport level header helps demultiplex data coming to a host to the correct applications
- Applications in TCP and UDP bind to a “port” which forms the core method for demultiplexing data

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Transport Address (cont.)

- SCTP also defined the same byte positions in its transport header for the two 16 bit port fields
- We term the combination of an SCTP port and an IP address an "SCTP Transport Address"
- The IP address in an SCTP Transport Address MUST be a routeable unicast address

    i.e. multicast and broadcast addresses are invalid

© 2003 Cisco Systems, Inc. All rights reserved.

An SCTP Endpoint

- **An SCTP endpoint is the logical end of the SCTP transport protocol - a communicating party**
- **An SCTP endpoint may have MORE than one IP address but it always has *one and only one port* number**
- **An application typically will open an SCTP socket and bind one address, a set of addresses, or all addresses to that socket**

  This socket can then be thought of as an SCTP endpoint

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Endpoints II

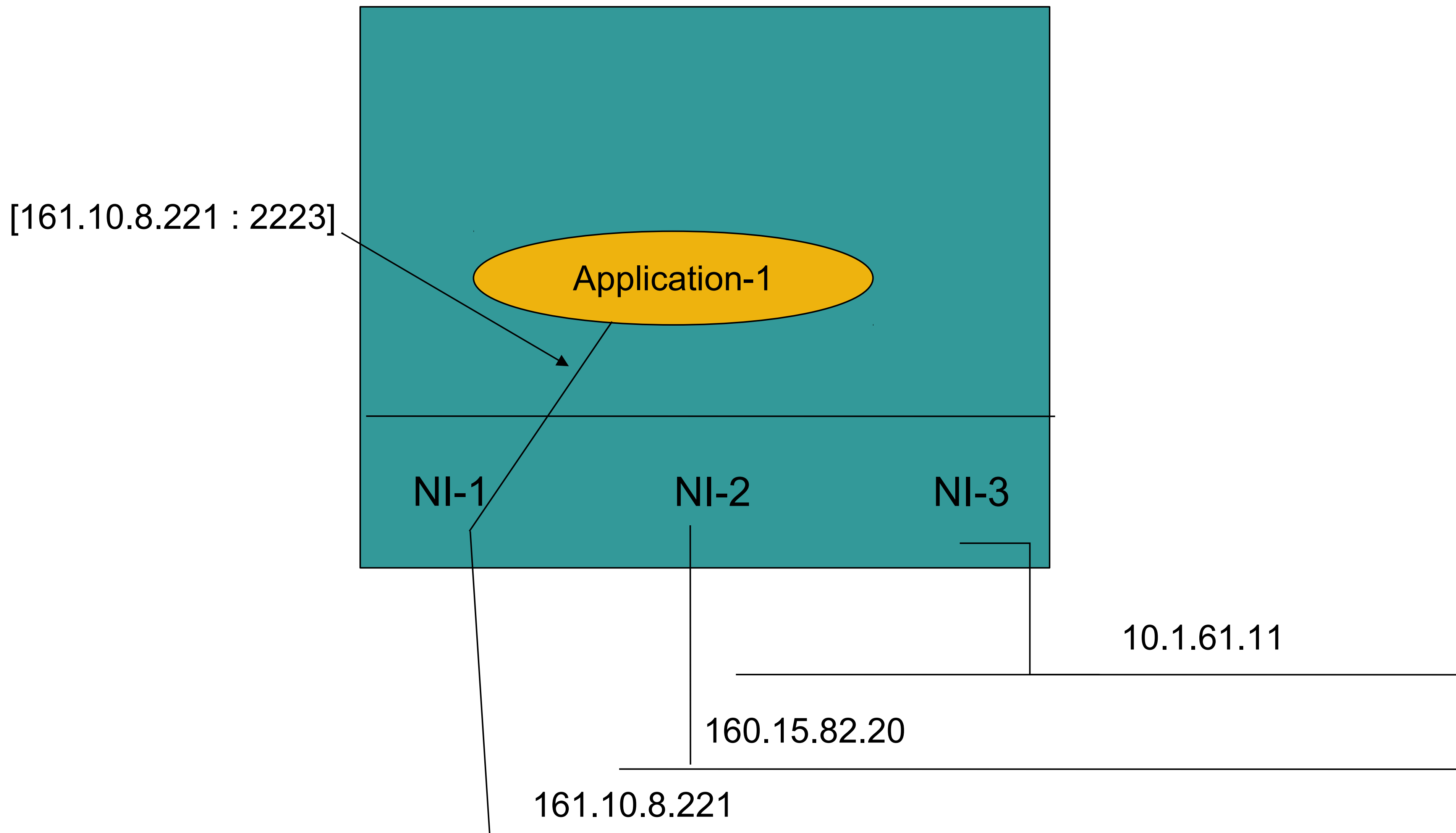
- An SCTP endpoint can be represented as a list of SCTP transport addresses with the same port:

    endpoint = [10.1.4.2, 10.1.5.3 : 80]

- An SCTP transport address can only be bound to one single SCTP endpoint

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Endpoint III



© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Endpoint IV

- Application-1 has bound one IP address of the host with the port 2223.
- If a new application is started Application-2, it may legally bind [160.15.82.20 : 2223] or [10.1.61.11 : 2223] or even [160.15.82.20, 10.1.61.11 : 2223]
- The new application will NOT be able to bind the existing SCTP Transport address that Application-1 has bound I.e: [161.10.8.221 : 2223]

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Associations

- Like TCP, SCTP is connection-oriented
- A connection-oriented protocol is one that requires a setup procedure to establish the communication relationship (and state) between two parties
- 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 (we examine this later)

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Association II

Cisco.com

- In TCP, the communication relationship between two endpoints is called a "connection"
- In SCTP, this is called an "association" this is because it is a broader concept than a single connection (i.e. multi-homing)
- 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]}

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Association III

- **An SCTP endpoint may have multiple associations**
- **Only one association may be established between any two SCTP endpoints**

© 2003 Cisco Systems, Inc. All rights reserved.

Operation of SCTP Associations

- An SCTP association provides reliable data transfer of messages
- Messages are sent within a **stream**, which is identified by a **stream identifier** (SID)
- Messages can be ordered or un-ordered:

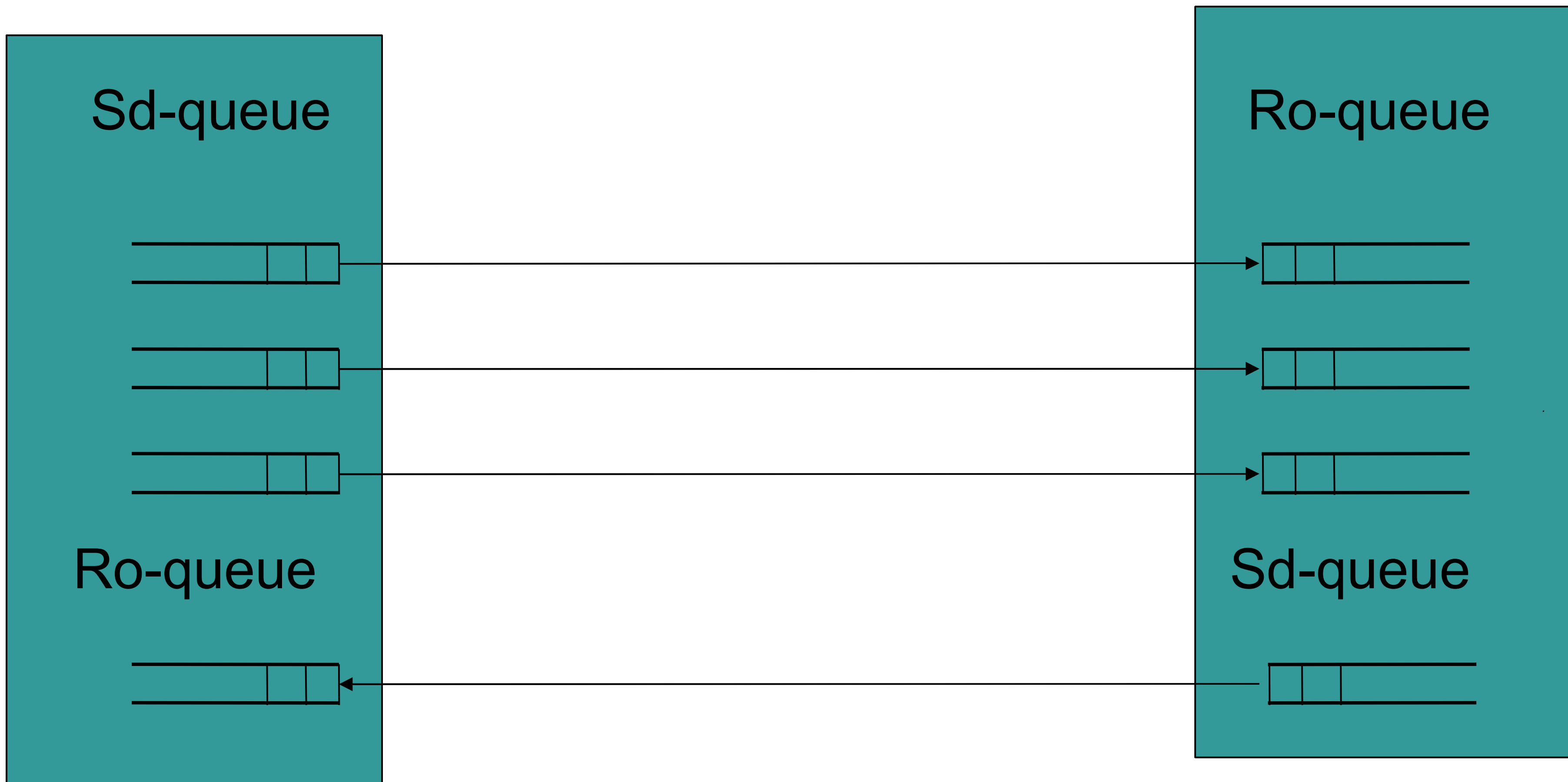
  Each ordered message sent within a stream is also assigned a **stream sequence number** (SSN)

  Unordered messages have no SSN and are delivered with no respect to ordering

© 2003 Cisco Systems, Inc. All rights reserved.

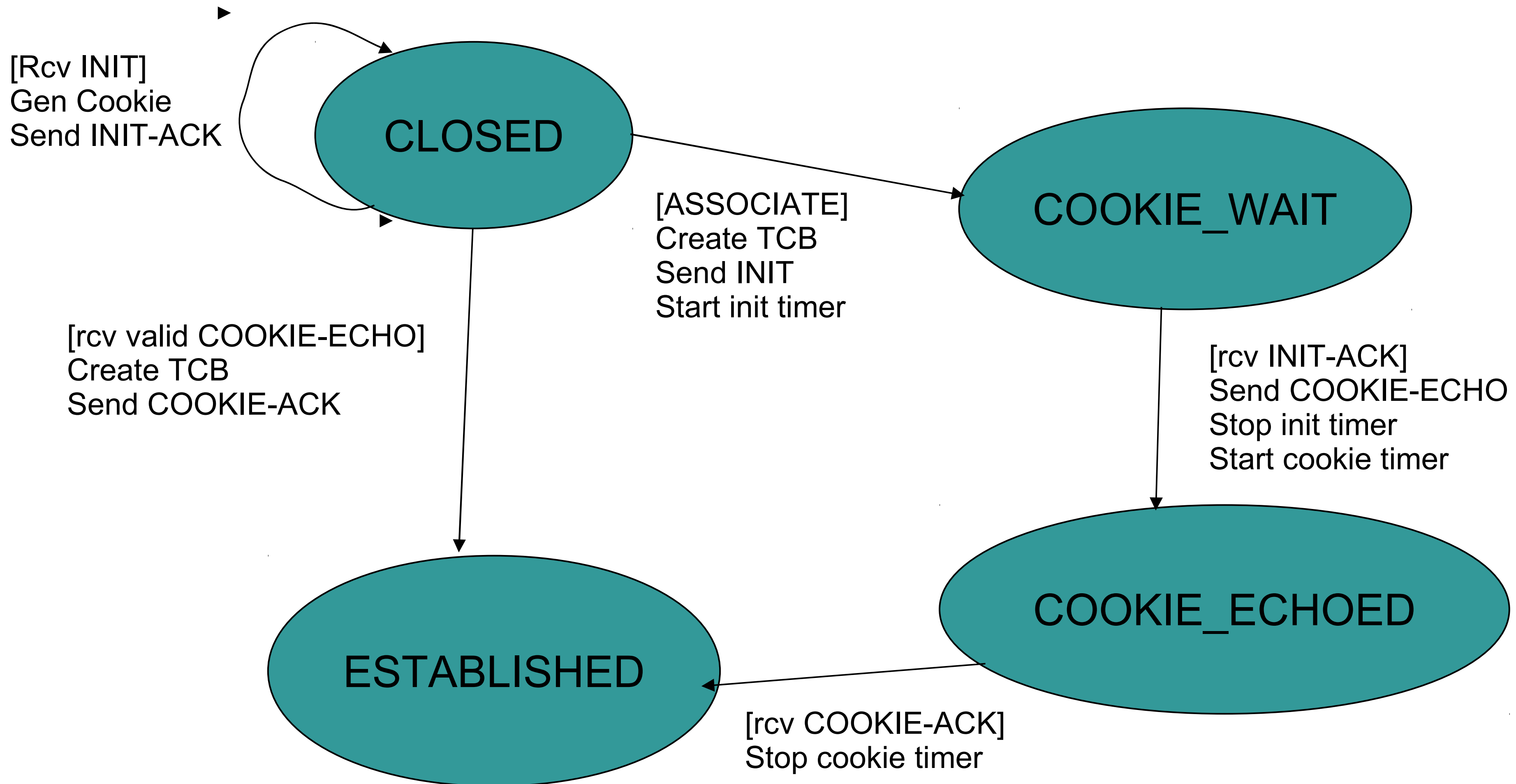
SCTP Streams

- **We will discuss further details in Data Transfer section later**



© 2003 Cisco Systems, Inc. All rights reserved.

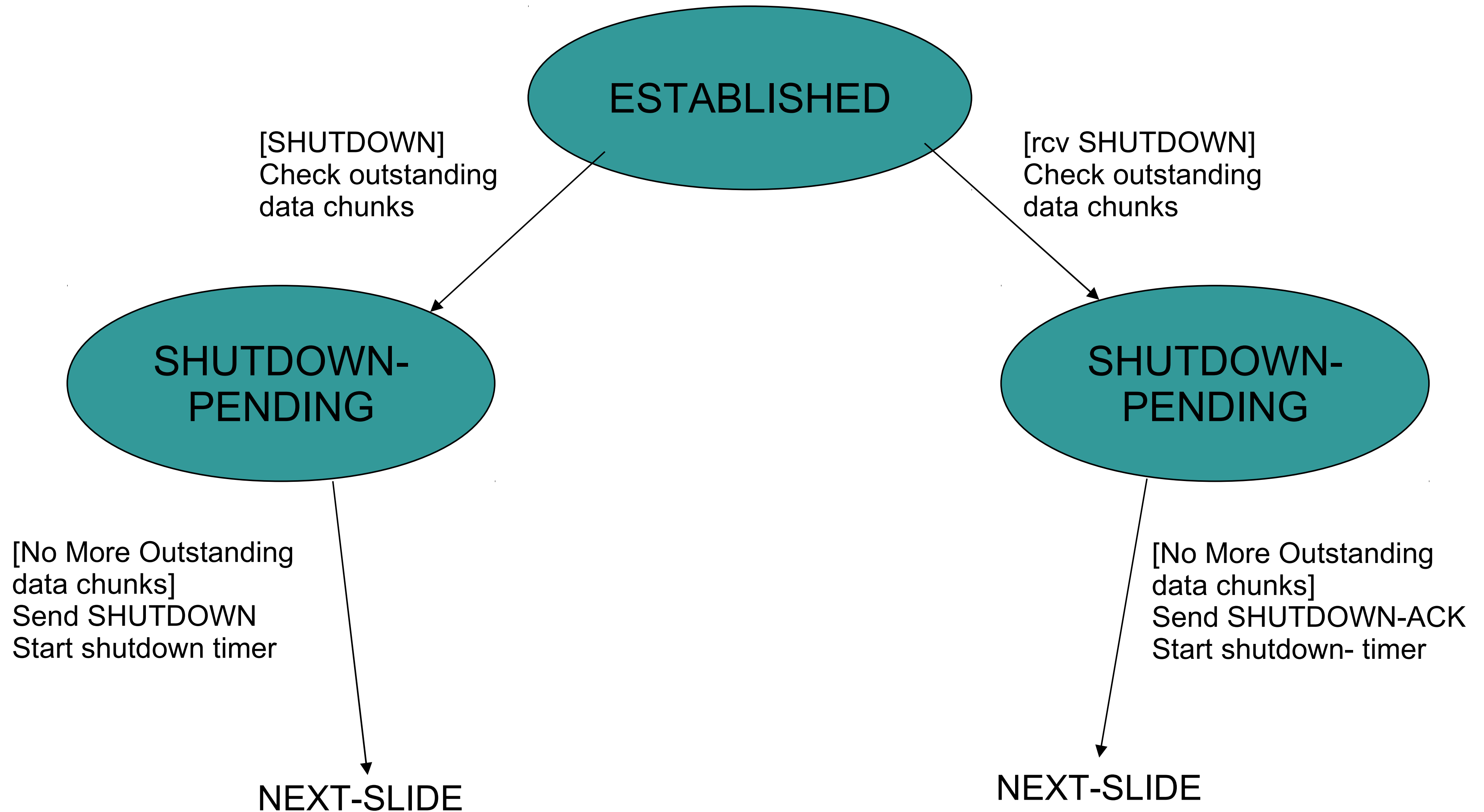
SCTP States I



Page 31 of the SCTP book

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP States II

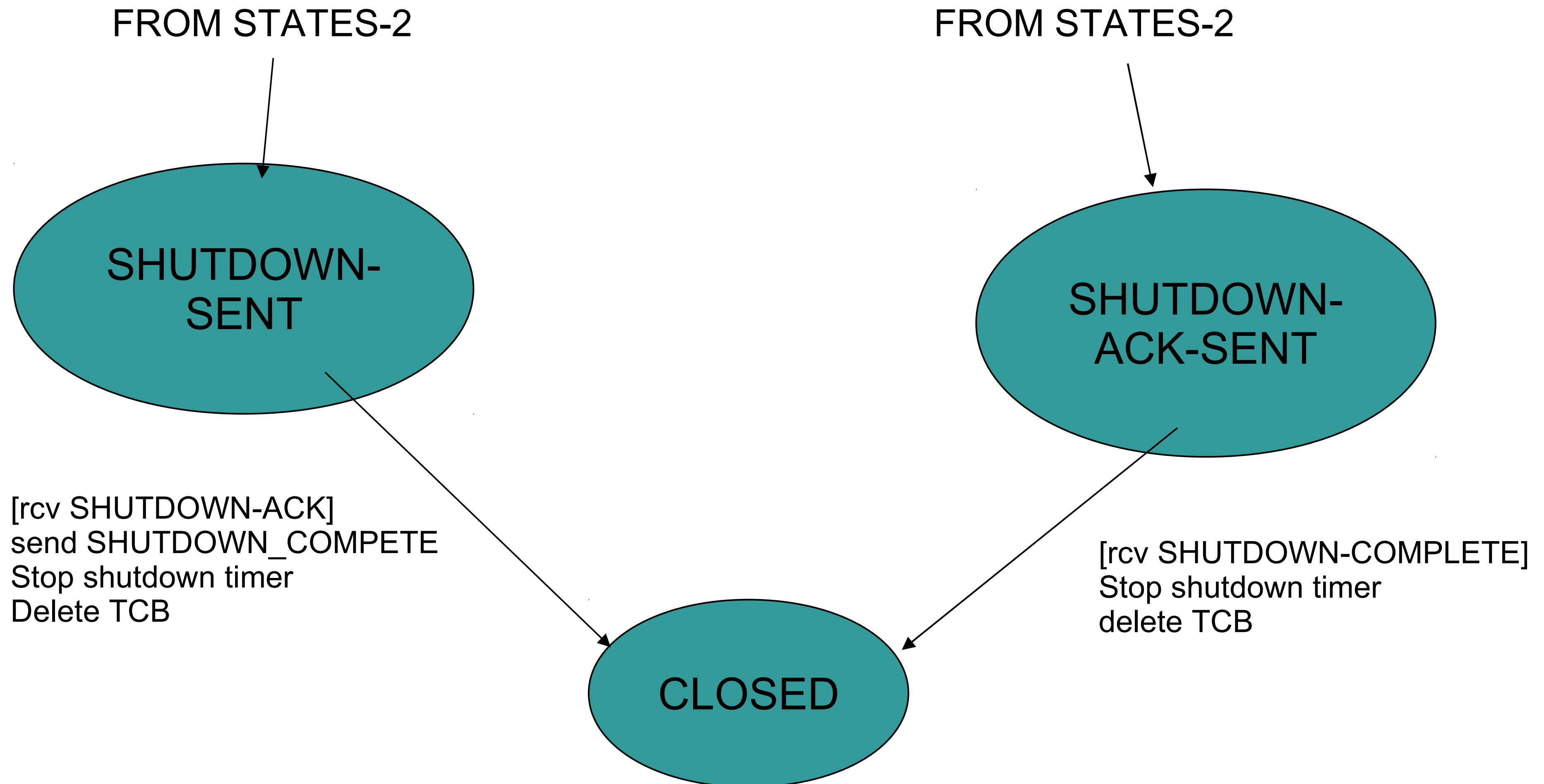


Page 32 of the SCTP book

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP States III

Cisco.com



Page 32 of the SCTP book

© 2003 Cisco Systems, Inc. All rights reserved.

Questions

- **Here we pause for any questions??**
- **Do you have any?**

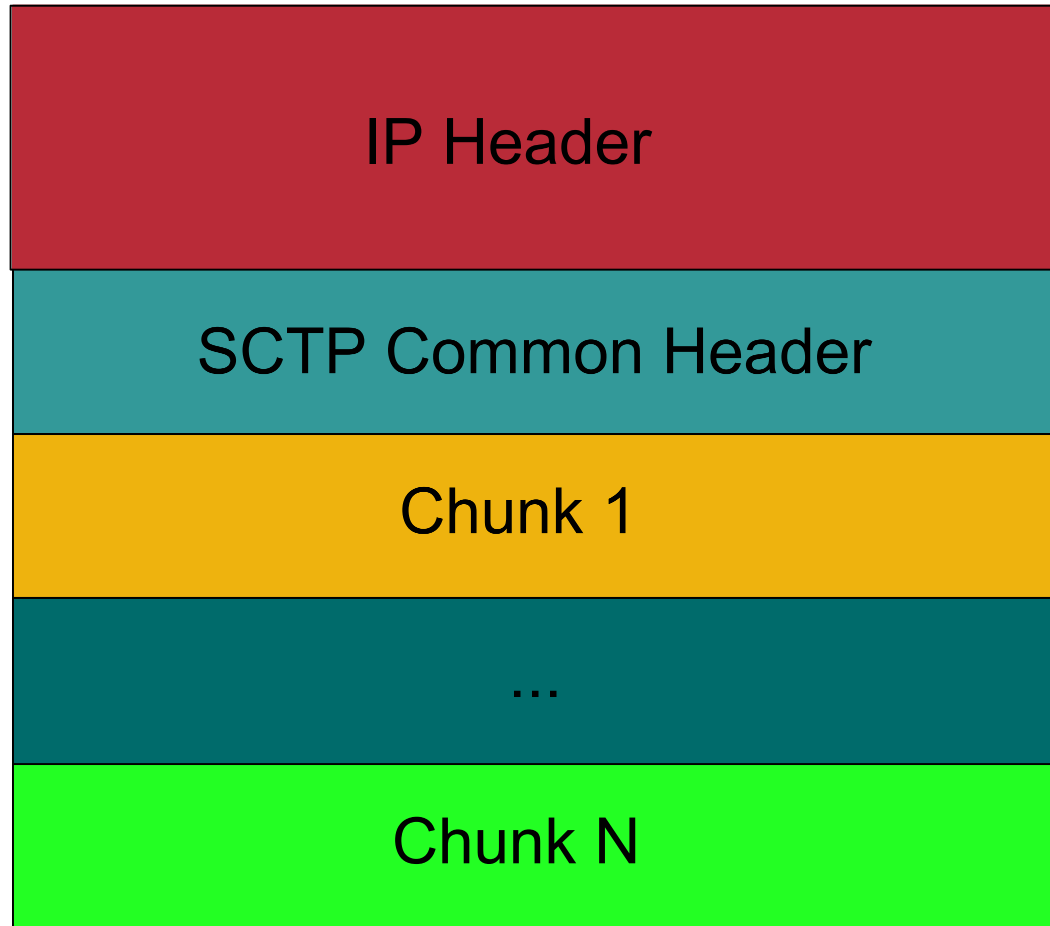
© 2003 Cisco Systems, Inc. All rights reserved.

Bits, Bytes, and Chunks

- We will now turn our attention to 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

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Packet With IP Header



© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Common Header

Source Port	Destination Port
Verification Tag	
CRC-32c Checksum	

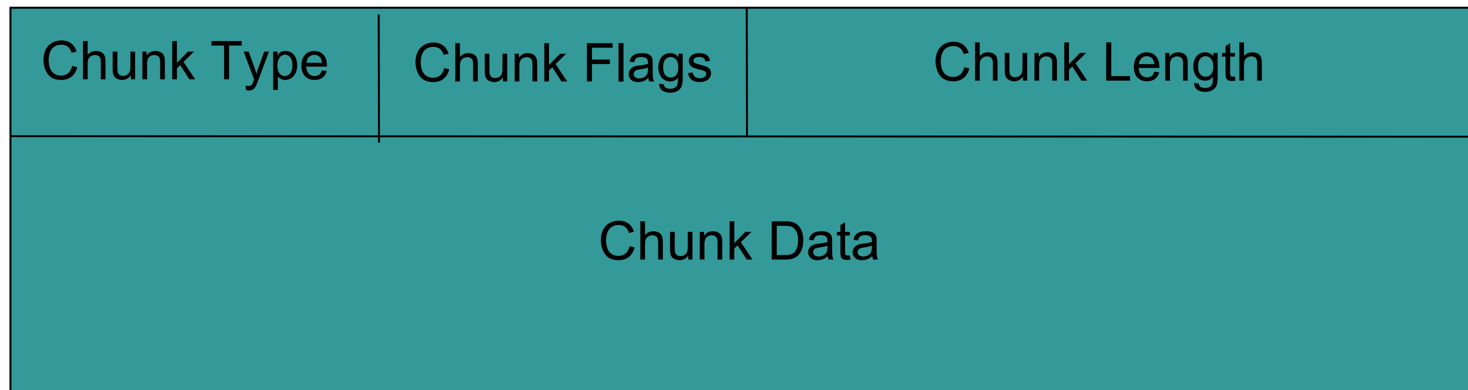
© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Common Header Fields

- **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)

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Chunks



© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Chunk Header Fields

- **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

© 2003 Cisco Systems, Inc. All rights reserved.

List of Chunk Types I

- **There are 20 chunk types currently defined in SCTP (including non-RFC/Internet Draft extensions):**

  (1) DATA (0x00)

  (2) INITIATION [INIT] (0x01)

  (3) INITIATION-ACKNOWLEDGMENT [INIT-ACK] (0x02)

  (4) SELECTIVE-ACKNOWLEDGMENT [SACK] (0x03)

  (5) HEARTBEAT (0x04)

© 2003 Cisco Systems, Inc. All rights reserved.

List of Chunk Types II

(6) HEARTBEAT-ACKNOWLEDGMENT [HEARTBEAT-ACK] (0x05)

(7) ABORT (0x06)

(8) SHUTDOWN (0x07)

(9) SHUTDOWN-ACKNOWLEGMENT [SHUTDOWN-ACK] (0x08)

(10) OPERATIONAL-ERROR [ERROR] (0x09)

(11) COOKIE-ECHO (0x0A)

(12) COOKIE-ACKNOWLEDGMENT [COOKIE-ACK] (0x0B)

© 2003 Cisco Systems, Inc. All rights reserved.

List of Chunk Types III

(13) EXPLICIT CONGESTION NOTIFICATION ECHO [ECNE] (0x0C)

(14) CONGESTION WINDOW REDUCE [CWR] (0x0D)

(15) SHUTDOWN-COMPLETE (0x0E)

© 2003 Cisco Systems, Inc. All rights reserved.

List of Chunks Types: Extensions

- **PR-SCTP - RFC 3758**

  (16) FORWARD-TSN (0xC0)

- **ADD-IP draft**

  (17) ADDRESS-CONFIGURATION [ASCONF] (0xC1)

  (18) ADDRESS-CONFIGURATION-ACKNOWLEDGMENT [ASCONF-ACK] (0x80)

- **Packet-Drop draft**

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

- **Authentication draft**

  (20) AUTHENTICATION [AUTH] (0x82) - about to undergo drastic changes and will probably add 2-3 chunks.

© 2003 Cisco Systems, Inc. All rights reserved.

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
- The chunk type upper two bits have specific meanings used for processing unrecognized chunks

© 2003 Cisco Systems, Inc. All rights reserved.

Chunk Type Processing

- A bit pattern of 00xxxxxx in the chunk type indicates that if this chunk is unknown by the receiver, silently drop it and stop processing the rest of the packet
- A bit pattern of 01xxxxxx in the chunk type indicates that if this chunk is unknown by the receiver, drop it, send an ERROR chunk in reply, and stop processing the rest of the packet

© 2003 Cisco Systems, Inc. All rights reserved.

Chunk Type Processing II

Cisco.com

- A bit pattern of 10xxxxxx in the chunk type indicates that if this chunk is unknown by the receiver, silently skip this chunk but continue to process the rest of the chunks in the packet
- A bit pattern of 11xxxxxx in the chunk type indicates that if this chunk is unknown by the receiver, skip this chunk but send an ERROR chunk in reply and continue to process the rest of the chunks in the packet

© 2003 Cisco Systems, Inc. All rights reserved.

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)

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Chunk Parameters

- Some chunks have parameters within them

    Examples: INIT, INIT-ACK, HEARTBEAT

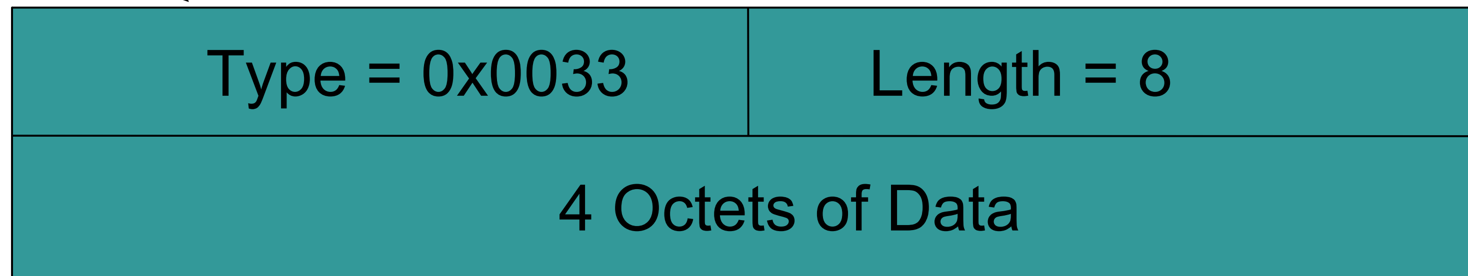
- A parameter also has a TLV format
- A parameter has a similar format to a chunk but slightly different (see the next slide).
- Processing rules for unknown parameters are similar to those for the chunk processing with slightly different connotations

© 2003 Cisco Systems, Inc. All rights reserved.

Parameters Format

Note 16 bit Parameter Type

Note 16 bit length Including the header



The Variable Length Data goes here

© 2003 Cisco Systems, Inc. All rights reserved.

Parameter Handling Rules I

Cisco.com

- The upper 2 bits of the 16 bit parameter is again used to tell an implementation what to do with an unknown parameter

  00xxxxxx-xxxxxxxx : indicates to stop processing the parameter and silently discard this chunk

  01xxxxxx-xxxxxxxx : indicates to stop processing the parameter, report this in an ERROR (or INIT-ACK) chunk, and discard this chunk

© 2003 Cisco Systems, Inc. All rights reserved.

Parameter Handling Rules II

Cisco.com

10xxxxxx-xxxxxxxx : indicates silently skip this parameter, and continue processing the rest of this chunk

11xxxxxx-xxxxxxxx : indicates skip this parameter, report this in an ERROR (or INIT-ACK) chunk, and continue processing the rest of this chunk

- Note that no matter what results from processing each individual parameter, the rest of the chunks in the packet are always processed

© 2003 Cisco Systems, Inc. All rights reserved.

Chunk Details

- We now turn our attention to the individual chunk details.
- We will examine each chunk in the order it would appear in a typical association setup, data exchange and shutdown.
- Extension chunks will be left up to the reader to explore in the individual drafts.

© 2003 Cisco Systems, Inc. All rights reserved.

INIT Chunk

Type=1	Flags=0	Length=variable
Initiation Tag		
Receiver window credit		
# Out Streams		Max # In Streams
Initial TSN		
Optional/Variable length parameters		

© 2003 Cisco Systems, Inc. All rights reserved.

INIT (and INIT-ACK) Chunk Fields

- Initiation Tag: non-zero random 32-bit nonce value
- Receiver Window Credit: initial rwnd used for flow control
- # of Outbound Streams: number of streams the sender wishes to use
- Max # of Inbound Streams: maximum number of streams the sender supports
- Initial TSN: initial 32-bit TSN used for data transfer which is also a random value (it may be copied from the initiation tag)

© 2003 Cisco Systems, Inc. All rights reserved.

INIT / INIT-ACK Chunk Summary

- INIT / INIT-ACK chunks have fixed and variable parts
- The variable part is made up of parameters
- The parameters specify options and features supported by the sender
- Most parameters are valid for both the INIT and the INIT-ACK

© 2003 Cisco Systems, Inc. All rights reserved.

INIT-ACK Chunk

Type=2	Flags=0	Length=variable
Initiation Tag		
Receiver window credit		
# Out Streams	Max # In Streams	
Initial TSN		
Optional/Variable length parameters		

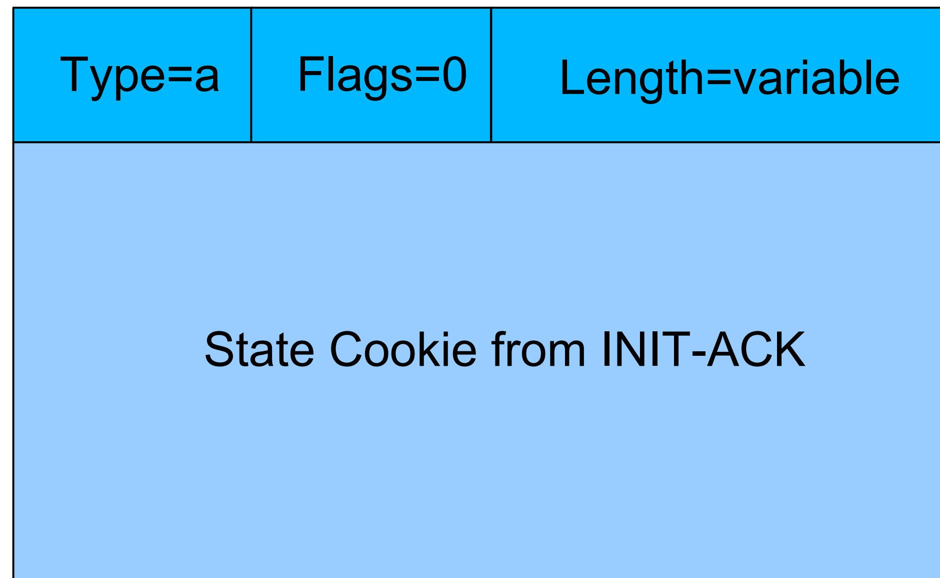
© 2003 Cisco Systems, Inc. All rights reserved.

INIT and INIT-ACK Parameters

PARAMETER	TYPE	INIT	INIT-ACK
IPv4 Address	0x0005	YES	YES
IPv6 Address	0x0006	YES	YES
Cookie Preservative	0x0009	YES	NO
ECN Capable	0x8000	YES	YES
Hostname Address	0x000B	YES	YES
Supported Address Types	0x000C	YES	YES
Unrecognized Parameters	0x0008	NO	YES
State Cookie	0x0007	NO	YES
PR-SCTP Supported	0xC001	YES	YES
Set Primary Address	0xC004	YES	YES
Adaption Layer Indication	0XC006	YES	YES

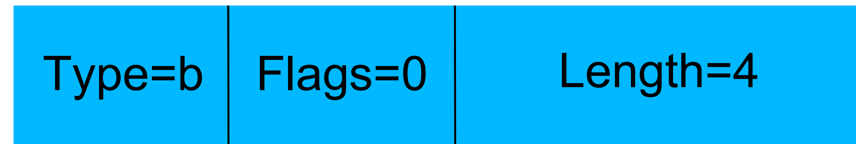
© 2003 Cisco Systems, Inc. All rights reserved.

Cookie Echo Chunk



© 2003 Cisco Systems, Inc. All rights reserved.

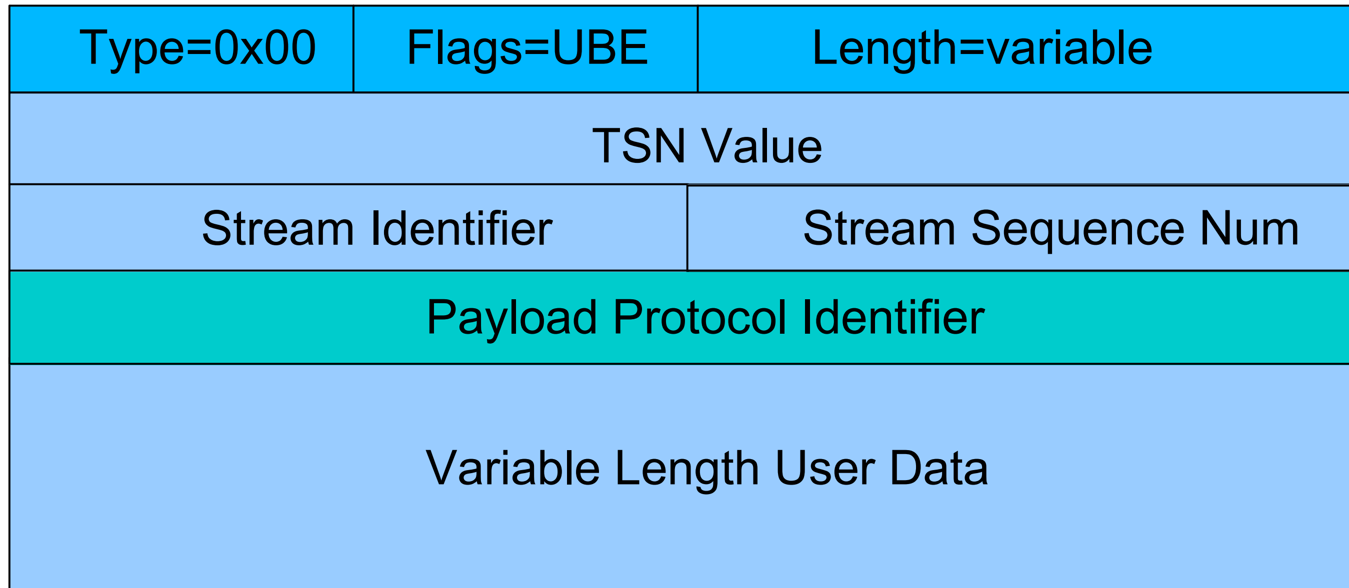
Cookie Ack Chunk



- **The Cookie-Echo and Cookie-ACK are simplistic chunks, but help prevent resource attacks**
- **They serve as the last part of the 4-way handshake that sets up an SCTP association**
- **Both allow bundling with other chunks, such as DATA**

© 2003 Cisco Systems, Inc. All rights reserved.

DATA Chunk



- **Flag Bits ‘UBE’ are used to indicate:**

  U – Unordered Data

  B – Beginning of Fragmented Message

  E – End of Fragmented Message

- **A user message that fits in one chunk would have both the B and E bits set**

© 2003 Cisco Systems, Inc. All rights reserved.

DATA Chunk Fields

- **TSN**: transmission sequence number used for ordering and reassembly and retransmission
- **Stream Identifier**: the stream number for this DATA
- **Stream Sequence Number**: identifies which message this DATA belongs to for this stream
- **Payload Protocol Identifier**: opaque value used by the endpoints (and perhaps network equipment)
- **User Data**: the user message (or portion of)

© 2003 Cisco Systems, Inc. All rights reserved.

SACK Chunk

Type=3	Flags=0	Length=variable	
Cumulative TSN			
Receiver window credit			
Num of Fragments=N		Num of Dup=M	
Gap Ack Blk #1 start		Gap Ack Blk #1 end	
Gap Ack Blk #N start		Gap Ack Blk #N end	
Duplicate TSN #1			
Duplicate TSN #M			

© 2003 Cisco Systems, Inc. All rights reserved.

SACK Chunk Fields

- **Cumulative TSN Acknowledgment**: the highest consecutive TSN that the SACK sender has received

  a.k.a. cumulative ack (cum-ack) point

- **Receiver Window Credit**: current rwnd available for the peer to send

- **# of Fragments**: number of Gap Ack Blocks included

- **# of Duplicates**: number of Duplicate TSN reports included

© 2003 Cisco Systems, Inc. All rights reserved.

SACK Chunk Fields II

- **Gap Ack Block Start / End TSN offset**: the start and end offset for a range of consecutive TSNs received relative to the cumulative ack point

    The TSNs not covered by a Gap Ack Block indicate TSNs that are "missing"

- **Duplicate TSN**: TSN that has been received more than once

    Note that the same TSN may be reported more than once

© 2003 Cisco Systems, Inc. All rights reserved.

SACK Chunk Example

Type=3	Flags=0	Length=variable	
Cum Ack=109965			
rwnd = 64200			
Num of Fragments=2		Num of Dup=2	
Gap start = 2		Gap end = 5	
Gap start = 7		Gap end = 9	
Duplicate TSN = 109963			
Duplicate TSN = 109964			

© 2003 Cisco Systems, Inc. All rights reserved.

SACK Example Dissected

- The sender’s cum-ack point is 109,965
- The sender has received TSN's 109,967 – 109,970
- The sender has received TSN's 109,972 – 109,974
- The sender is missing 109,966 and 109,971.
- The sender received duplicate transmissions of 109,963 and 109,964
- Question: Would you ever see a Gap Ack start of 1?

© 2003 Cisco Systems, Inc. All rights reserved.

Heartbeat Chunk

Type=4	Flags=0	Length=variable
Param Type = 1		Length=variable
Heartbeat Data		

- **Data within the Heartbeat Data parameter is implementation specific**

© 2003 Cisco Systems, Inc. All rights reserved.

Heartbeat Ack Chunk

Type=5	Flags=0	Length=variable
Param Type = 1		Length=variable
Heartbeat Data		

- **Data within the Heartbeat Data parameter is implementation specific and is a straight echo of what was received in the Heartbeat chunk**

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Chunks

Type=7	Flags=0	Length=8
Cumulative TSN		

SHUTDOWN

Type=8	Flags=0	Length=4

SHUTDOWN-ACK

Type=14	Flags=T	Length=4

SHUTDOWN-COMPLETE

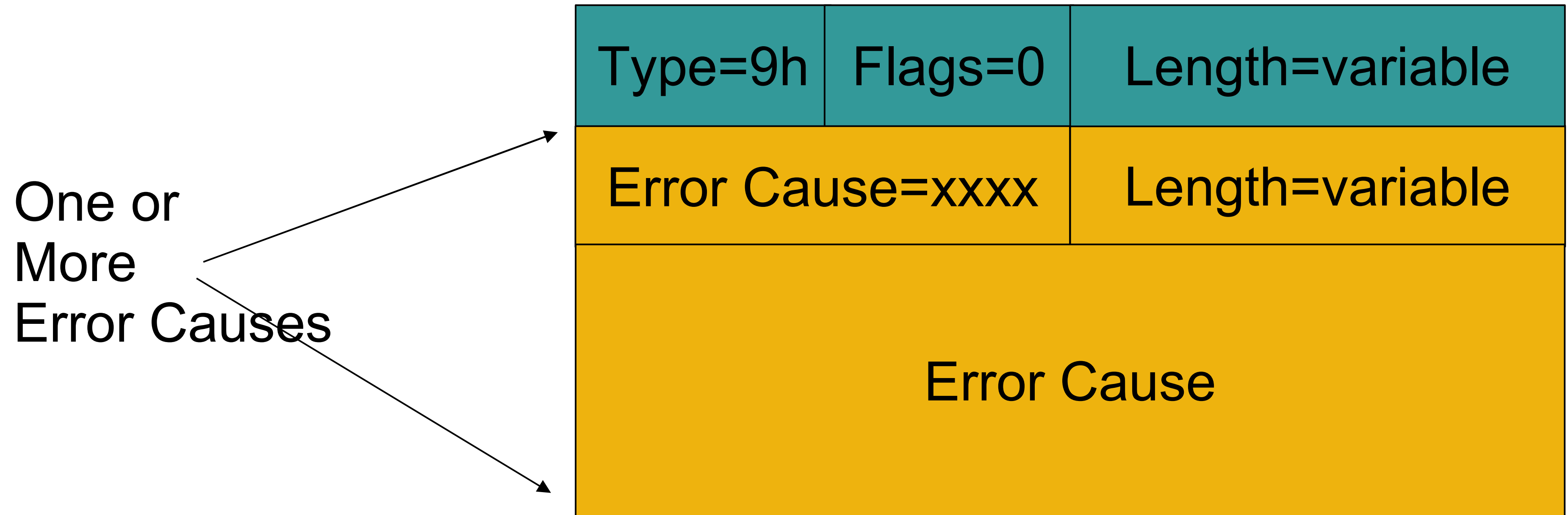
© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Chunk Fields

- **The SHUTDOWN chunk also carries a Cumulative TSN Acknowledgment field to indicate the highest TSN that the SHUTDOWN sender has seen.**
- **A SACK chunk may be bundled to give a more complete picture (e.g. Gap Ack Blocks) of the sender's receive state.**

© 2003 Cisco Systems, Inc. All rights reserved.

Operational Error Chunk



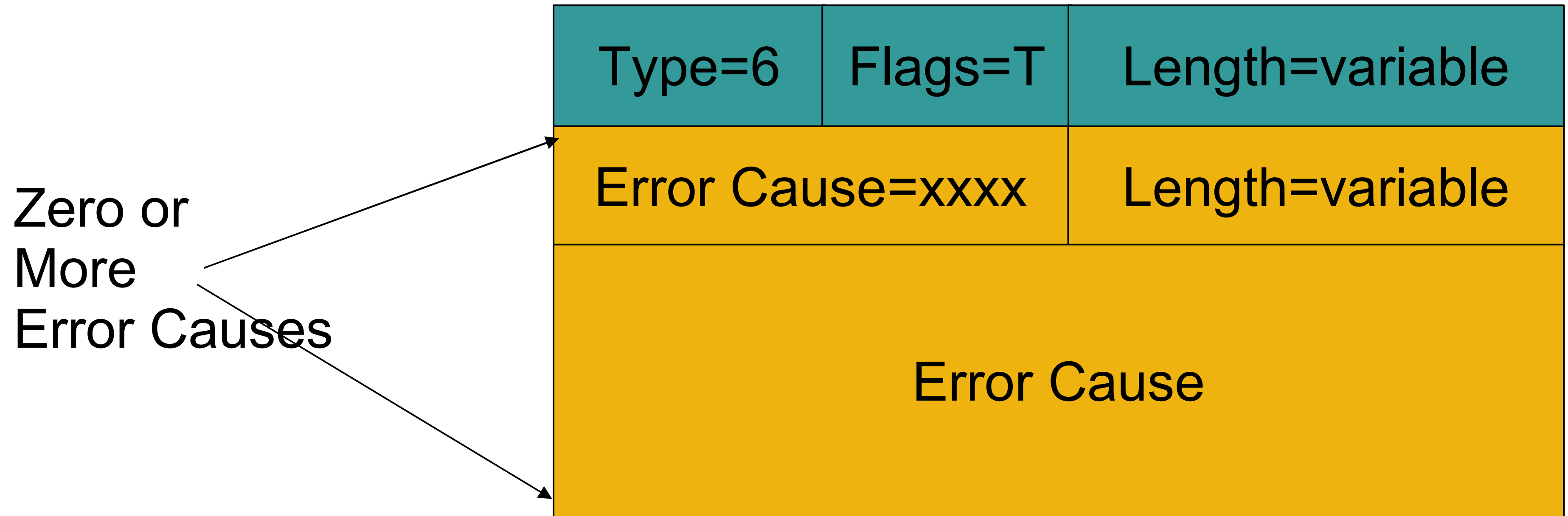
© 2003 Cisco Systems, Inc. All rights reserved.

Summary of Error Causes

Error Cause	Type Value
Invalid Stream Identifier	0x0001
Missing Mandatory Parameter	0x0002
Stale Cookie Error	0x0003
Out of Resource	0x0004
Unresolvable Address	0x0005
Unrecognized Chunk Type	0x0006
Invalid Mandatory Parameter	0x0007
Unrecognized Parameter Type	0x0008
No User Data	0x0009
Cookie Received While Shutting Down	0x0010
Restart of Association With New Addresses	0x0011
User Initiated Abort	0x0012
Protocol Violation	0x0013

© 2003 Cisco Systems, Inc. All rights reserved.

Abort Chunk



© 2003 Cisco Systems, Inc. All rights reserved.

The T-bit

Cisco.com

- Both the SHUTDOWN-COMPLETE and ABORT chunk use one flag value
- The T bit is the first bit: i.e.: binary -------x
- When this bit is set to 0, the sender has a TCB and the V-Tag (in the common header) is the correct one for the association.
- When this bit is set to 1, the sender has NO TCB and the V-Tag is set to what was in the V-Tag value of the packet that is being responded to.

© 2003 Cisco Systems, Inc. All rights reserved.

Forward-TSN Chunk

Type=192	Flags=0	Length=variable
New Cumulative TSN		
Stream Id 1	Stream Seq 1	
Stream Id N	Stream Seq N	

© 2003 Cisco Systems, Inc. All rights reserved.

Forward-TSN Chunk Fields

- **New Cumulative TSN**: the new cumulative ack point that the receiver should move forward (skip) to

  Treat all TSNs up to this new point as having been received

- **Stream Identifier/Stream Sequence Number**: the largest stream sequence number being skipped for a given stream

- Multiple Stream Identifier-Sequence Number pairs may be included if the Forward TSN covers multiple messages

© 2003 Cisco Systems, Inc. All rights reserved.

Forward TSN Operation

- **Used to move the cumulative ack point forward without retransmitting data.**

  Note the receiver could move the point forward further if the Forward TSN skips past a missing block of TSNs

- **Has zero or more stream and sequence numbers listed to help a receiver free stranded data.**
- **Is part of the soon to be RFC'd PR-SCTP document.**

© 2003 Cisco Systems, Inc. All rights reserved.

Other Extensions

- **Several SCTP extensions exist**
- **Packet Drop is a Cisco originated extension that inter-works the router with the endpoint.**
- **ADD-IP allows for dynamic addition and subtraction of IP addresses**
- **AUTH allows for two endpoints to negotiate the signing of specific chunks (such as ADD-IP chunks). It uses the Purpose Built Key's (PBK) draft**

© 2003 Cisco Systems, Inc. All rights reserved.

Parameters and Error Causes

- RFC 2960 lays out all the basic data formats
- The SCTP book on pages 47-55 also hold illustrations of the various chunk layouts and details.
- Error causes are also in the RFC and can also be found on pages 65-73 of the SCTP book
- The SCTP Implementors Guide (draft) contains a few new parameters mentioned previously
- We will let your curiosity guide you in viewing these bits and bytes if your interested

© 2003 Cisco Systems, Inc. All rights reserved.

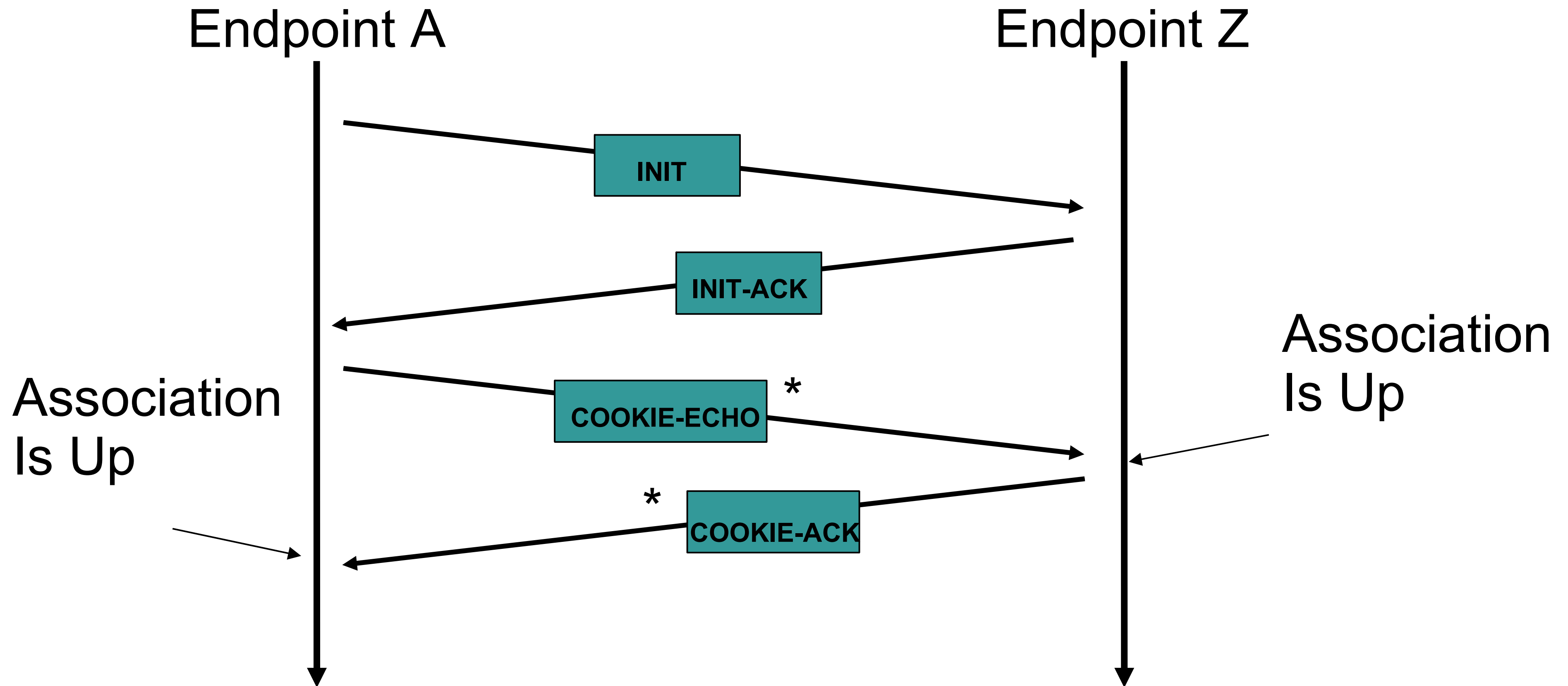
Questions

Cisco.com

- **Questions before we break**
- **In the next sections, we will begin going through the protocol operation details**

© 2003 Cisco Systems, Inc. All rights reserved.

Setting Up an Association



* -- User data can be attached

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Association Setup

- SCTP uses a four-way handshake to set up an association
- The side doing the active (or implicit) open will formulate and send an INIT chunk
- The sender of the INIT includes various parameters:

  IPv4 and IPv6 address parameters identifying all bound addresses within the peer’s scope

  Extensions such as PR-SCTP, Adaption Layer Indication and possibly a Supported Address list

  There could also be cookie preservatives and other sundry items as well

© 2003 Cisco Systems, Inc. All rights reserved.

Sending an INIT

- Two important random values that a sender of an INIT (and an INIT-ACK) generates:

  A Verification Tag (V-Tag) will provide the peer with a nonce that must be present in every packet sent (this is placed in the initiate tag field)

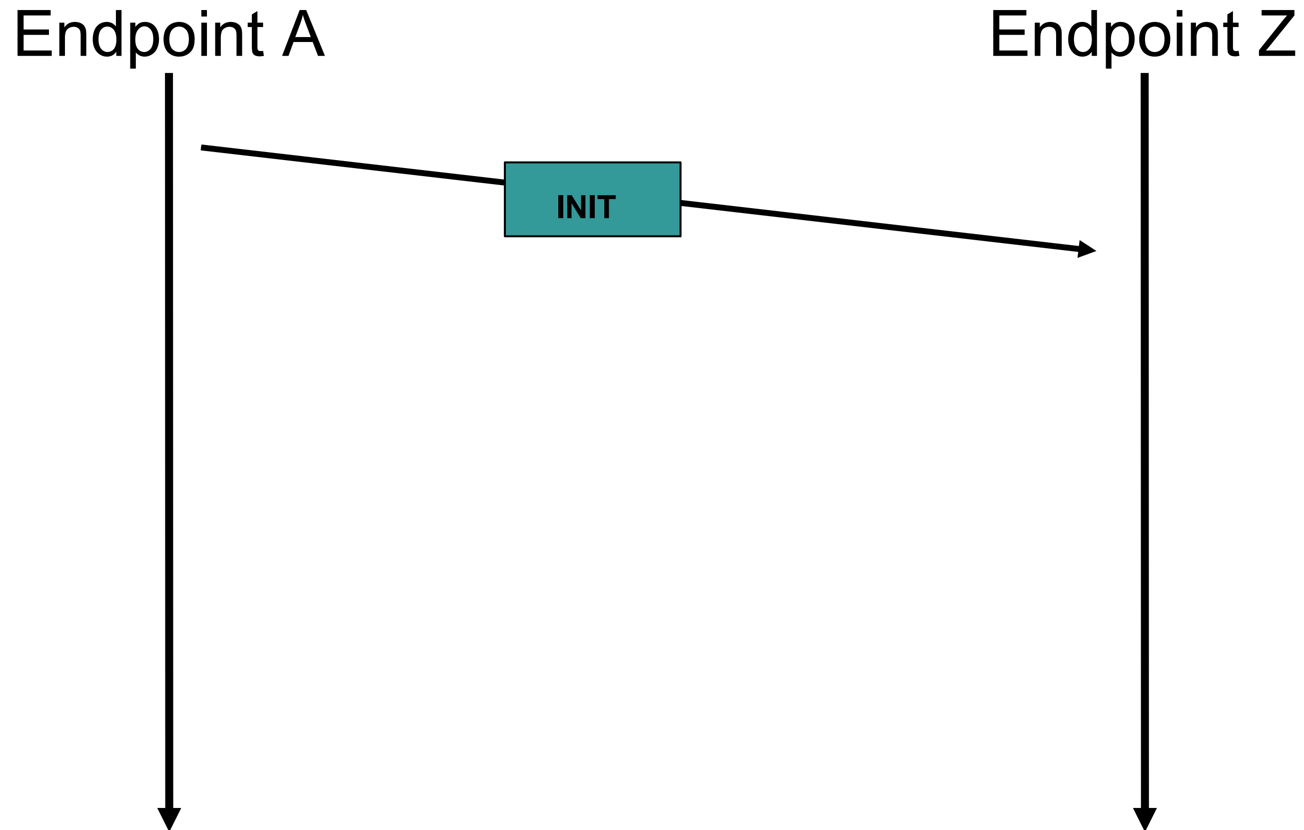
  An Initial TSN provides the starting point for the transport sequence space

- The V-Tag provides modest security for the association and also removes the need for a psuedo-header in the checksum

© 2003 Cisco Systems, Inc. All rights reserved.

The INIT is in Flight

Cisco.com



© 2003 Cisco Systems, Inc. All rights reserved.

Receiving an INIT

- The receiver of the INIT will validate that a listener exists for the destination port. If not, it will send an ABORT back to the sender.
- It may do some checking and validation, but in general it will always send back an INIT-ACK saving NO state. This prevents SCTP from being subject to the TCP SYN-like attacks.
- In formulating an INIT-ACK, the responder will include all the various parameters just like what a sender does when formulating an INIT, but with one important addition.

© 2003 Cisco Systems, Inc. All rights reserved.

Formulating the INIT-ACK Response

- The receiver of the INIT MUST include a state cookie parameter in the INIT-ACK response.
- The state cookie parameter:

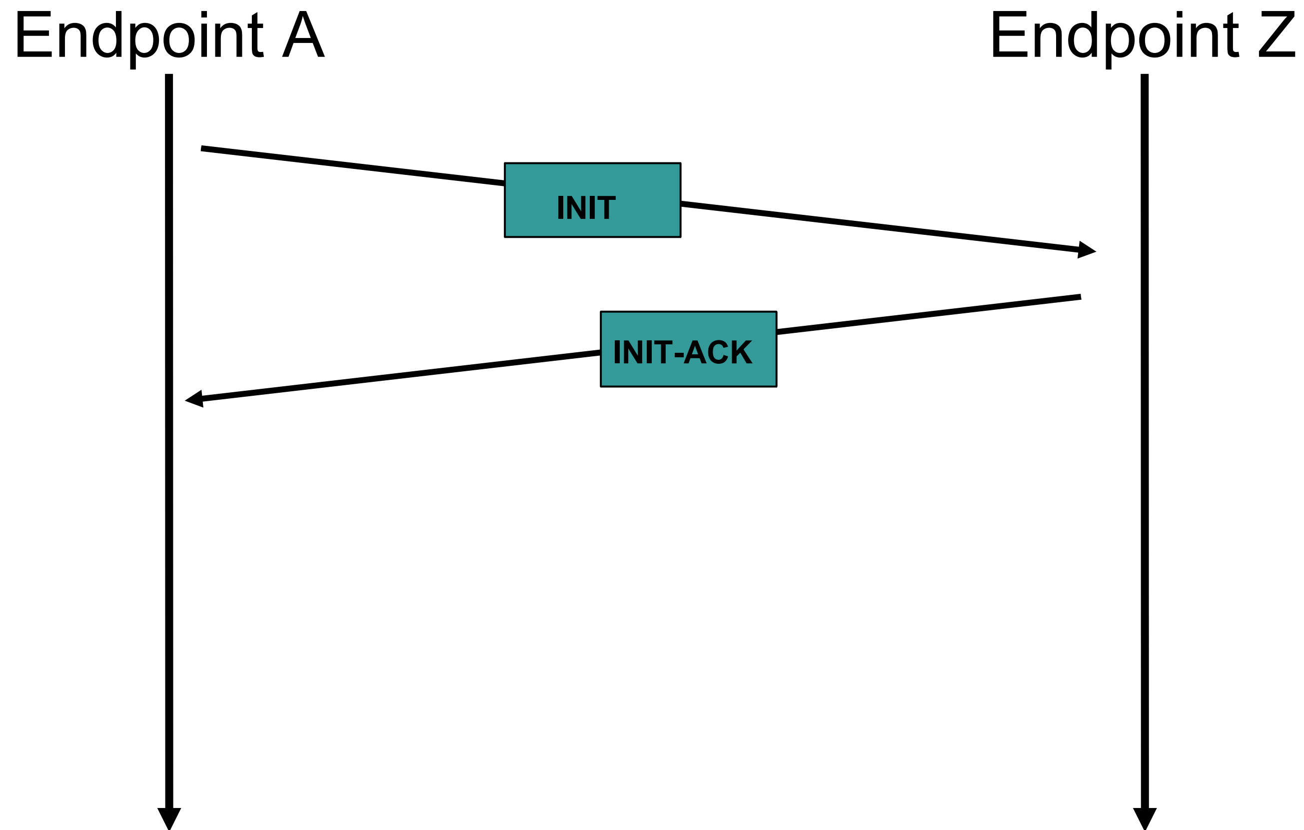
  Is signed (usually with MD5 or SHA-1)

  Contains ALL the state needed to setup the association (usually the entire INIT and some pieces of the INIT-ACK)

  Is implementation specific, but must include a timestamp
- Page 86-88 of the SCTP reference book goes into more details of state cookie generation

© 2003 Cisco Systems, Inc. All rights reserved.

Back Goes the INIT-ACK



© 2003 Cisco Systems, Inc. All rights reserved.

When the INIT-ACK Arrives…

- The receiver of the INIT-ACK must take special care in finding the association for the endpoint that sent the INIT.
- In particular it must look at the address list inside the INIT-ACK in case the source address is not the same as where the INIT was sent.
- After finding the association, the receiver will add all of the peer’s information (addresses, V-Tag, initial sequence number, etc.) to the local TCB.

© 2003 Cisco Systems, Inc. All rights reserved.

More on Processing the INIT-ACK

- **At this point the receiver must reply back with a COOKIE-ECHO chunk:**

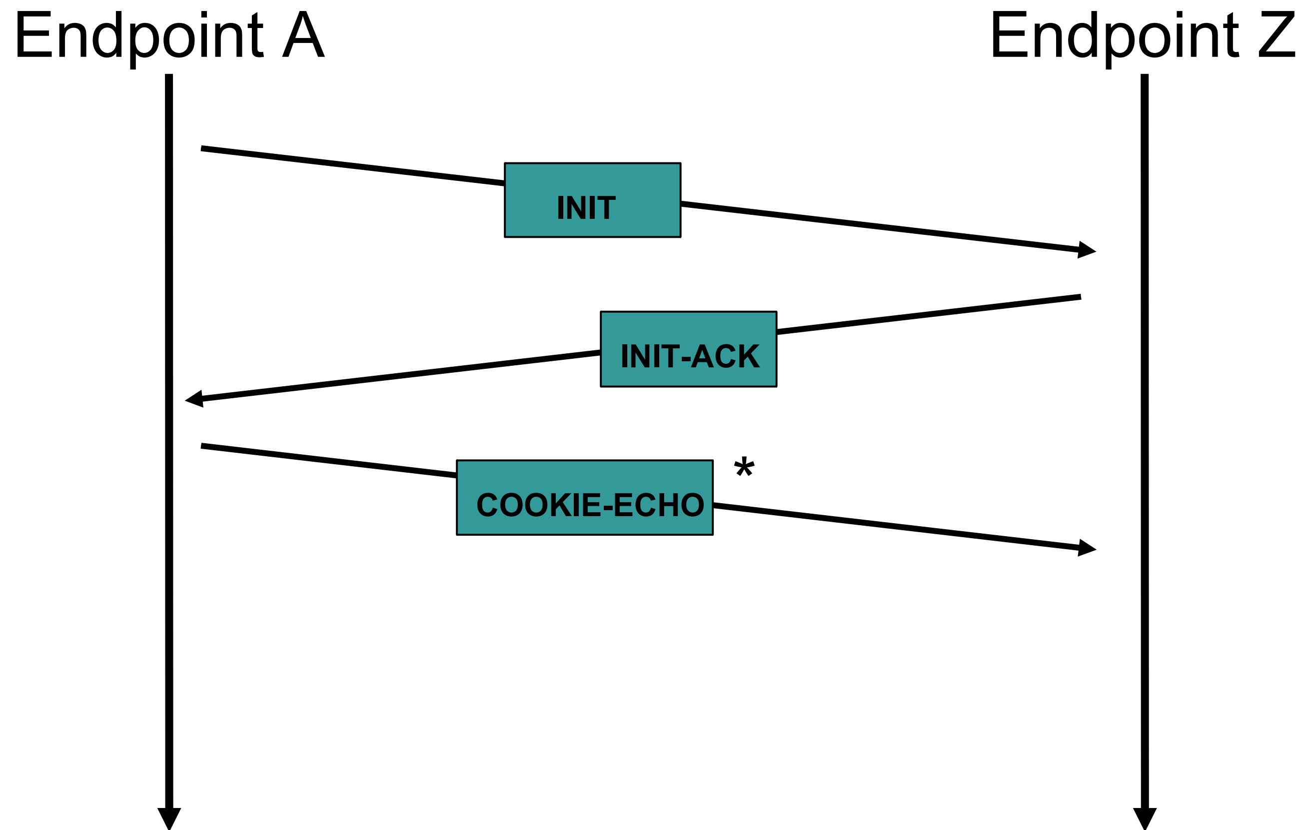
  The cookie is retrieved by simply finding the state-cookie parameter and changing the first two bytes into the chunk type and flags field (set to 0) of the COOKIE-ECHO chunk.

  This chunk is sent back to the source address of the INIT-ACK packet.

  As long as the COOKIE-ECHO chunk is first in the packet, any queued DATA chunks may be bundled in the SCTP packet.

© 2003 Cisco Systems, Inc. All rights reserved.

Feed the Peer a Cookie

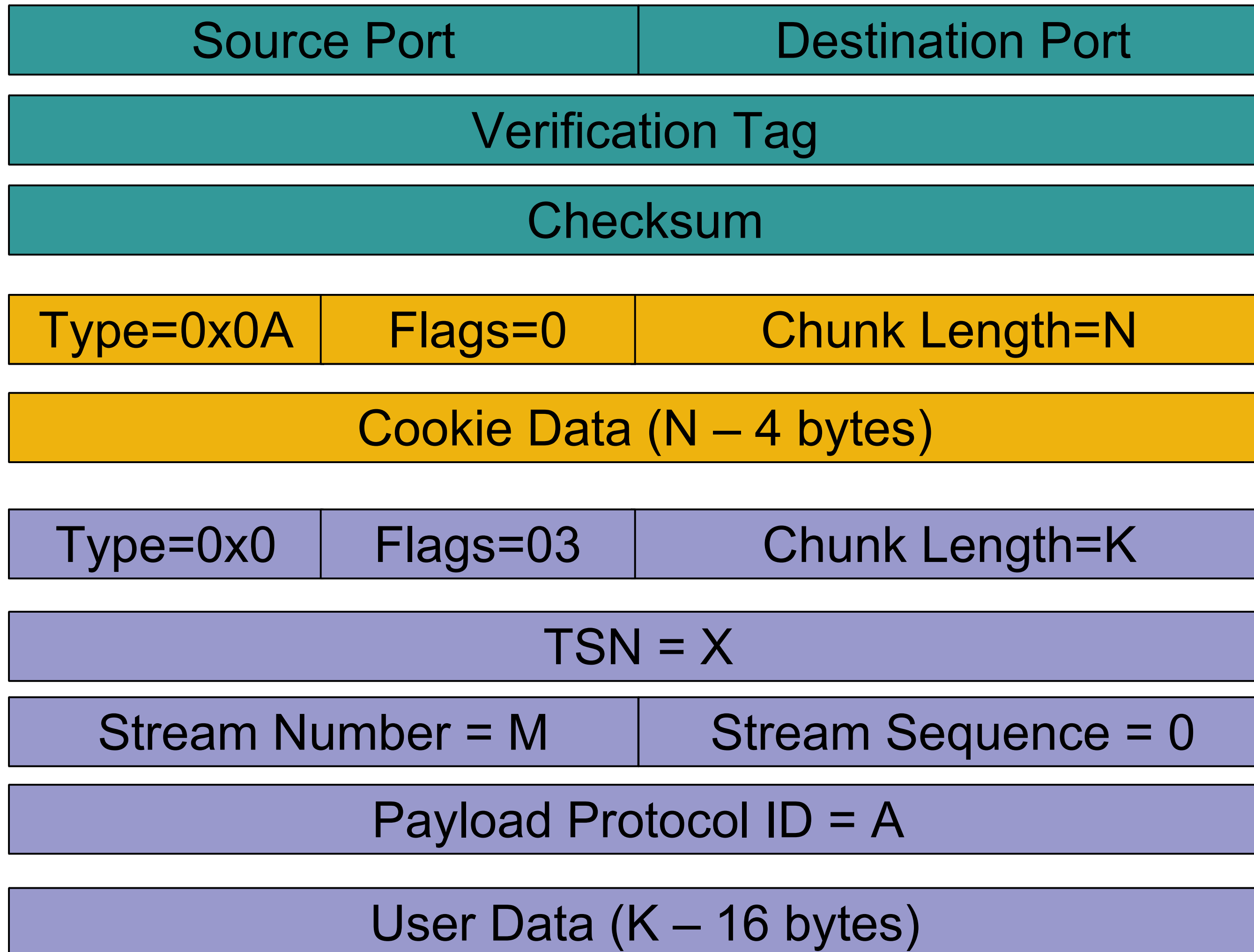


* -- User data can be attached

© 2003 Cisco Systems, Inc. All rights reserved.

A Packet with the COOKIE-ECHO

Cisco.com



© 2003 Cisco Systems, Inc. All rights reserved.

Processing the Cookie-Echo

- First, validate that the state cookie has not been modified by running the hash over it and the internal secret key. If they do not match, the cookie is silently discarded.
- Next, the timestamp field in the cookie is checked. If it proves to be an old cookie, a stale cookie error is sent to the peer.
- Otherwise, the cookie is used to create a new TCB.
- The association now enters the ESTABLISHED state.

© 2003 Cisco Systems, Inc. All rights reserved.

More on Cookie Processing

- **Note that this quick summary assumes a normal non-collision, non-restart case. Collision cases are accounted for in the specification.**
- **After the cookie is processed and the TCB is created, the endpoint then processes any additional chunks contained in the packet.**
- **Note that the additional chunks are processed in the ESTABLISHED state, since the cookie processing was completed.**

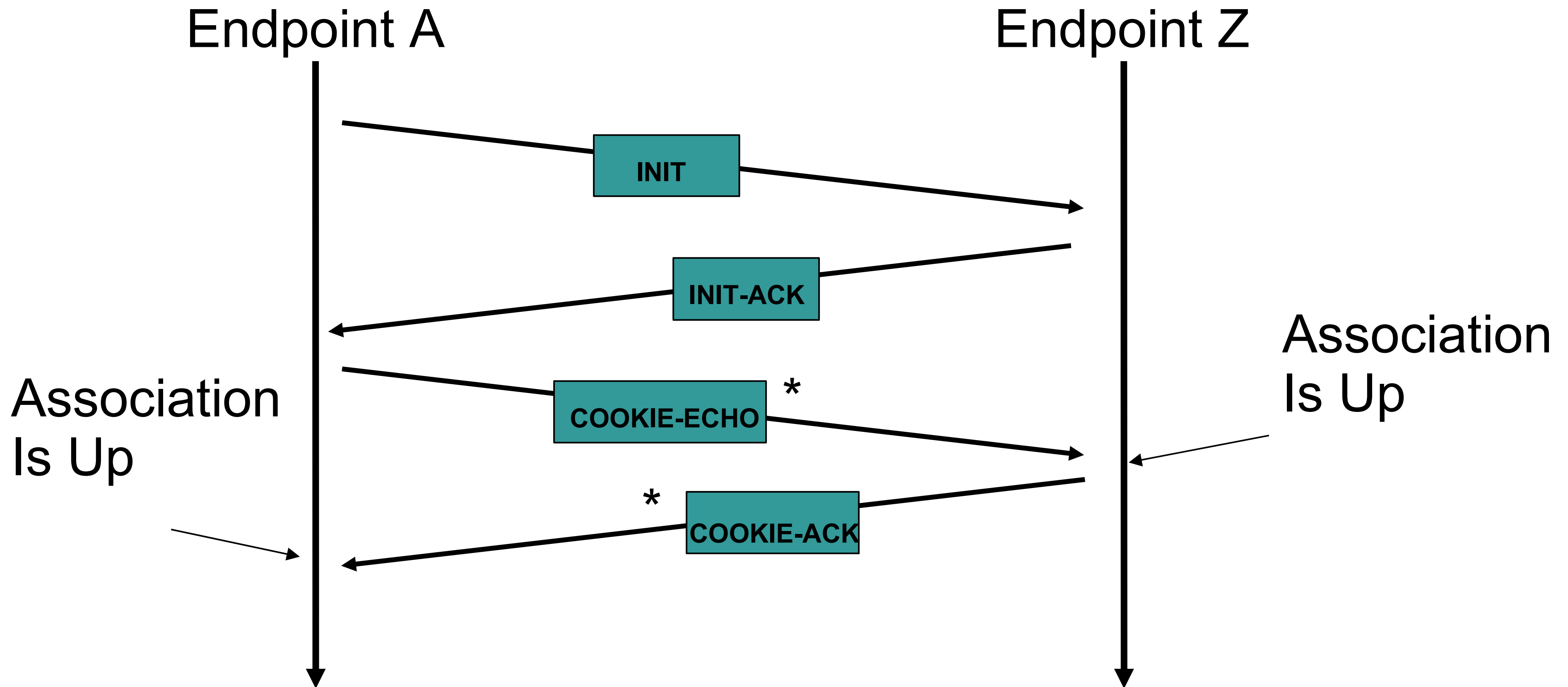
© 2003 Cisco Systems, Inc. All rights reserved.

Acknowledge the Eaten Cookie

- After the packet with the COOKIE-ECHO is fully processed, a COOKIE-ACK response is sent back.
- At this point, any other chunks (DATA, SACK, etc) can also be bundled with the COOKIE-ACK.
- One final interesting note, most implementations will include within the state cookie the address to which the INIT-ACK was sent. This is due to the fact that this address will be the only one that is considered “confirmed” initially.

© 2003 Cisco Systems, Inc. All rights reserved.

Association Completed



* -- User data can be attached

© 2003 Cisco Systems, Inc. All rights reserved.

Other Association Setup Issues to Consider

- The SCTP book contains additional material regarding INIT and INIT-ACK chunks.
- A large set of special case handling is described in section 4.7 (pages 103 – 122) of the SCTP reference book. These cases deal with collisions and restarts.
- We will walk through the restart case (4.7.4) and discuss tie-tags briefly.
- Refer to the SCTP book for details on all of the other cases (it is the only place that such collisions are documented to my knowledge).

© 2003 Cisco Systems, Inc. All rights reserved.

Association Restart

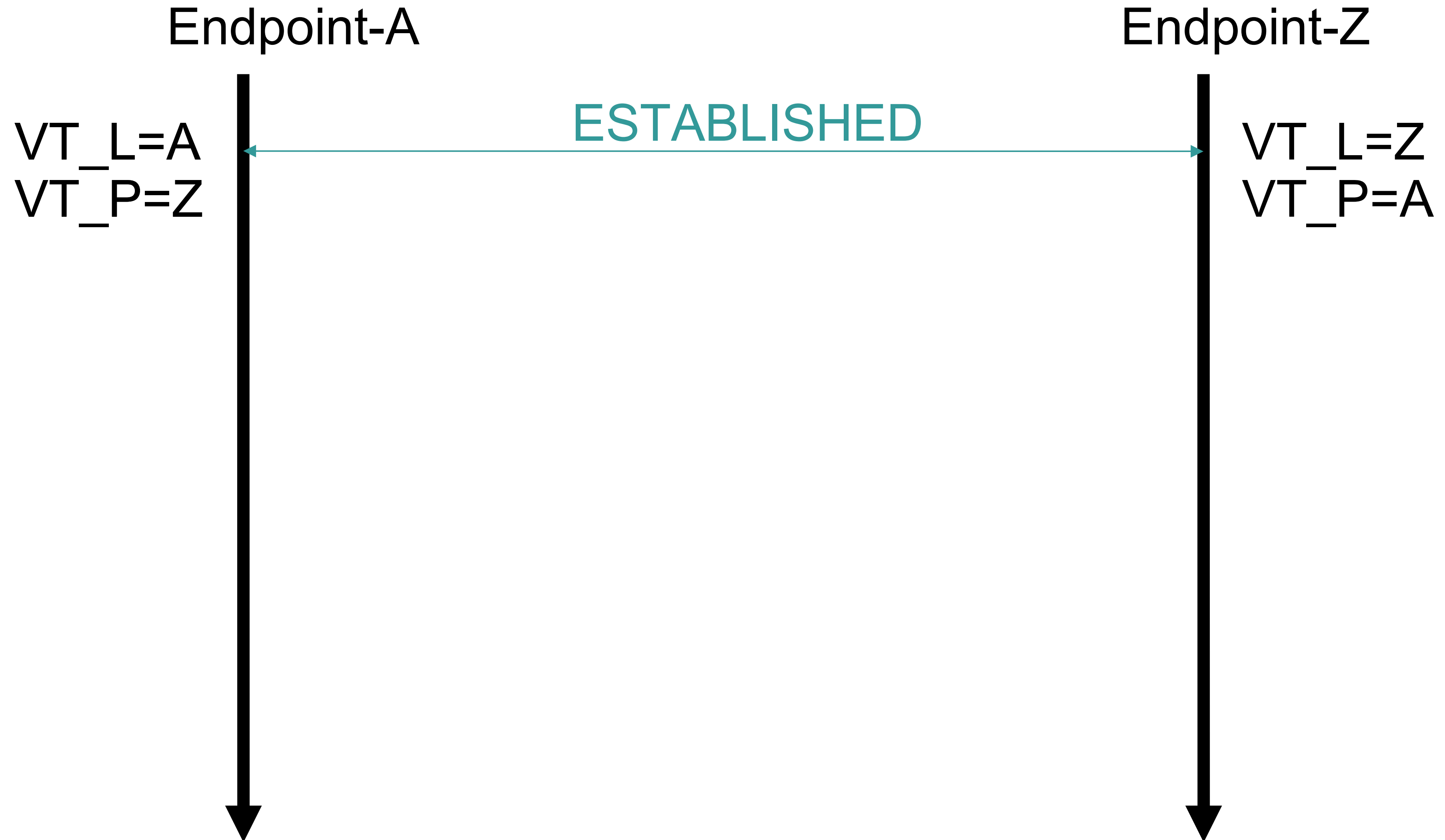
- **An association restart occurs when a peer crashes and restarts rapidly.**
- **The restart and association re-establish attempt must occur before the non-restarting peer's HEARTBEAT is sent.**

  (HEARTBEAT's are discussed later)

- **We start our scenario with the following picture:**

© 2003 Cisco Systems, Inc. All rights reserved.

Restart: Initial Condition



© 2003 Cisco Systems, Inc. All rights reserved.

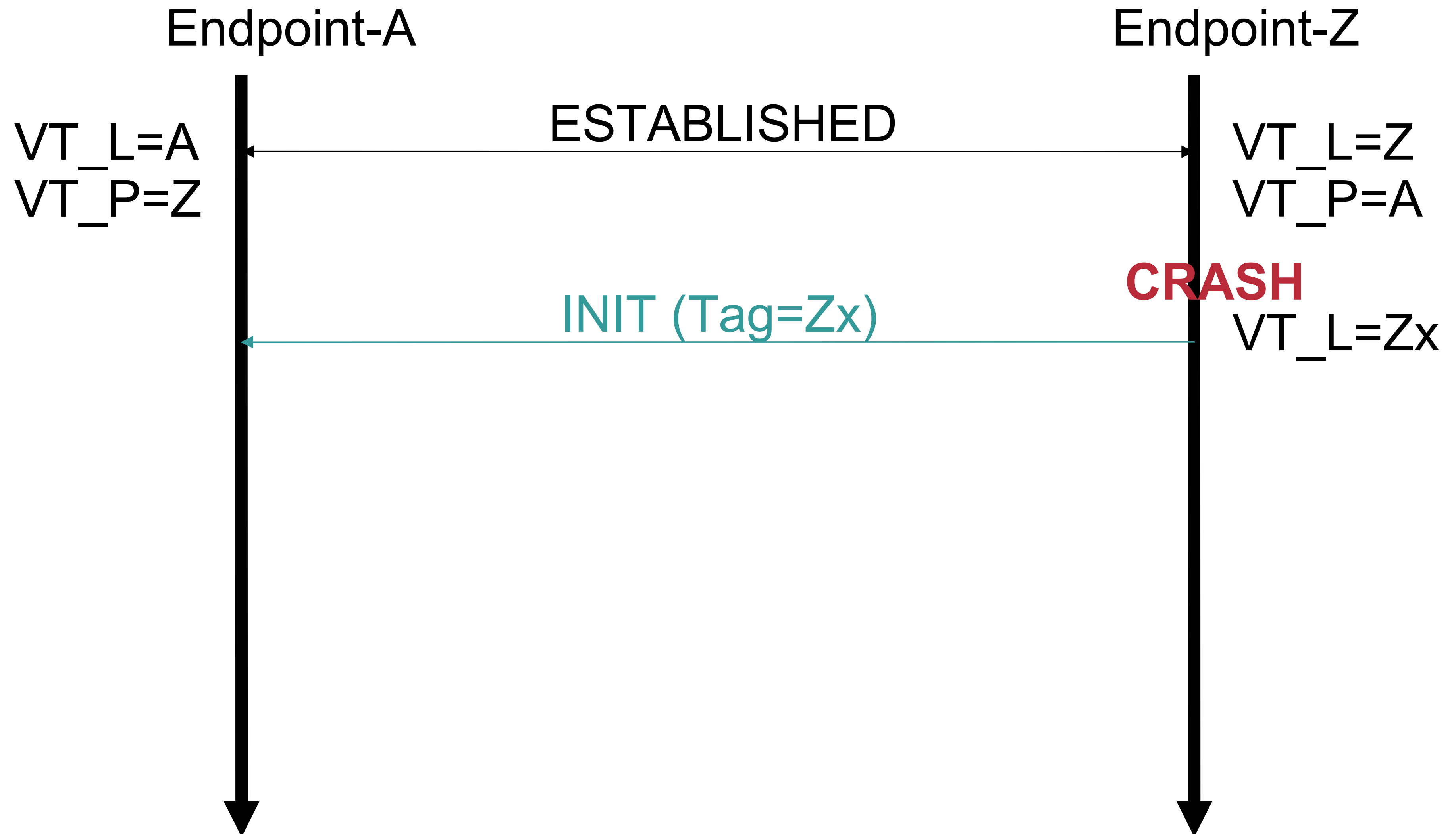
Restart: Initial Condition Description

- Peers Endpoint-A and Endpoint-Z have their association in the ESTABLISHED state.
- VT_L (Verification Tag Local) is the value that the endpoint expects in each V-Tag for each received packet.
- VT_P (Verification Tag Peer) is the value that each endpoint will send as the V-Tag in every packet.
- So, if Endpoint-A sends a packet to Endpoint-Z, it places "Z" in the V-Tag field of the common header.

© 2003 Cisco Systems, Inc. All rights reserved.

Restart: The CRASH

Cisco.com



© 2003 Cisco Systems, Inc. All rights reserved.

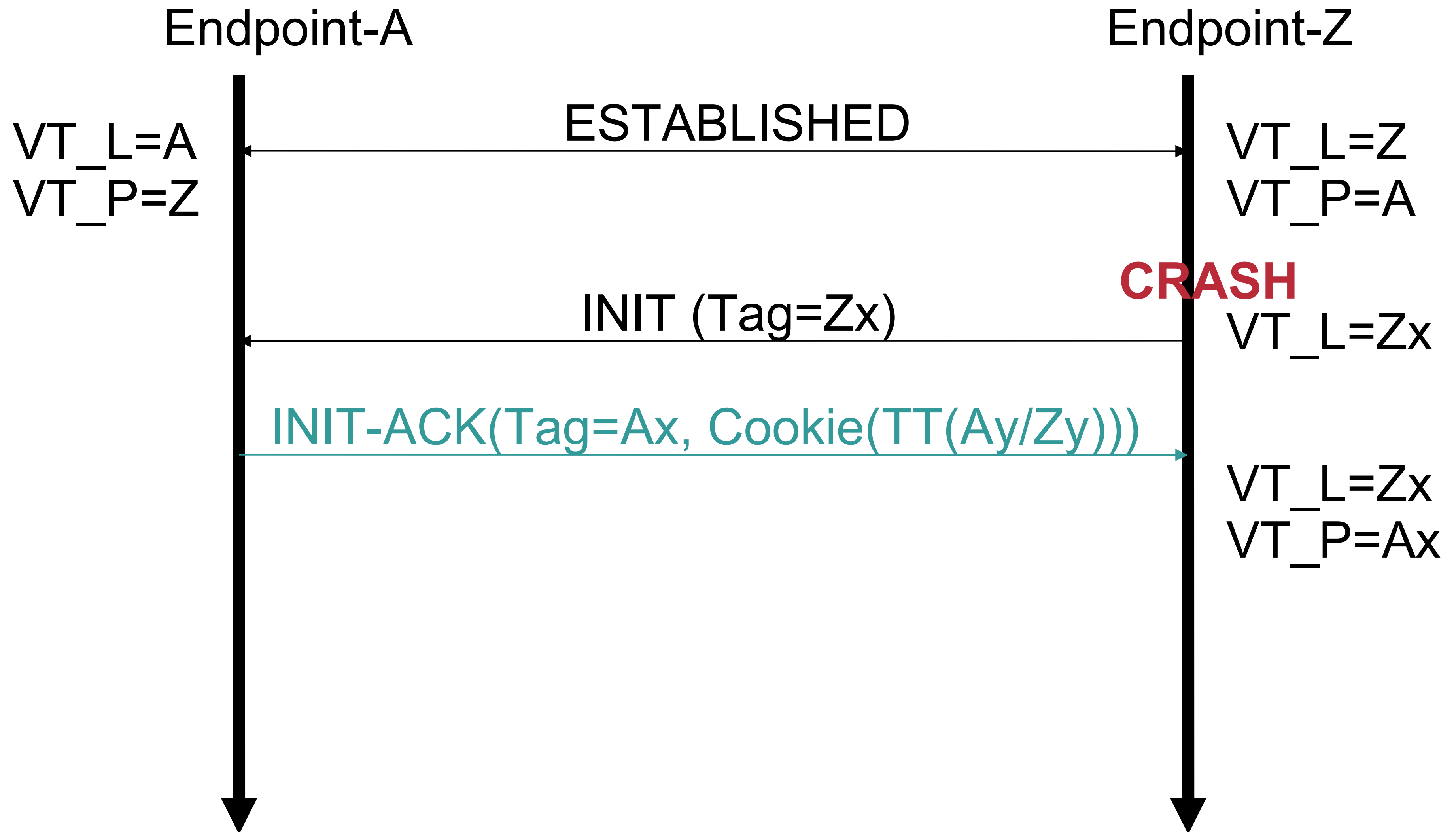
Restart: The Crash Described

- Endpoint-Z suddenly crashes and restarts.
- After the application restarts, it (re-)attempts to setup an association with Endpoint-A using the same local SCTP transport addresses
- Endpoint-Z chooses a new random tag “Zx” and sends off a new INIT to its ‘potential’ peer

  Remember, Endpoint-Z’s SCTP stack is un-aware of the previous association

© 2003 Cisco Systems, Inc. All rights reserved.

Restart: Hmm… A New Association?



© 2003 Cisco Systems, Inc. All rights reserved.

Restart: Handling the New INIT

- Endpoint-A receives the new INIT from its peer out of the blue.
- Endpoint-A cannot necessarily trust this INIT since the V-Tag it expects in every packet is NOT present (since Endpoint-Z restarted).
- Endpoint-A will respond with an INIT-ACK with:

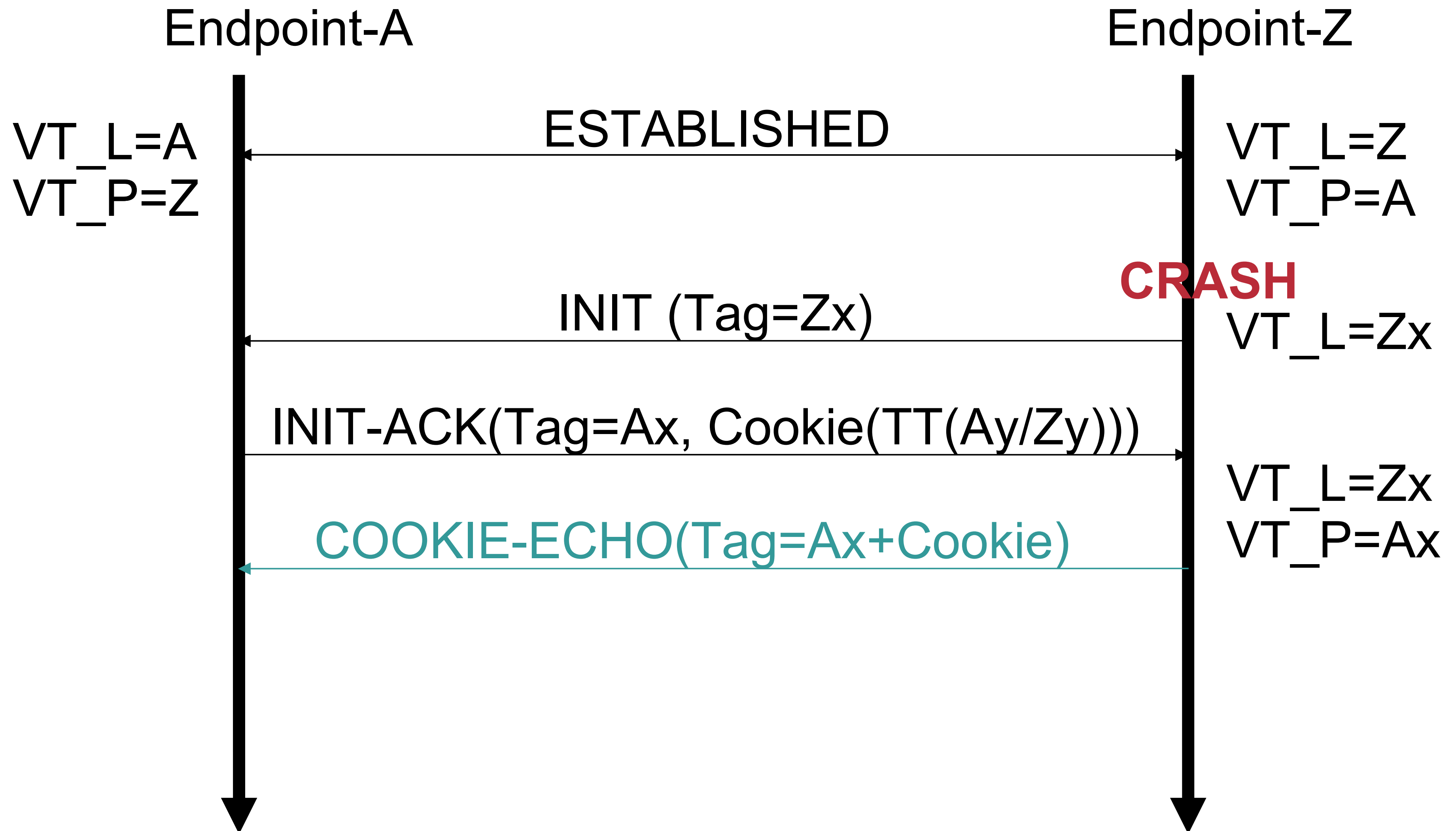
  A new random verification tag (Ax)

  Two new random Tie-Tags (Ay and Zy) sent in the state cookie (and also stored in the TCB)

© 2003 Cisco Systems, Inc. All rights reserved.

Restart: Everything Normal (Sort-of)

Cisco.com



© 2003 Cisco Systems, Inc. All rights reserved.

Restart: Tie-Tags

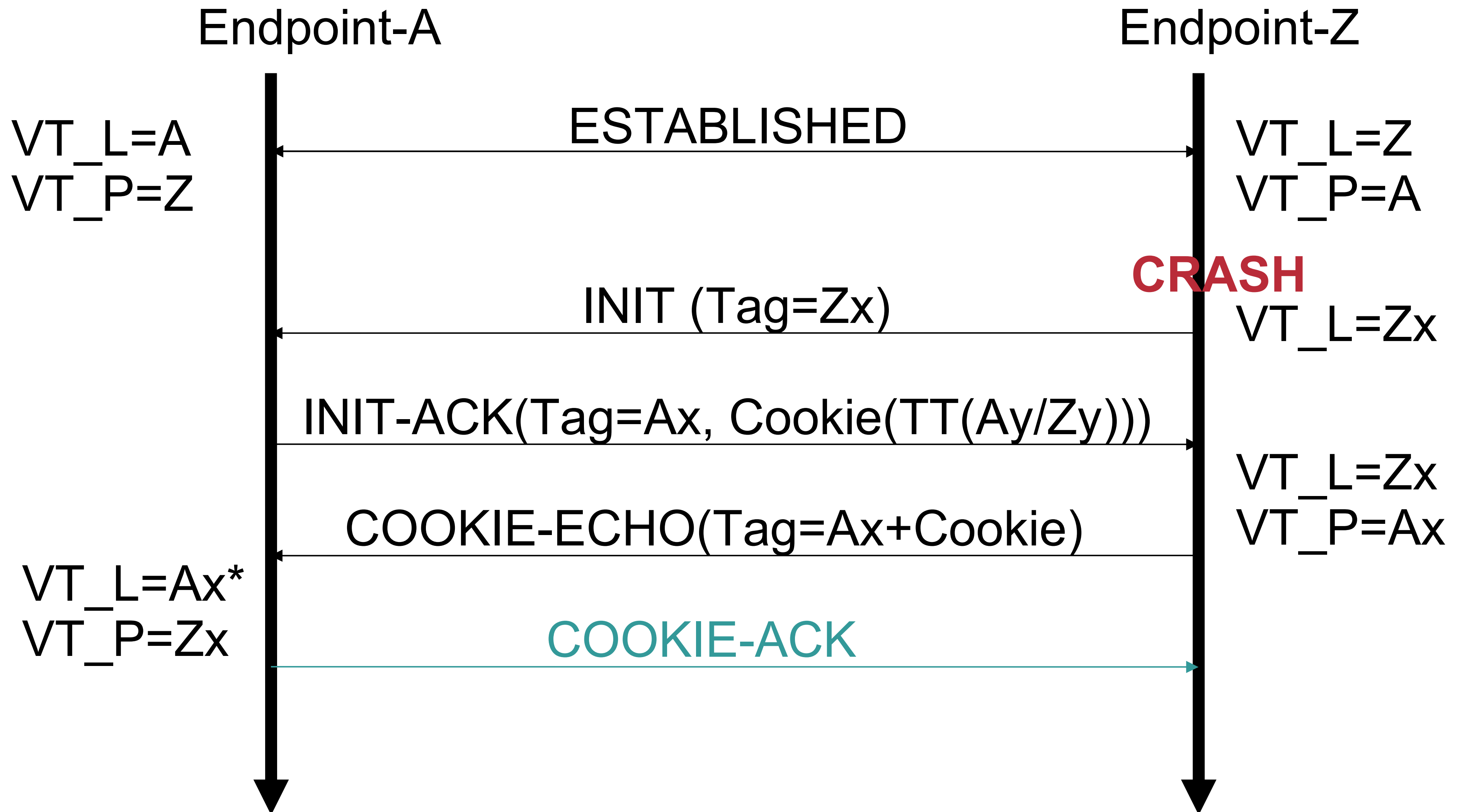
- RFC2960 and the SCTP reference book will instruct that the old V-Tags be used as the Tie-Tags.
- The most recent I-G has changed this so that V-Tags are never revealed on the wire except during their initial exchange. (Tie-Tags now are basically 32 bit random nonces that represent the TCB).

  This new change in the I-G adds extra security for a minimal additional TCB storage cost.

- The restarting peer considers everything normal when the INIT-ACK arrives and sends off the COOKIE-ECHO which holds the Tie-Tags.

© 2003 Cisco Systems, Inc. All rights reserved.

Peer Restart



* App is given Restart notification

© 2003 Cisco Systems, Inc. All rights reserved.

Restart: Final Processing

- Endpoint-A will unpack and verify the state cookie. As part of validation it will use the Tie-Tags to determine that a peer restart as occurred.
- It will reply with a COOKIE-ACK to the restarted peer (Endpoint-Z).
- It will also notify its upper layer or application that a peer restart has occured.
- Note that the SCTP stack on Endpoint-Z is never aware that a restart of the association has occurred.

© 2003 Cisco Systems, Inc. All rights reserved.

Questions

- **Questions**

© 2003 Cisco Systems, Inc. All rights reserved.

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.

© 2003 Cisco Systems, Inc. All rights reserved.

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 transfering 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.

© 2003 Cisco Systems, Inc. All rights reserved.

Receiving and Sending Messages

- A user will NEVER see two different messages in a buffer returned from a single rcvmsg() call
- An SCTP user will pass a message to the sndmsg() or sctp_sndmsg() function call for sending (more on these two calls later)
- The user message will then take one of two paths through the SCTP stack:

    Fragmentation –or–

    Singleton

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Data Chunk Size

- In the case of a singleton, the message must fit entirely in one SCTP chunk.
- The maximum chunk size SCTP uses is usually dictated by the smallest MTU of all of the peer's destination addresses.
- Recall that PMTU discovery is part of RFC2960 and must be implemented.

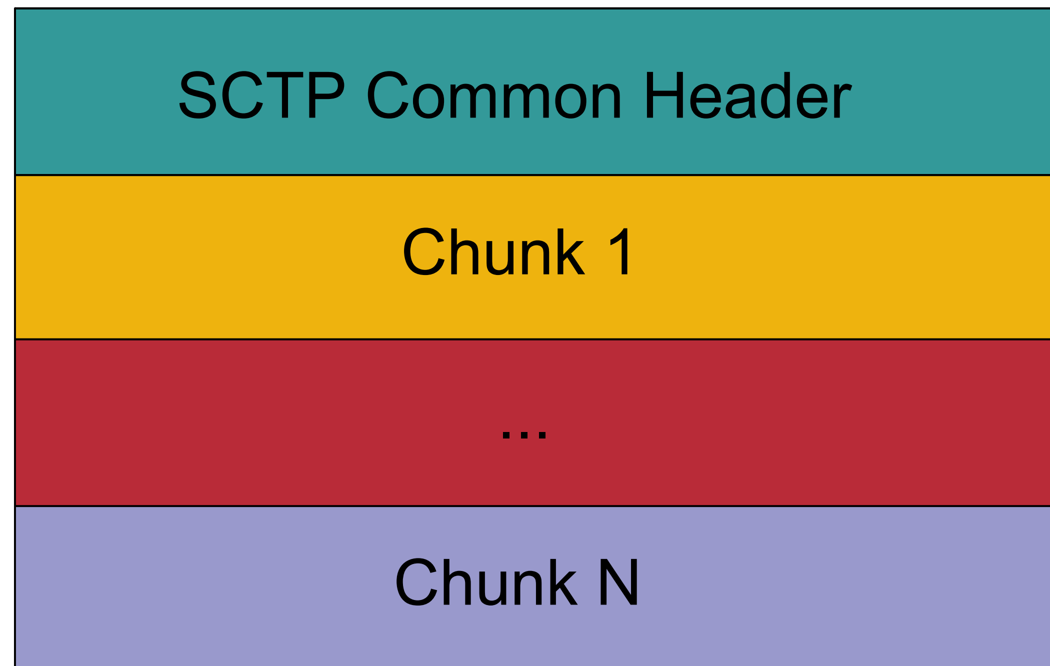
© 2003 Cisco Systems, Inc. All rights reserved.

Adding the Headers

- A DATA chunk header is prefixed to the user message.
- TSN, Stream Identifier, and Stream Sequence Number (if ordered) are assigned to each DATA chunk.
- The DATA chunk is then queued for bundling into an SCTP packet.
- An SCTP packet is a common header plus a collection of chunks (both control and DATA)

© 2003 Cisco Systems, Inc. All rights reserved.

An SCTP Packet



© 2003 Cisco Systems, Inc. All rights reserved.

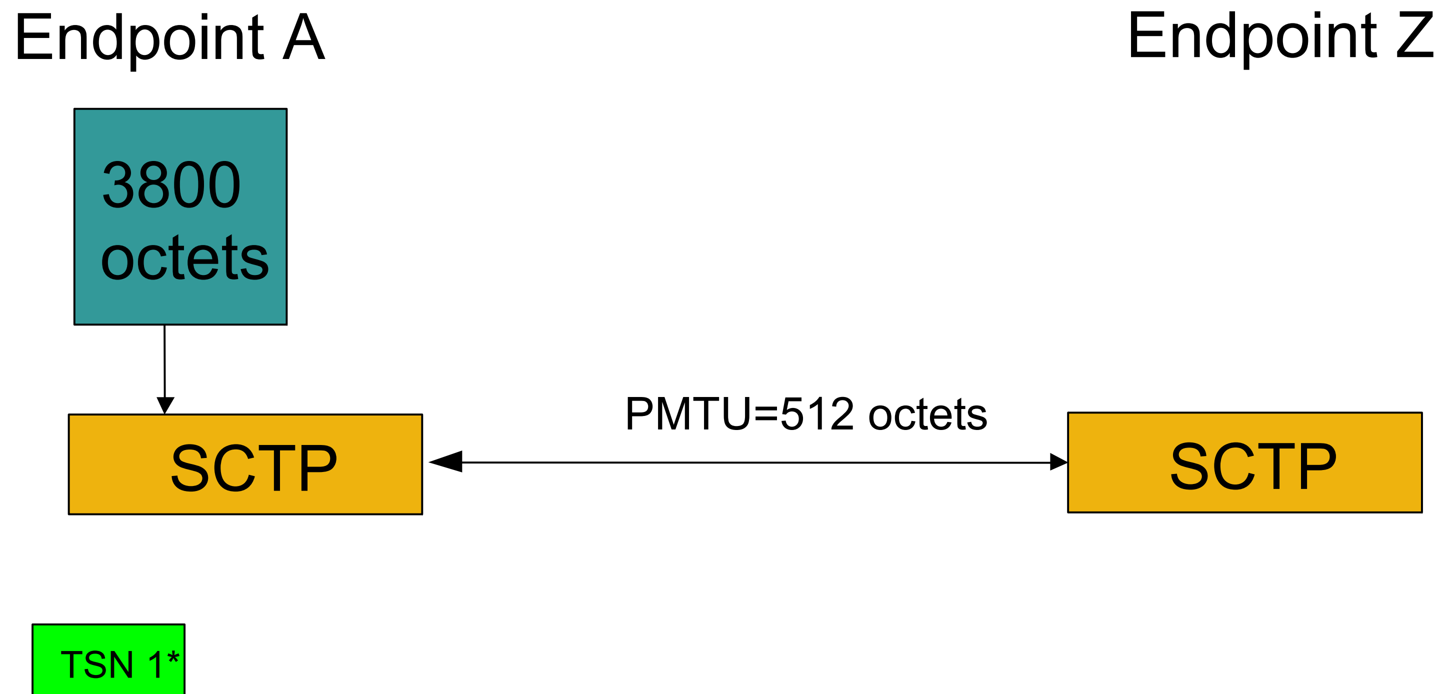
What To Do When It Won't All Fit?

- **The process of splitting messages up into multiple parts is called fragmentation.**
- **A message that cannot fit into a single chunk is chopped up into multiple parts.**
- **All parts of the same message use the same Stream Identifier (SID) and Stream Sequence Number (SSN).**
- **Each part will use a unique TSN (in consecutive order) and appropriate flag bits to indicate if it is a first, last, or middle piece of a message.**

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Cisco.com



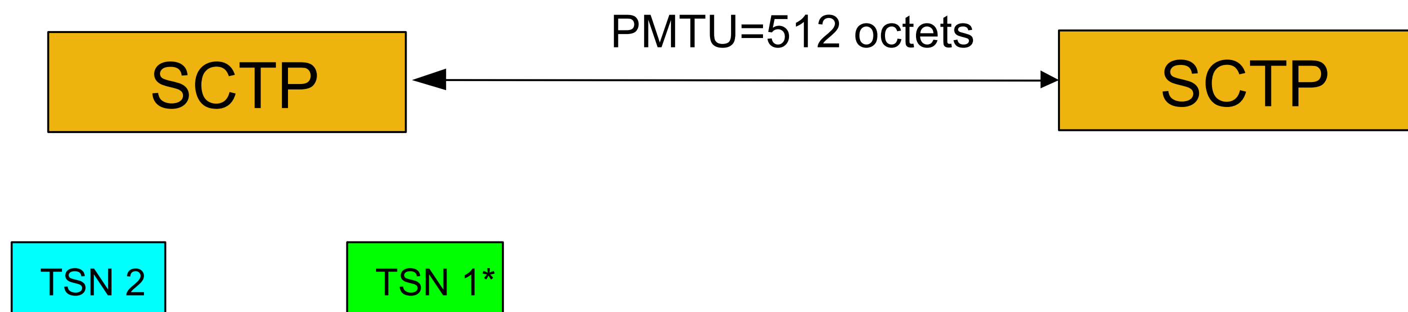
* - B bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



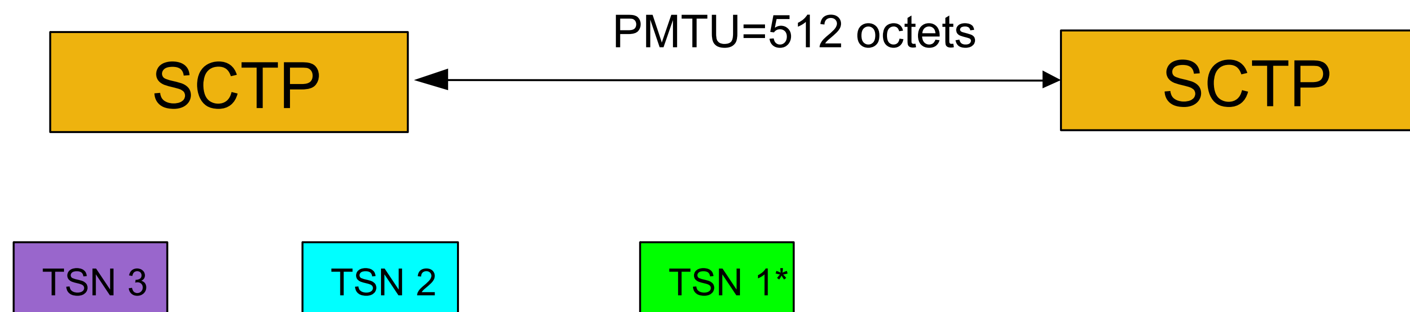
* - B bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



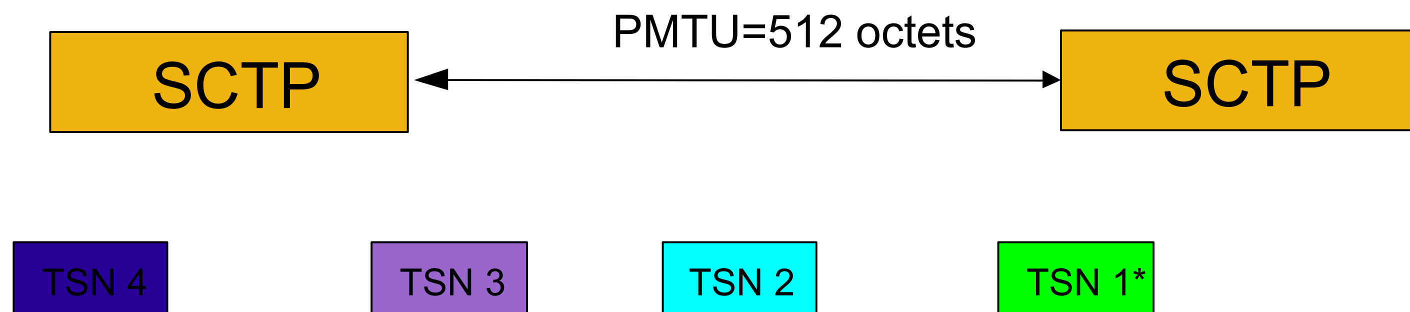
* - B bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



* - B bit set to 1

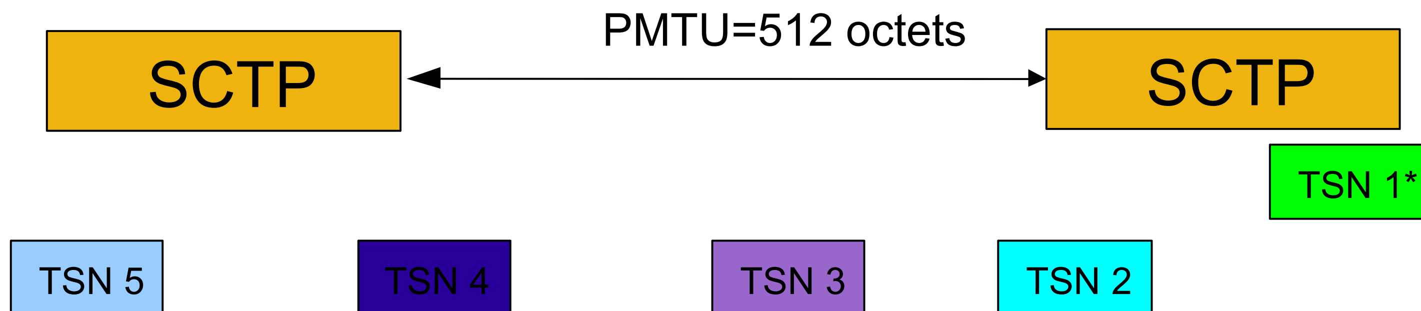
© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Cisco.com

Endpoint A

Endpoint Z



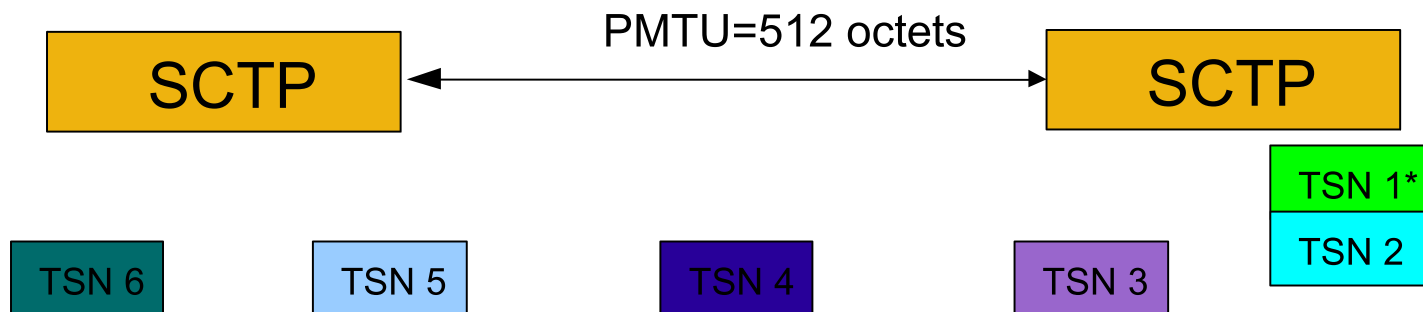
* - B bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



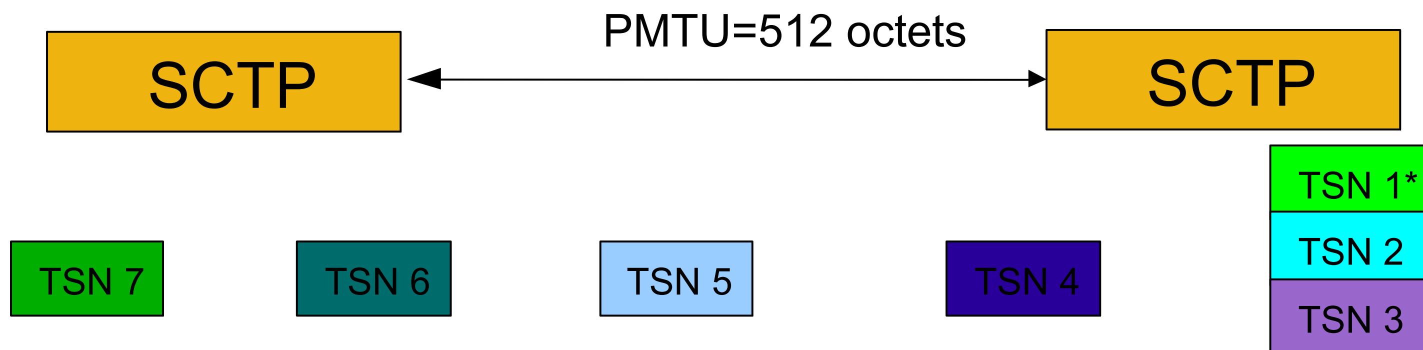
* - B bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



* - B bit set to 1

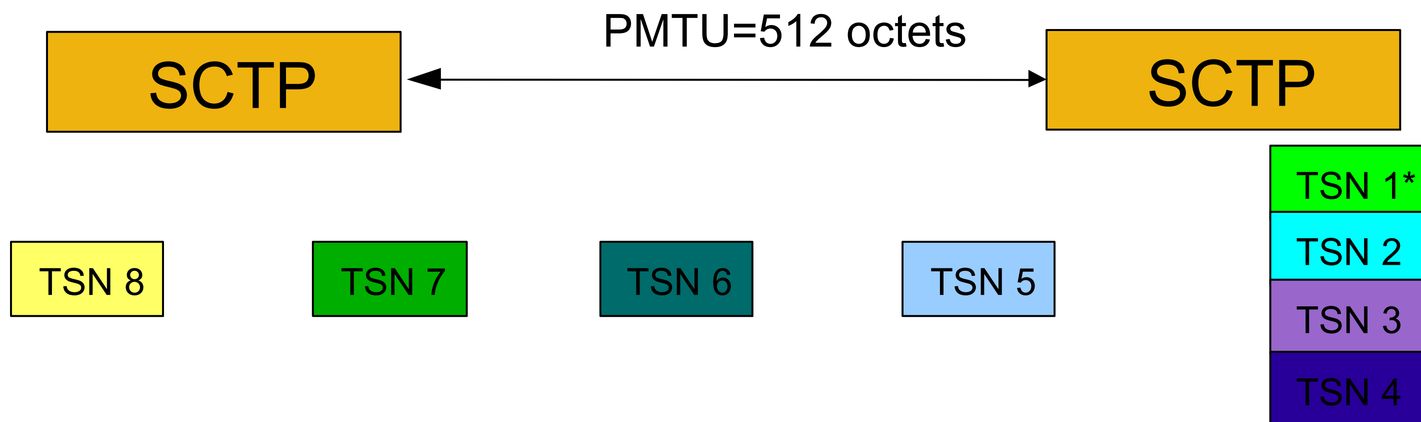
© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Cisco.com

Endpoint A

Endpoint Z



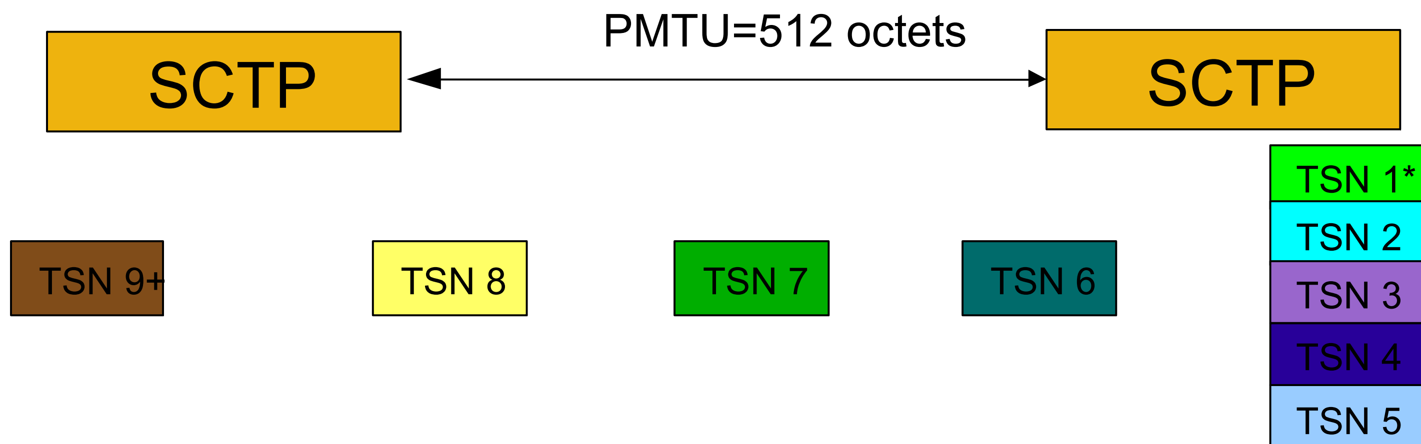
* - B bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



* - B bit set to 1

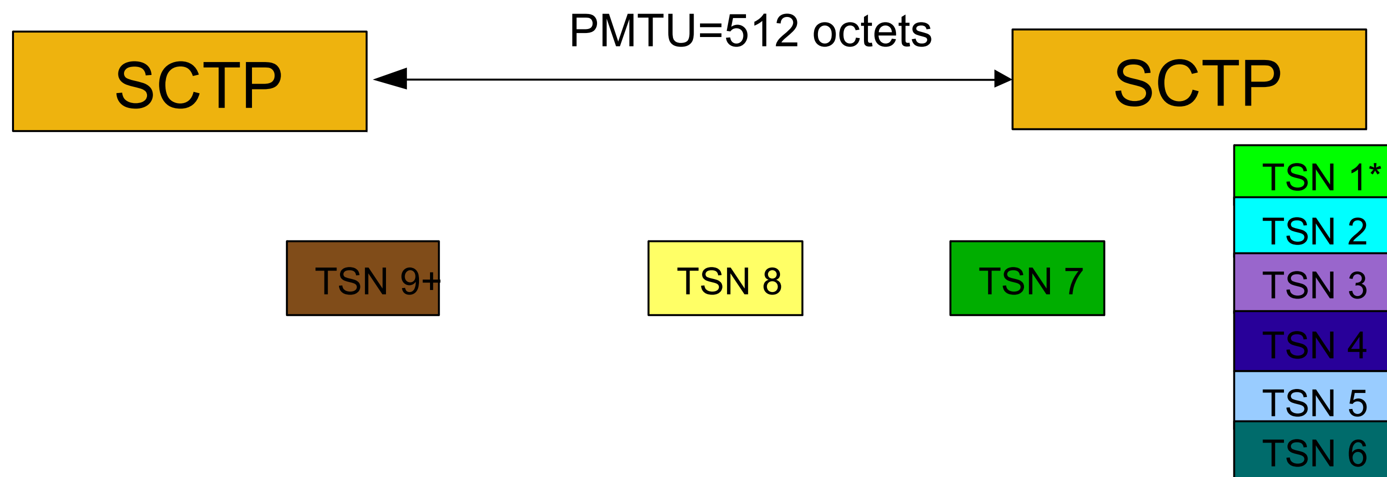
\+ - E bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z



* - B bit set to 1

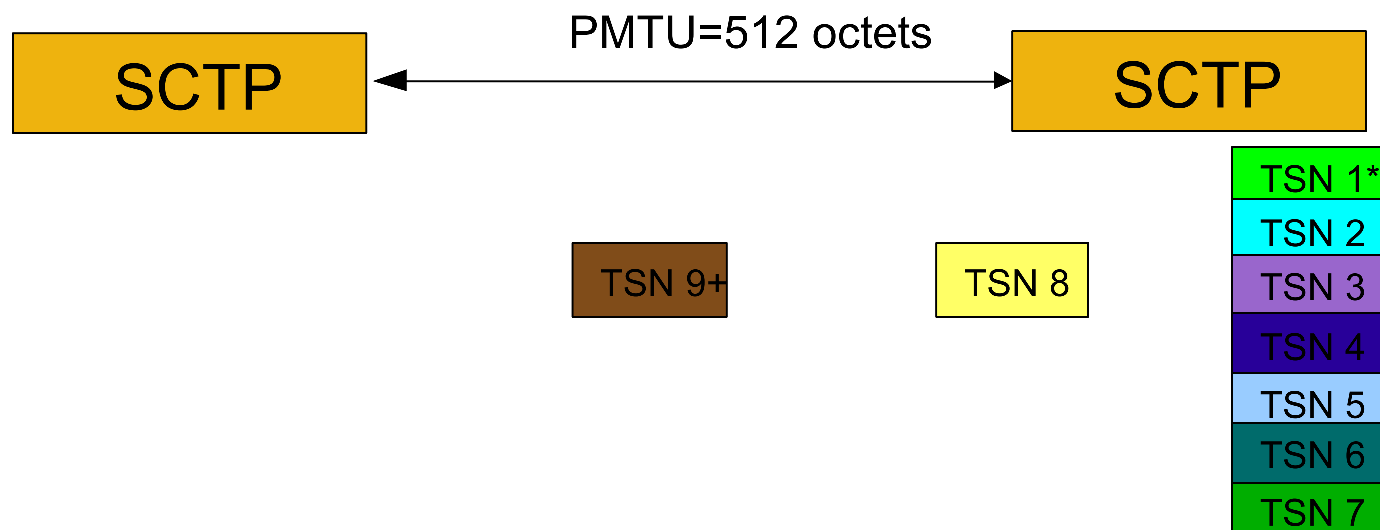
+ - E bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

Endpoint A

Endpoint Z

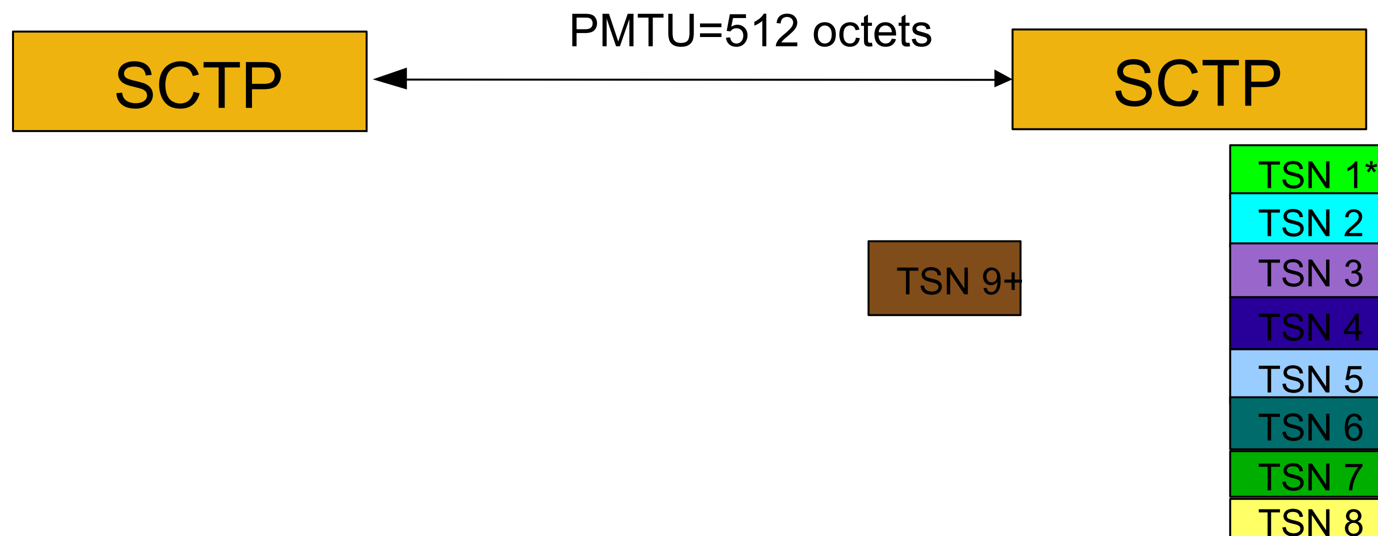


* - B bit set to 1

+ - E bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer

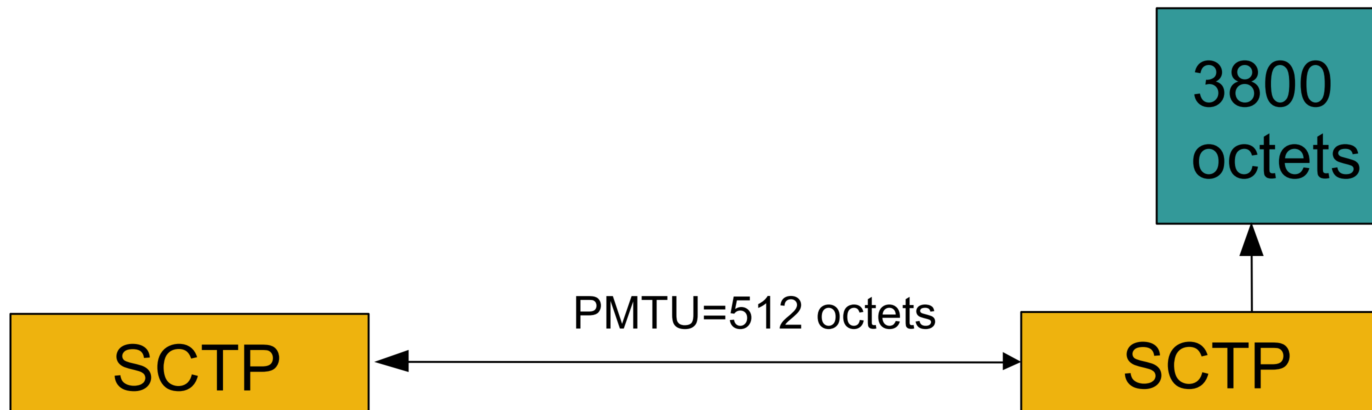


* - B bit set to 1

\+ - E bit set to 1

© 2003 Cisco Systems, Inc. All rights reserved.

A Large Message Transfer



© 2003 Cisco Systems, Inc. All rights reserved.

Data Reception

- When a SCTP packet arrives all control chunks are processed first.
- Data chunks have their chunk headers detached and the user message is made available to the application.
- Out-of-order messages within a stream will be held for stream sequence re-ordering.
- If a fragmented message is received it is held until all pieces of it are received.

© 2003 Cisco Systems, Inc. All rights reserved.

More on Data Reception

- All pieces are received when the receiver has a chunk with the first (B) bit set, the last (E) bit set, and all intervening TSN's between these two chunks.
- The data is reassembled into a user message using the TSN to order the middle pieces from lowest to highest.
- After reassembly, the message is made available to the upper layer (within ordering constraints).

© 2003 Cisco Systems, Inc. All rights reserved.

Transmission Rules

- As in TCP, a congestion window (cwnd) and receive window (rwnd) are used to control sending of user data.
- A sender must not transmit more than the calculated cwnd on a destination address.
- The sender also must not attempt to send more than the peer's rwnd to the peer.

© 2003 Cisco Systems, Inc. All rights reserved.

More on Transmission

- **However, if the peer closes its rwnd to 0 and the sender has no data chunks in flight, it may always send one packet with data to probe for a change in the rwnd.**

© 2003 Cisco Systems, Inc. All rights reserved.

Selective Acknowledgment

Cisco.com

- Data is acknowledged via a delayed SACK scheme similar to TCP.
- A SACK chunk includes the cumulative ack point (cum-ack) point.
- cum-ack is the highest sequential TSN that has been received.
- Out-of-order segments received are reported with “gap ack blocks” in the SACK

© 2003 Cisco Systems, Inc. All rights reserved.

More on SACK

- We always attempt to send a SACK back towards the destination address where the DATA came from.
- With the cum-ack point and gap ack blocks, a SACK chunk fully describes all TSN's received within PMTU constraints:

  For a 1500 byte ethernet frame, this means that over 360 gap blocks can be included in addition to the fixed fields of a SACK chunk.

- A SACK may also contain indications of duplicate data reception.

© 2003 Cisco Systems, Inc. All rights reserved.

More on SACKs

- A receiver is allowed to revoke any data previously acknowledged in gap ack blocks.

  Example: receiver's reassembly buffer is memory limited

- This means that a sender must hold a TSN until after the cum-ack has reached it.

© 2003 Cisco Systems, Inc. All rights reserved.

Retransmission Timer

- **SCTP maintains a Round Trip Time (RTT) and a Retransmission Time Out (RTO).**
- **Most SCTP implementations will use an integer approximation for the RTT formula created by Van Jacobson for TCP i.e. SCTP and TCP use a similar formula but in practice everyone uses the same exact math for both TCP and SCTP.**

© 2003 Cisco Systems, Inc. All rights reserved.

More on Retransmission

- While sending data, a endpoint try's to measure the RTT once every round trip.
- We do NOT measure the RTT of any packet that is retransmitted (since upon acknowledgment we don't know which transmission the SACK goes with).
- Since SCTP is a multi-homed protocol, there is a small complication of how the T3-rtx timer is managed.

© 2003 Cisco Systems, Inc. All rights reserved.

Even More on Retransmission Timer

Cisco.com

- A general rule of thumb is that for any destination that has outstanding data (unacknowledged data) a retransmission timer should be running.
- When all data that was in-flight to a destination is acknowledged, the timer should be stopped.
- A peer revoking acknowledgement may also cause a sender to restart a T3-rtx.
- When starting the T3 timer, we always use the RTO value not the RTT.

© 2003 Cisco Systems, Inc. All rights reserved.

Other Retransmissions

- **Like TCP, SCTP uses Fast Retransmit (FR) to expedite retransmission without always requiring a T3-rtx timeout.**
- **The SCTP sender keeps track of the “holes” that gap ack blocks report are missing by maintaining a strike count for those chunks.**
- **When the strike count reaches four, the DATA chunk is retransmitted.**

© 2003 Cisco Systems, Inc. All rights reserved.

More on Fast Retransmit

- When a FR occurs, a cwnd adjustment is made, but not as drastic as a T3-rtx timeout. [more on this later]
- Only one adjustment is made per flight of data so that multiple FR's in the same window do NOT cut the cwnd more than once (note the I-G has more details on this procedure).
- This single reduction is sometimes referred to as “NewReno”. NewReno is named after the version of TCP that it originated in.

© 2003 Cisco Systems, Inc. All rights reserved.

What Happens at Timer Expiration

- A cwnd adjustment is made [more on this later]
- The RTO is doubled.
- All outstanding data to that destination is marked for retransmission.
- If the receiver is multi-homed, an alternate address is selected for the data chunks that were in-flight.
- Retransmit up to one MTU's worth of data towards the peer.

© 2003 Cisco Systems, Inc. All rights reserved.

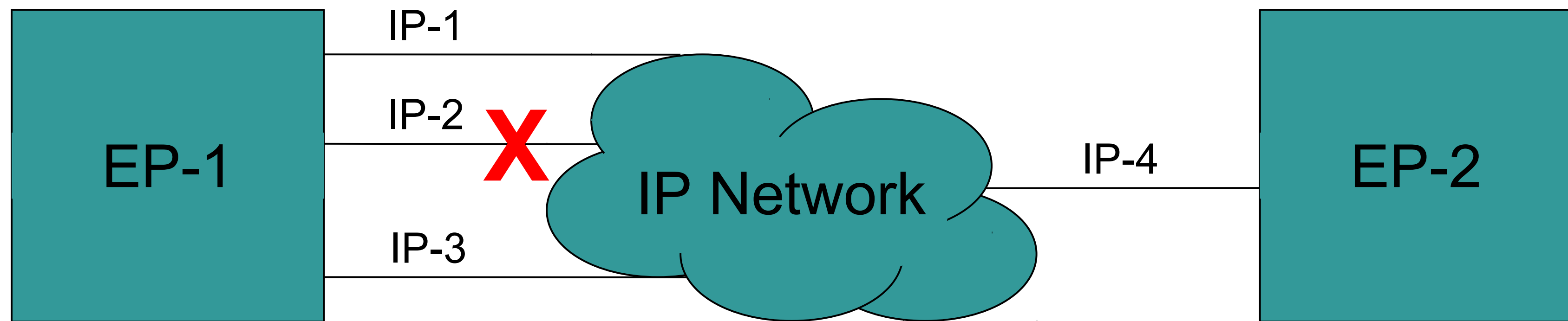
Multi-homed Considerations

Cisco.com

- When a peer is multi-homed, a “primary destination address” will be selected by the SCTP endpoint.
- By default, all data will be sent to this primary address.
- When the primary address fails, the sender will select an alternate primary address until it is restored or the user changes the primary address.
- SACK's may also require some special handling, consider the following:

© 2003 Cisco Systems, Inc. All rights reserved.

A Multi-homed Peer With a Failure



© 2003 Cisco Systems, Inc. All rights reserved.

Special Considerations

- **If IP-2 was EP-2's primary address, then the association may still fail even though EP-1 has multiple addresses.** [more on association failures later]
- **In the preceding drawing imagine that EP-1 is sending packets with source address IP-2.**
- **If EP-2 always sends SACK's back to IP-2, EP-1 will never receive a SACK.**
- **To prevent this, a receiver will generally alter the destination address of a SACK if it receives duplicate data.**

© 2003 Cisco Systems, Inc. All rights reserved.

Streams and Ordering

- **A sender tells the sndmsg() or sctp_sndmsg() 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.**

© 2003 Cisco Systems, Inc. All rights reserved.

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.

© 2003 Cisco Systems, Inc. All rights reserved.

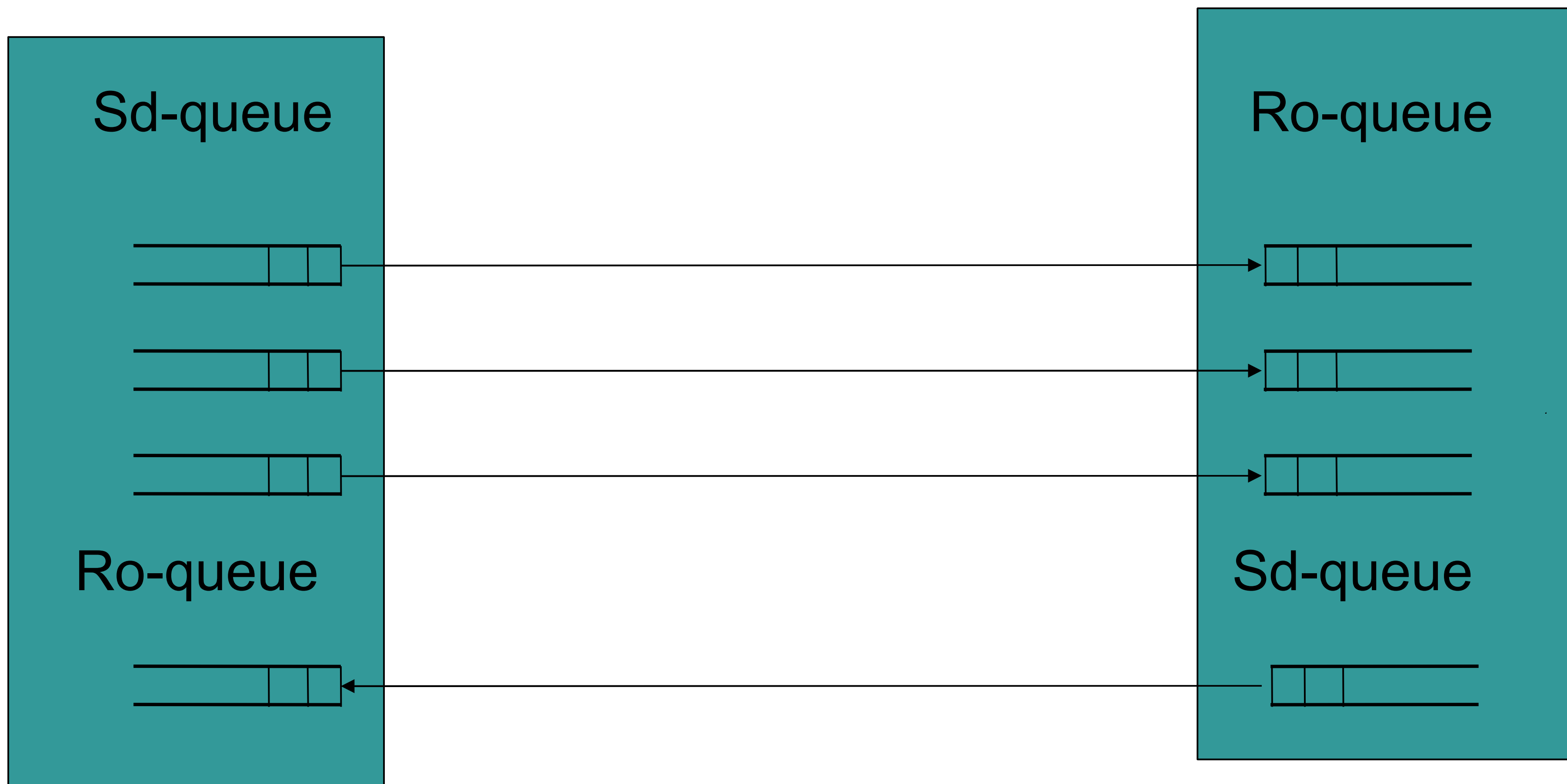
Stream Queues

Cisco.com

- 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.

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Streams



© 2003 Cisco Systems, Inc. All rights reserved.

Payload Protocol Identifier

- Each DATA chunk also includes a Payload Protocol Identifier (PPID).
- This PPID is used by the application and network monitoring equipment to understand the type of data being transmitted.
- SCTP pays no attention to this field (it's opaque).

© 2003 Cisco Systems, Inc. All rights reserved.

Partial Delivery

- Normally, a user gets an entire message when it reads from its socket. The Partial Delivery API provides an exception to this.
- The PD-API is invoked when a message is large in size and the SCTP stack needs to begin delivery of the message to help free some of the resources held by it during re-assembly.
- The pieces are always delivered in order.
- The API provides a “you have more” indication.

© 2003 Cisco Systems, Inc. All rights reserved.

Partial Delivery II

- The application must continue to read until this indication clears and assemble the large message.
- At no time, once the PD-API is invoked, will the application receive any other message (even if fully received by SCTP) until the entire PD-API message has been read.
- Normally the PD-API is not invoked unless the message is very large (usually ½ or more of the receive buffer).

© 2003 Cisco Systems, Inc. All rights reserved.

Error Protection Revisited

- SCTP was originally defined with the Adler-32 checksum.
- This checksum was easy to calculate but was shown to be weak and in-effective for small messages.
- After MUCH debate the checksum was changed to CRC32c (the same one used by iSCSI) in RFC3309.
- This provides MUCH stronger data integrity than UDP or TCP but does run an additional cost in computation.

© 2003 Cisco Systems, Inc. All rights reserved.

More Errors

- If a endpoint receives a packet with a bad checksum, the packet is silently discarded.
- Other types of errors may also occur, such as the sender using a stream number that was not negotiated up front (i.e. out of range):

  In this case, a ERROR report would be sent back to the peer, but the TSN would be acknowledged.
- If a empty DATA chunk is received (i.e. no user data) the association will be ABORTED.

© 2003 Cisco Systems, Inc. All rights reserved.

Questoions??

- **Questions**

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control (CC)

- **We will now go into congestion control (CC)**

  For some of you who have worked in transport, this will be somewhat repeatitive (sorry).

- **CC originally did not exist in TCP. This caused a series of congestion collapses in the late 80's.**

- **Congestion collapse is when the network is passing lots of data but almost ALL of that data is retransmissions of data that has already arrived at the peer.**

  RFC896 provides lots of details for those interested in congestion collapse

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control II

- In order to avoid congestion collapse, CC was added to TCP. An Additive Increase Multiplicative Decrease (AIMD) function is used to adjust sending rate.
- The basic idea is to slowly increase the amount an endpoint is allowed to send (cwnd), but collapse cwnd rapidly when there is sign of congestion.
- Packet loss is assumed to be the primary indicator and result of congestion.

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control Variables

- Like TCP, SCTP uses AIMD, but there are differences though in how it all works (compared to TCP).
- SCTP uses four control variables per destination address:

  cwnd – congestion window, or how much a sender is allowed to send towards a specific destination

  ssthresh – slow start threshold, or where we cut over from Slow Start to Congestion Avoidance (CA)

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control Variables II

Cisco.com

flightsize – or how much data is unacknowledged and thus "in-flight". Note that in RFC2960 the term flightsize is avoided, since it does not really have to be coded as a variable (an implementation may re-count flightsize as needed).

pba – partial bytes acknowledged. This is a new control variable that helps determine when a cwnd's worth of data has been sent and acknowledged while in CA

- We will go through the use of these variables in a example, so don't panic!

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control: Initialization

Cisco.com

- Initially a new destination address starts with a initial cwnd of two MTU's. However, the latest I-G changes this to min[4 MTU's, 4380 bytes].
- ssthresh is set theoretically infinity, but it is usually set to the peer's rwnd.
- flightsize and pba are set to zero.
- Slow Start (SS) is used when cwnd <= ssthresh. Note that initially we are in Slow Start (SS).

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control: Sending Data

- As long as there is room in the cwnd, the sender is allowed to send additional data into the network.

  There is room in the cwnd as long as flightsize < cwnd.

- This is slightly different then TCP in that SCTP can “slop” over the cwnd value. If the flightsize is (cwnd-1), another packet can be sent.
- Every time a SACK arrives, one of two algorithms, Slow Start (SS) or Congestion Avoidance (CA), is used to increment the cwnd.

© 2003 Cisco Systems, Inc. All rights reserved.

Controlling cwnd Growth

- When a SACK arrives in SS, we increment the cwnd by the either the number of bytes acknowledged or one MTU, whichever is less.

  Slow Start is used when cwnd <= ssthresh

- When a SACK arrives in CA, we increment pba by the number of bytes acknowledged. When pba > cwnd increment the cwnd by one MTU and reduce pba by the cwnd.

  Congestion Avoidance is used when cwnd > ssthresh

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control

- pba is reset to zero when all data is acknowleged
- We NEVER advance cwnd if the cumulative acknowledgment point is not moving forward.
- A Max Burst Limit is always applied to how many packets may be sent at any opportunity to send

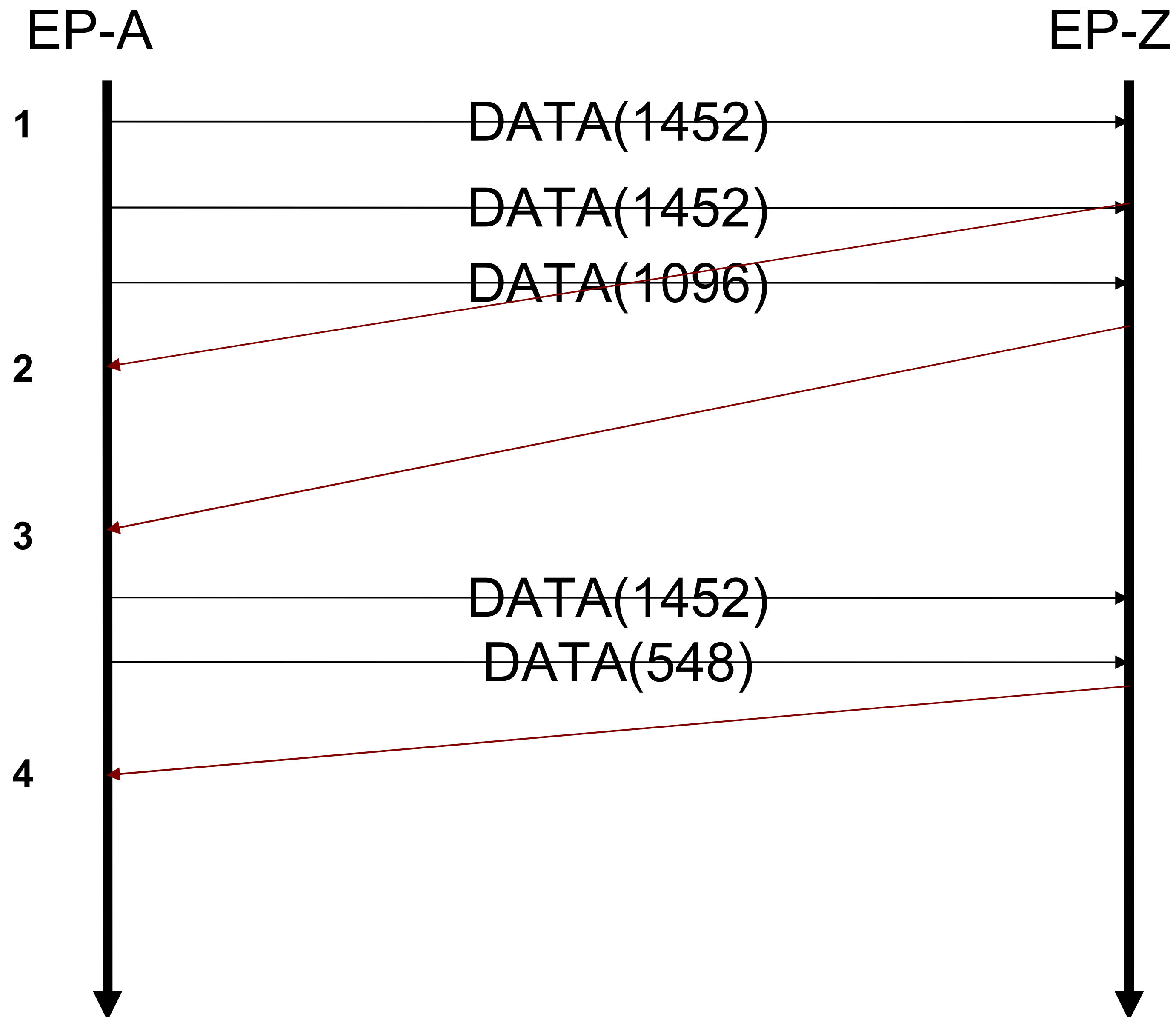
  This limit is usually 4

  An opportunity to send is any event that will cause data transmission (SACK arrival, user sending of data, etc.)

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control Example

Cisco.com



© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control Example II

- In our example, at point 1 we are at the initial stage, cwnd=3000, ssthresh = infinity, pba=0, flightsize=0. Our application sends 4000 bytes.
- The implementation sends these (note there is no block by cwnd).
- At point 2, the SACK arrives and we are in SS. The cwnd is incremented to 4500 bytes, i.e: add min(1500, 2904).

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control Example III

- At point 3, the SACK arrives for the last data segment, but no cwnd advance is made, why?
- Our application now sends 2000 bytes. These can be sent since flightsize is 0, cwnd is 4500.
- At point 4, no congestion control advancement is made.
- So we end with flightsize=0, pba=0, cwnd=4500, and ssthresh still infinity.

© 2003 Cisco Systems, Inc. All rights reserved.

Reducing cwnd and Adjusting ssthresh

- The cwnd is lowered on two events, all regarding a retransmission event.
- Upon a T3-rtx timeout, set ssthresh to ½ the value of cwnd or 2 MTU whichever is more. Then set cwnd to 1 MTU.
- Upon a Fast Retransmit (FR), set ssthresh again to ½ the cwnd or 2 MTU whichever is more. Then set cwnd to the value of ssthresh.

© 2003 Cisco Systems, Inc. All rights reserved.

Congestion Control

- Note this means that if we were in CA, we move back to SS for either FR or T3-rtx adjustments to cwnd.
- So how do we tell if we are in CA or SS?

  Any time the cwnd is larger than the ssthresh we perform the CA algorithm. Otherwise we are in SS.

© 2003 Cisco Systems, Inc. All rights reserved.

Path MTU Discovery

Cisco.com

- PMTU Discovery is “built” into the SCTP protocol.
- A SCTP sender always sets the DF bit in IPv4.
- When a packet with DF bit set will not “fit”, then an ICMP message is returned by the trusty router.
- This message is used to reset the PMTU and possibly the smallest MTU.
- Note that this may also mean re-chunking may occur as well (in some situations).

© 2003 Cisco Systems, Inc. All rights reserved.

Questions

- **Questions?**

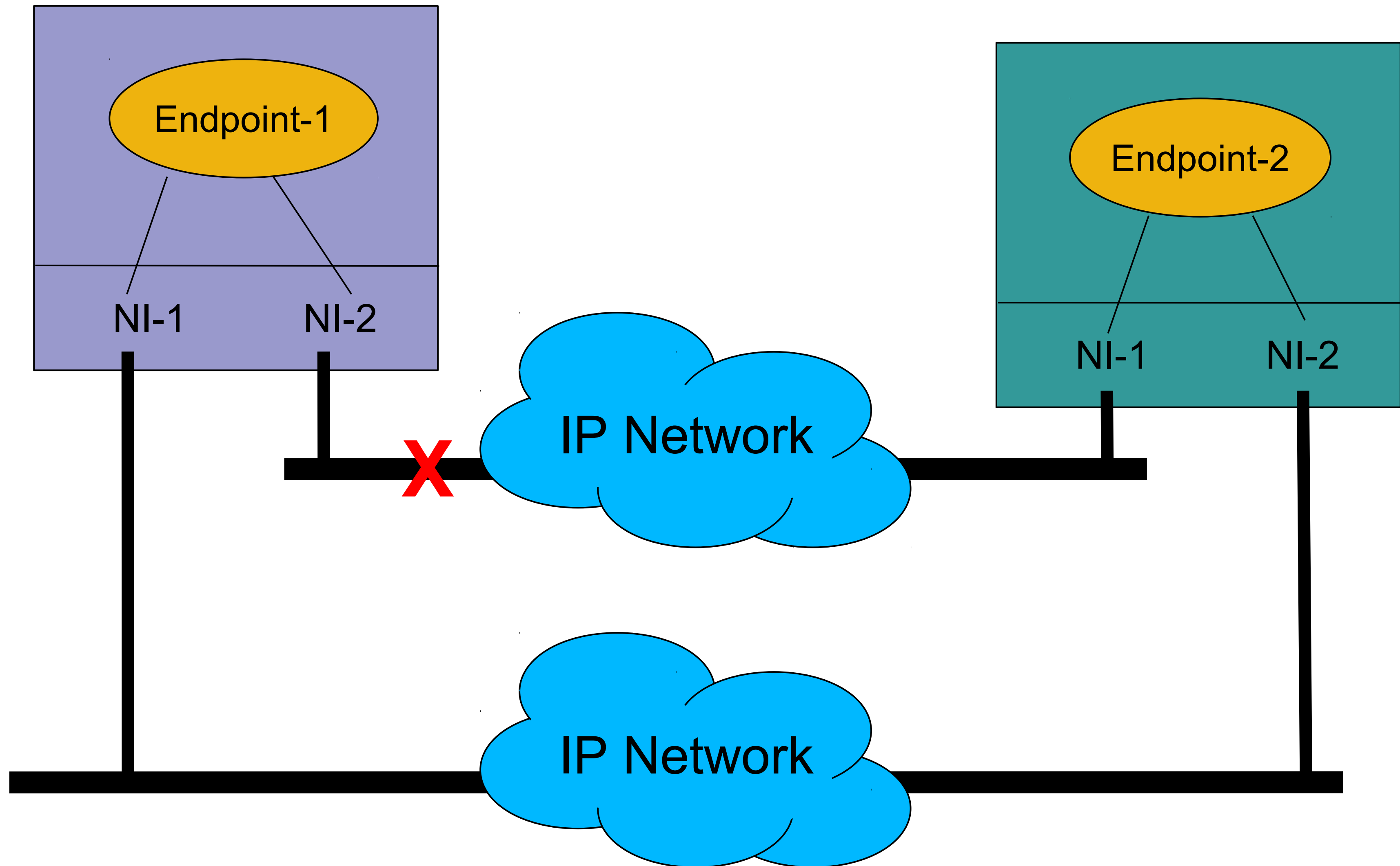
© 2003 Cisco Systems, Inc. All rights reserved.

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 either a hardware or network failure**

© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Destination Address



© 2003 Cisco Systems, Inc. All rights reserved.

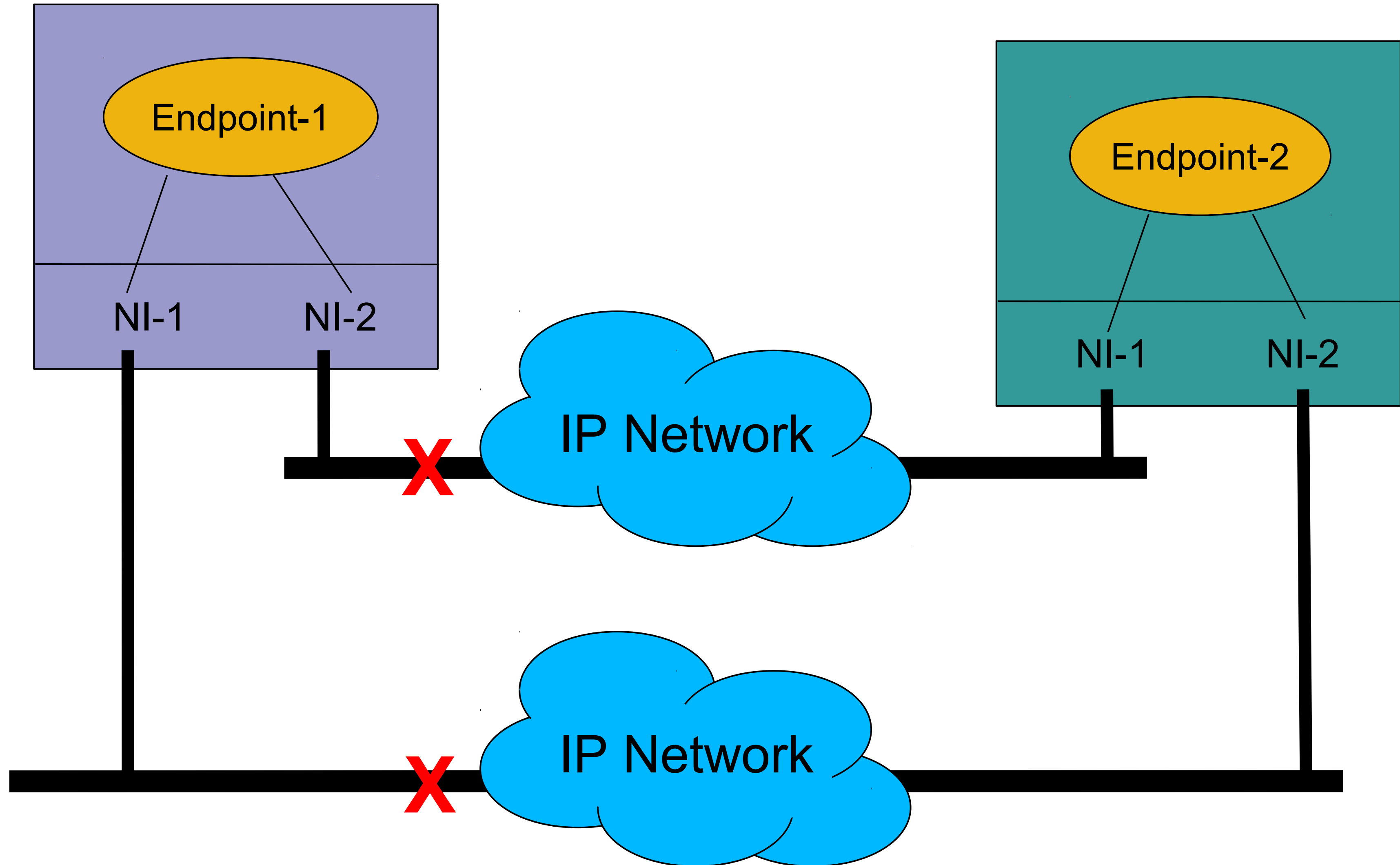
Unreachable Peer Failure

Cisco.com

- A peer may be unreachable due to either:
  - A complete network failure
  - Or, more likely, a peer software or machine failure
- To an SCTP endpoint, both cases appear to be the same failure event (network failure or machine failure).
- In cases of a software failure if the peers SCTP stack is still alive the association will be shutdown either gracefully or with an ABORT message.

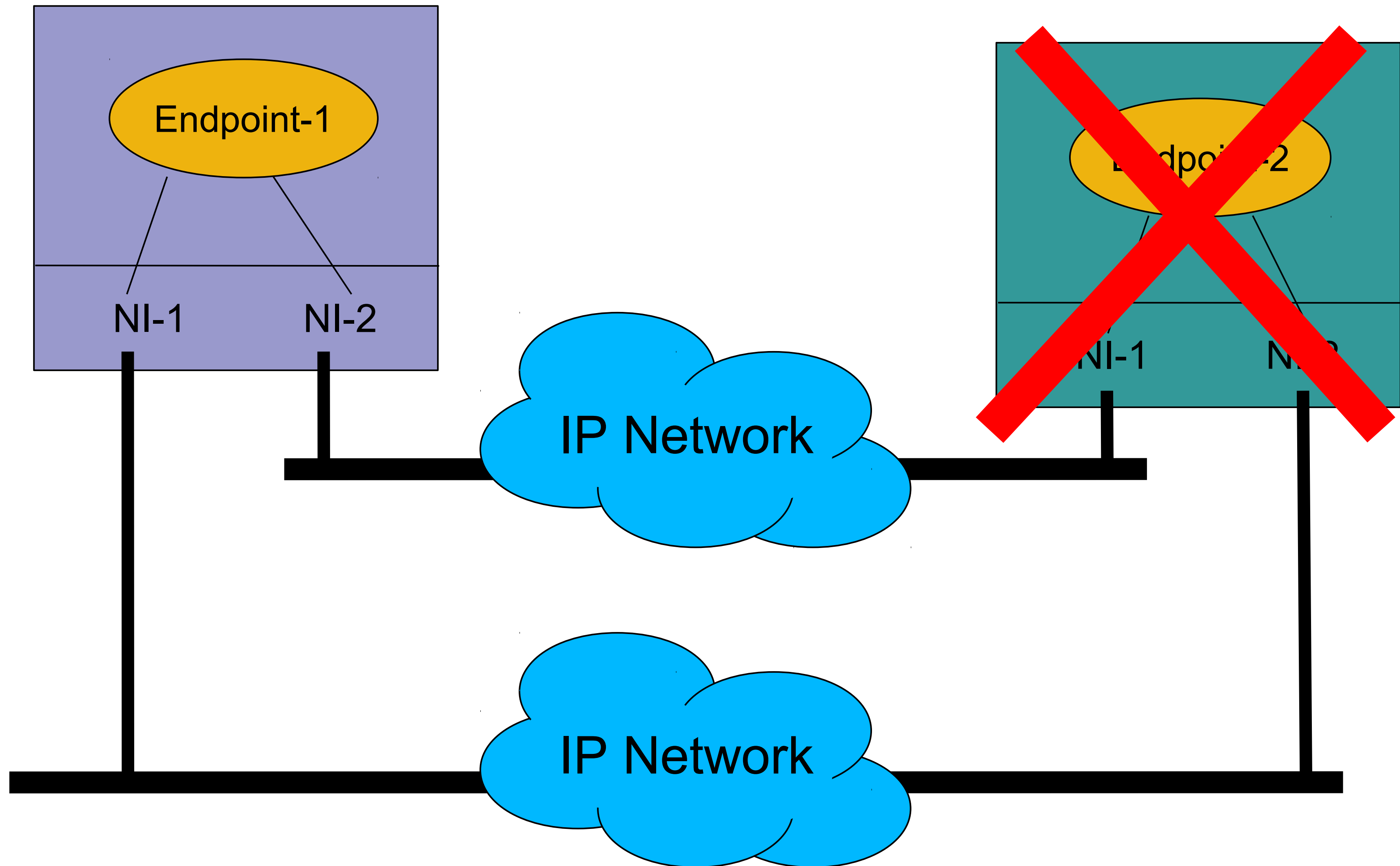
© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Peer: Network Failure



© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Peer: Endpoint Failure



© 2003 Cisco Systems, Inc. All rights reserved.

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

© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Destination Detection

- Each time a HEARTBEAT is sent, a Destination Error count for that destination is incremented.
- Any time a HEARTBEAT-ACK is received, the Error count is cleared.
- Any time DATA is acknowledged that was sent to a destination, its Error count is cleared.
- Any time a DATA T3-rtx timeout occurs on a destination, the Error count is incremented.
- Any time the Destination Error count exceeds a threshold (usually 5), the destination is declared unreachable.

© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Destination II

- If a primary destination is marked "unreachable", an alternate is chosen (if available).
- Heartbeats will continue to be sent to "unreachable" addresses.
- If a Heartbeat is ever answered, the Error count is cleared and the destination is marked "reachable".

  If it was the primary destination and no user intervention has occurred, it is restored as the primary destination.

© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Peer I

- In addition to the Destination Error count, an overall Association Error count is also maintained.
- Each time a Destination Error count is incremented, so is the Association Error count.
- Each time a Destination Error count is cleared, so is the Association Error count.
- If the Association Error count exceeds a threshold (usually 8), the peer is marked as unreachable and the association is torn down.

© 2003 Cisco Systems, Inc. All rights reserved.

Unreachable Peer II

- Note that the two control variables are seperate and unrelated (i.e. Destination Error threshold and the Association Error threshold).
- It is possible that ALL destinations are unreachable and yet the Association Error count has not exceeded its threshold for association tear down.
- This is what is known as being in the Dormant State.
- In this state, MOST implementations will at least continue to send to one address.

© 2003 Cisco Systems, Inc. All rights reserved.

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

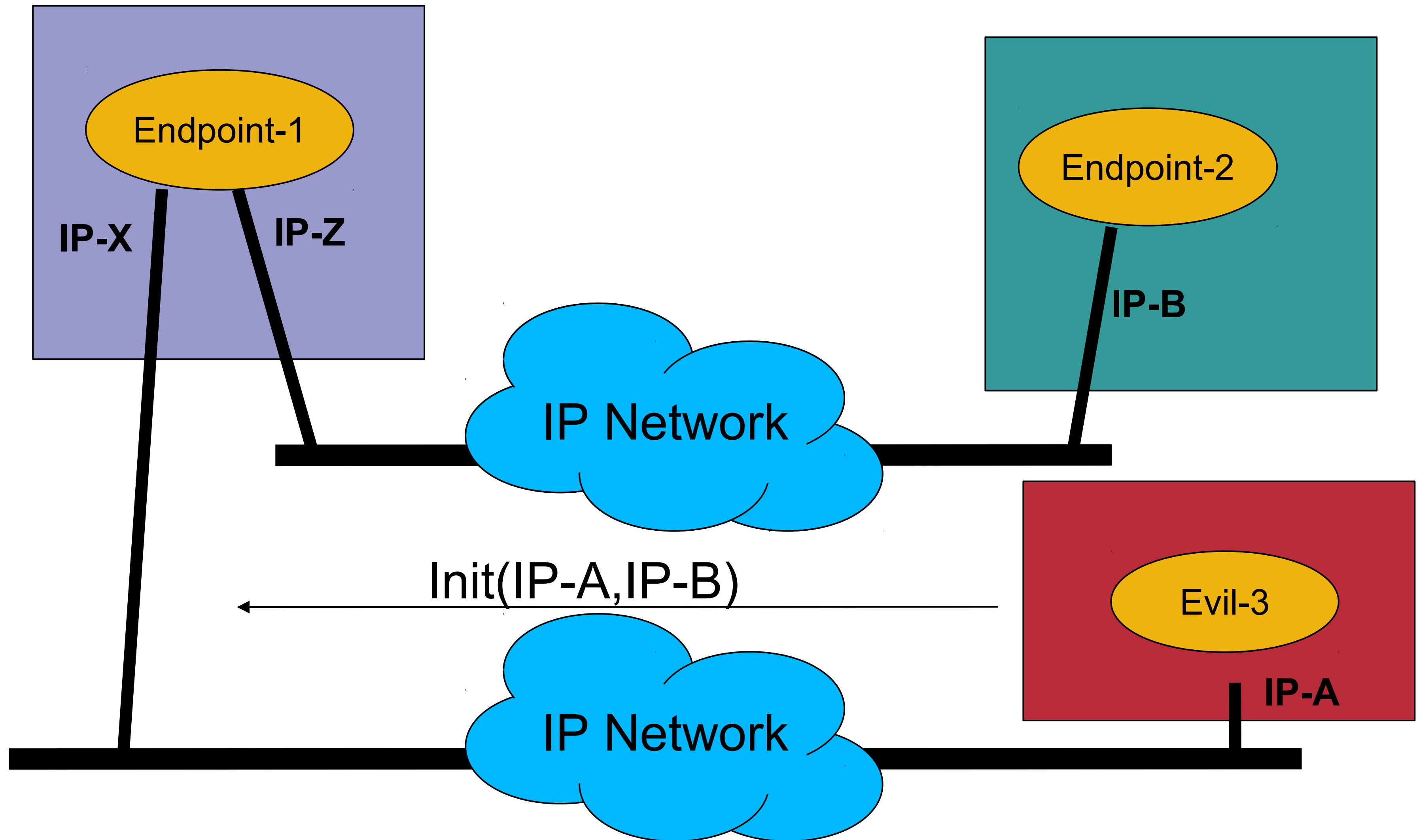
© 2003 Cisco Systems, Inc. All rights reserved.

Address Confirmation

- **All addresses added to an association via INIT or INIT-ACK's address lists that were NOT supplied by the user or used to exchange the INIT and INIT-ACK are considered to be suspect.**
- **These address are marked unconfirmed and CANNOT be marked as the primary address.**
- **A Heartbeat with a 64-bit nonce must be sent and an Heartbeat-Ack with the proper nonce returned before an address can leave the unconfirmed state.**

© 2003 Cisco Systems, Inc. All rights reserved.

Why Address Confirmation



© 2003 Cisco Systems, Inc. All rights reserved.

Heartbeat Controls

Cisco.com

- 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.

© 2003 Cisco Systems, Inc. All rights reserved.

Heartbeat Controls II

- All heartbeats have a random delta (jitter) added to them to prevent synchronization.
- The heartbeat interval will equate to

  RTO + HB.Interval + (delta).
- The random delta is +/- 0.50 of RTO.
- Unanswered heartbeats cause RTO doubling.

© 2003 Cisco Systems, Inc. All rights reserved.

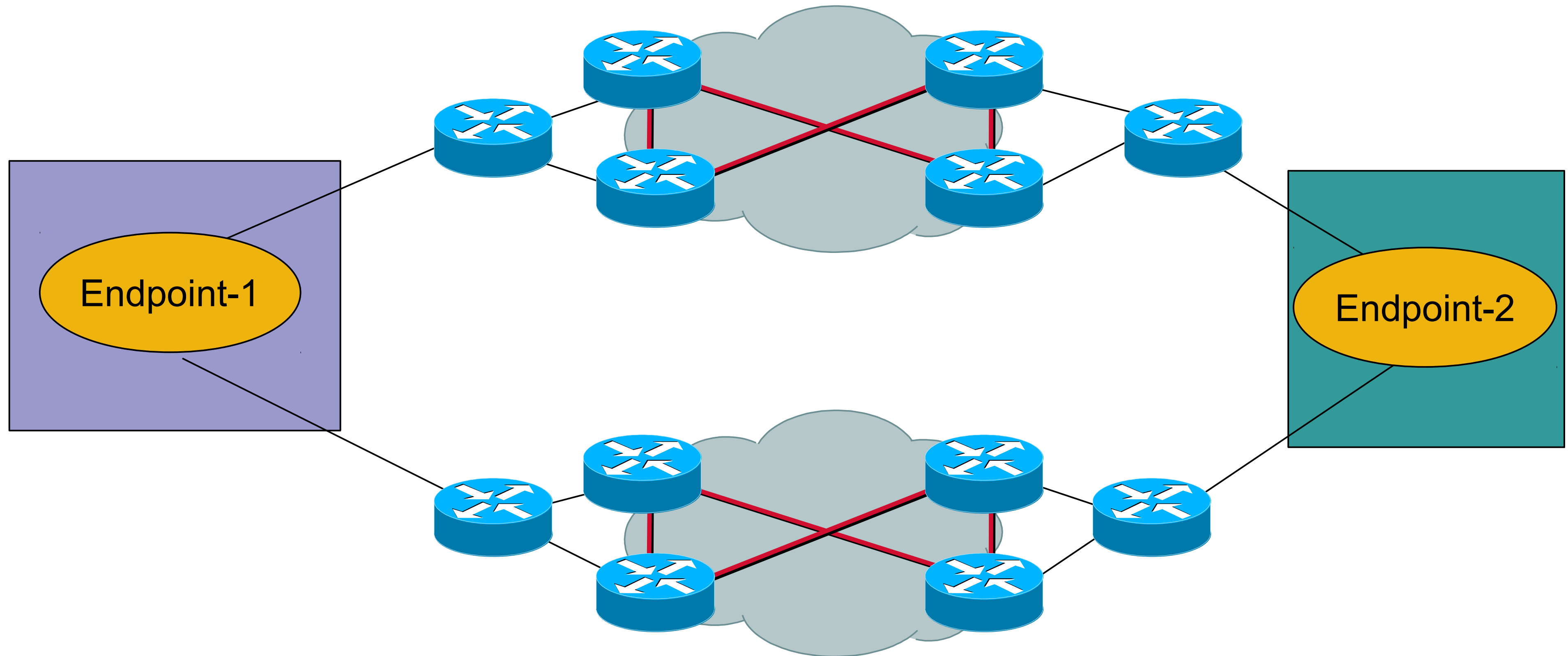
Network Diversity and Multi-homing

- 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?

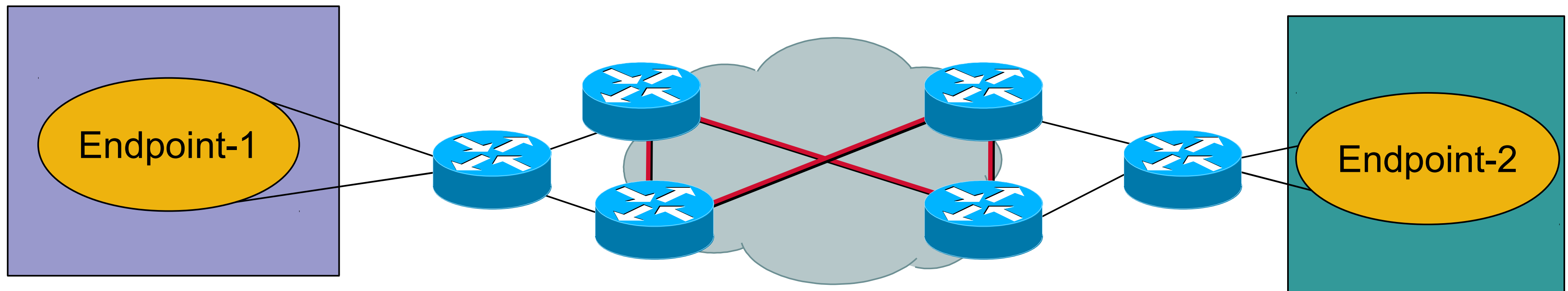
© 2003 Cisco Systems, Inc. All rights reserved.

Maximum Path Diversity



© 2003 Cisco Systems, Inc. All rights reserved.

Minimum Path Diversity



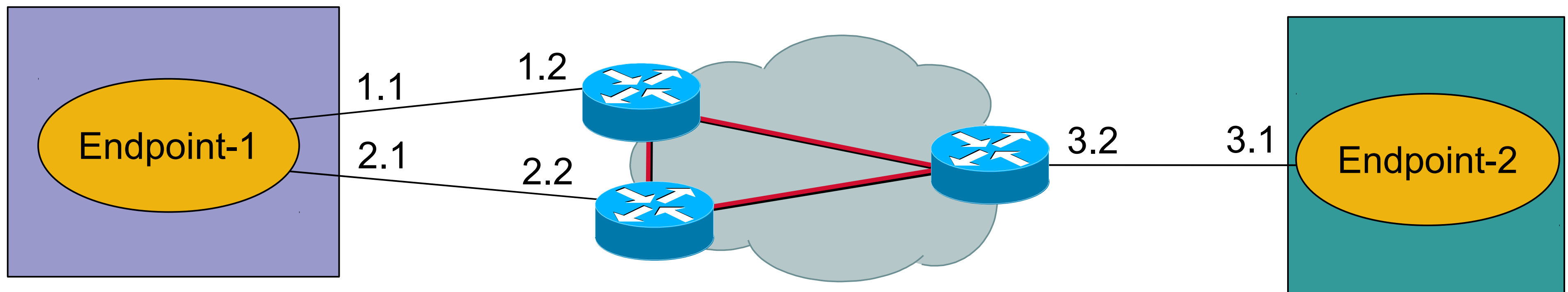
© 2003 Cisco Systems, Inc. All rights reserved.

Asymmetric Multi-homing

- In some cases, one side will be multi-homed while the other side is singly-homed.
- In this configuration, a single failure on the multi-homed side may still disable the association.
- This failure may occur even when an alternate route exists.
- Consider the following picture:

© 2003 Cisco Systems, Inc. All rights reserved.

Aysmmetric Multi-Homing



E-1 Route Table

3.0 -> 1.2

E-2 Route Table

1.0 -> 3.2

2.0 -> 3.2

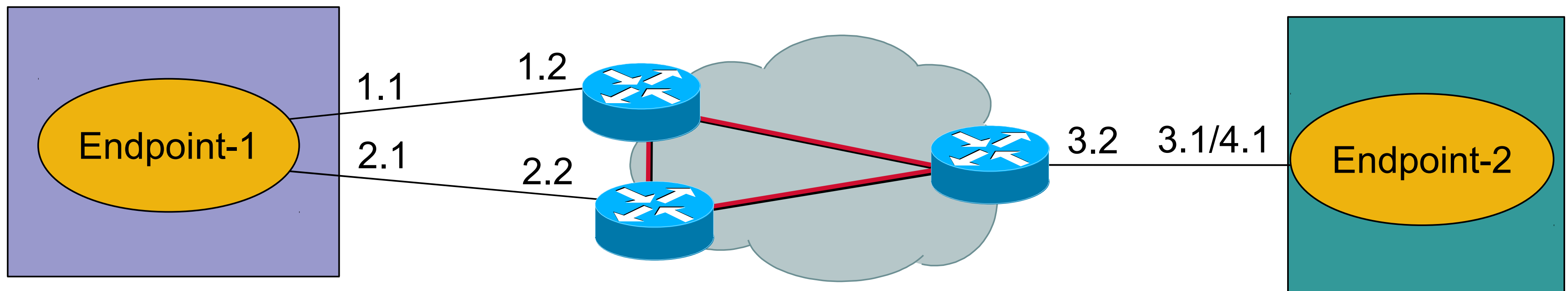
© 2003 Cisco Systems, Inc. All rights reserved.

Solutions to the Problem

- One possible solution is shown in the next slide.
- One disadvantage is that an extra route must be added to the network, thus using additional address space.
- Routing setup is more complicated (most hosts like to use simple default routes)

© 2003 Cisco Systems, Inc. All rights reserved.

Solution 1



E-1 Route Table

3.0 -> 1.2
4.0 -> 2.2

E-2 Route Table

1.0 -> 3.2
2.0 -> 3.2

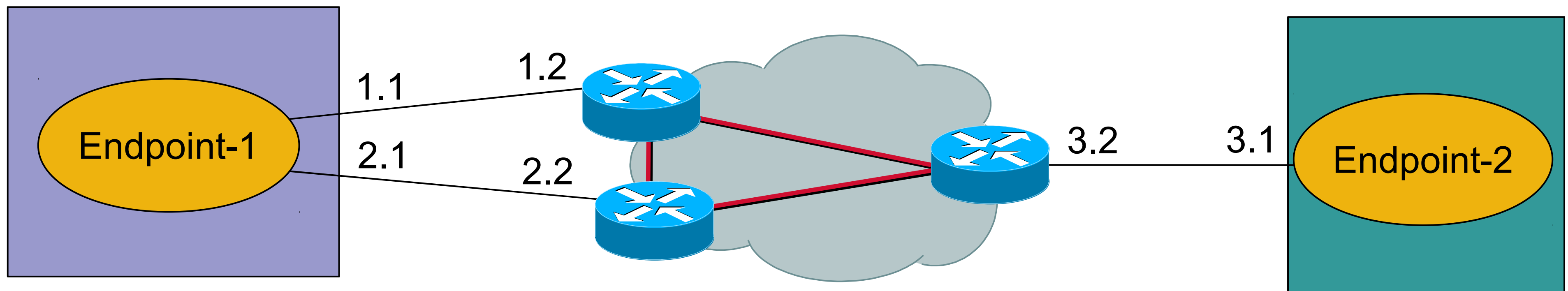
© 2003 Cisco Systems, Inc. All rights reserved.

A Simpler Solution

- **A simpler solution can be made by the assitance of the multi-homed host's routing table.**
- **It first must be setup to allow duplicate routes at any level in its routing table.**
- **Support must be added to query the routing table for an "alternate" route.**
- **When SCTP hits a set error threshold, it asks for an "alternate" route then the previously cached one .**

© 2003 Cisco Systems, Inc. All rights reserved.

Solution 2



E-1 Route Table

Default -> 1.2

Default -> 2.2

E-2 Route Table

1.0 -> 3.2

2.0 -> 3.2

© 2003 Cisco Systems, Inc. All rights reserved.

Auxiliary Packet Handling

- Sometimes, unexpected or "Out of the Blue" (OOTB) packets are received.
- In general, an OOTB packet has NO SCTP endpoint to communicate with (note these rules are only for SCTP protocol packets).
- When an OOTB packet is received, a specific set of rules must be followed.

© 2003 Cisco Systems, Inc. All rights reserved.

Auxiliary Packet Handling II

- 1) If the address is non-unicast, the packet is silently discarded.
- 2) If the packet holds an ABORT chunk, the packet is silently discarded.
- 3) If the OOTB is an INIT or COOKIE-ECHO, follow the setup procedures.
- 4) If it is a SHUTDOWN-ACK, send a SHUTDOWN-COMPLETE with the T bit set [more details in next section]

© 2003 Cisco Systems, Inc. All rights reserved.

Auxiliary Packet Handling III

- **If the OOTB is a SHUTDOWN-COMPLETE, silently discard the packet.**
- **If the OOTB is a COOKIE-ACK or ERROR, the packet should be silently discarded.**
- **For all other cases, send back an ABORT with the T bit set.**

  When the T bit is set, it indicates no TCB and the V-Tag is copied from the incoming packet to the outbound ABORT.

© 2003 Cisco Systems, Inc. All rights reserved.

Verification Tag Rules

- **All packets hold a V-Tag in the common header.**

  The V-Tag is a 32 bit nonce that each side picks during association setup (in the INIT and INIT-ACK chunks)

- **All packets received have the checksum calculated and the V-Tag verified.**
- **There is a set of rules for handling V-Tags just like there are for OOTB**

© 2003 Cisco Systems, Inc. All rights reserved.

Two Basic V-Tag Rules

Cisco.com

- When the packet does NOT contain an ABORT, INIT, SHUTDOWN-COMPLETE, or COOKIE-ECHO, two basic rules apply for V-Tags apply

  Rule 1: When sending packets to a peer, the V-Tag is set to the Initiate Tag the peer specified in the INIT or INIT-ACK

  Rule 2: When receiving an SCTP packet from a peer, the receiving endpoint must validate that the V-Tag matches the Initiate Tag it used in the INIT or INIT-ACK it sent.

© 2003 Cisco Systems, Inc. All rights reserved.

V-Tag Rules: INIT

- **For INIT packets, the following rules apply:**

    Rule 3: The sender of an INIT must set the V-Tag of the packet to zero.

    Rule 4: If the received packet has a V-Tag set to zero, the receiver must check for an INIT.

        If an INIT is present, the standard setup rules for SCTP are followed.

        Otherwise, an ABORT is sent with the T bit set.

© 2003 Cisco Systems, Inc. All rights reserved.

V-Tag Rules: ABORT

- **For packets carrying an ABORT, Rules 5 – 7 apply:**

  Rule 5: When sending an ABORT, the sender should try to populate the proper V-Tag in the common header, if known.

  Rule 6: If the V-Tag of the peer is not available, the sender will set the T bit and use (copy) the V-Tag from the received packet that is causing the ABORT

  Rule 7: When an ABORT chunk is present in a packet, it must be accepted if the V-Tag matches the expected value OR the T bit is set and the V-Tag matches the peer's V-Tag (i.e. the V-Tag used for outbound packets).

© 2003 Cisco Systems, Inc. All rights reserved.

V-Tag Rules: Shutdown Chunks

- **For packets carrying a SHUTDOWN-COMPLETE:**

  Rule 8: If a SHUTDOWN-ACK is received for an unknown association, send a SHUTDOWN-COMPLETE with the T bit set and the use the V-Tag from the SHUTDOWN-ACK.

- **When a SHUTDOWN-COMPLETE is received: If**

  the T bit is set, compare the received V-Tag with the peer's V-Tag to validate the SHUTDOWN-COMPLETE

  Otherwise, compare with your V-Tag to validate the SHUTDOWN-COMPLETE

© 2003 Cisco Systems, Inc. All rights reserved.

V-Tag Rules: COOKIE-ECHO

- **Packets carrying a COOKIE-ECHO have special handling, since the receiver generally has NO TCB:**

  Rule 9: When sending a COOKIE-ECHO the V-Tag used will be the Initiate Tag inside the INIT-ACK.

  Rule 10: Before comparing V-Tags, the rules for handling state cookies must be executed first. Then, the V-Tag may be verified.

    Some implementations do not bother to check the V-Tag when the state cookie's MAC has much stronger protection then the V-Tag

© 2003 Cisco Systems, Inc. All rights reserved.

Break

- **Questions**

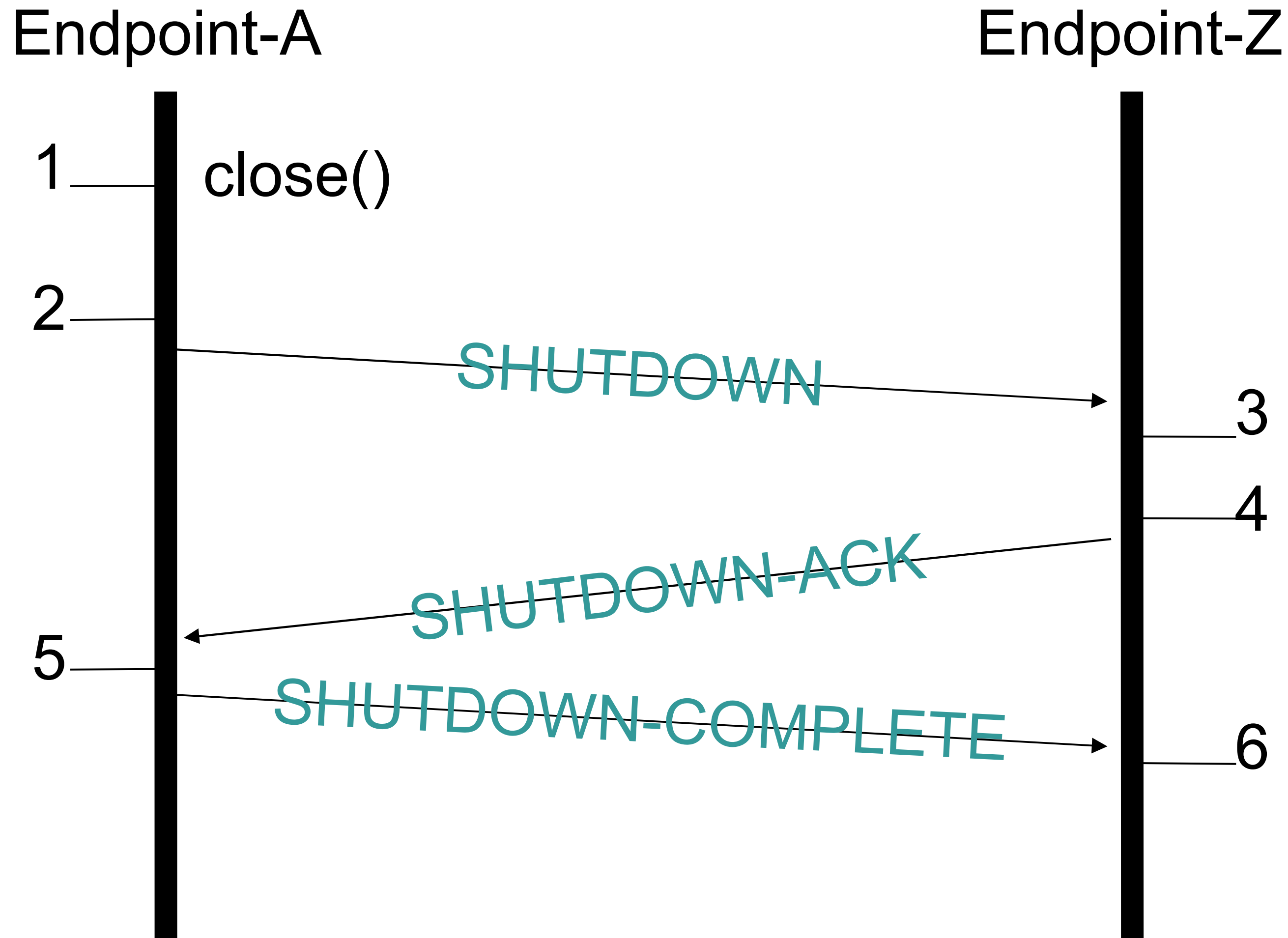
© 2003 Cisco Systems, Inc. All rights reserved.

How Do We Close an Association

- Like TCP, SCTP uses a 3-way handshake when closing an association
- Unlike TCP, SCTP does NOT support a half-closed state
- This means that once either endpoint closes an association, both sides are forced to close the association, sending appropriate notifications to the upper layer.

© 2003 Cisco Systems, Inc. All rights reserved.

The Shutdown Handshake



© 2003 Cisco Systems, Inc. All rights reserved.

User Closes

- There are six significant points in our shutdown handshake scenario.
- Point 1: the application issues a close() at Endpoint-A. At this point, NO NEW data can be sent from Endpoint-A.
- Point 2: the SCTP implementation at Endpoint-A has sent and received acknowledgement for all queued data (before the close). At this point, the endpoint sends a SHUTDOWN chunk.

© 2003 Cisco Systems, Inc. All rights reserved.

User Closes II

- Point 3: Endpoint-Z receives the SHUTDOWN so the upper layer is notified and NO NEW data will be accepted for transmission
- Point 4: Endpoint-Z has received acknowledgment for all its queued data so it sends a SHUTDOWN-ACK
- Point 5: Endpoint-A destroys the association/TCB and sends back a SHUTDOWN-COMPLETE
- Point 6: Endpoint-Z receives the SHUTDOWN-COMPLETE and destroys its association/TCB

© 2003 Cisco Systems, Inc. All rights reserved.

User Closes III

Cisco.com

- **Note that Points 1 and 2 may or may not be at the same moment in time depending on how much data is enqueued**
- **Note the same also holds true for Points 3 and 4.**

© 2003 Cisco Systems, Inc. All rights reserved.

The Shutdown: A Closer Detailed Look

- Let’s look from the perspective of the endpoint that initiates the shutdown sequence
- An application/upper layer initiates the shutdown sequence by either:
  - closing the socket
  - making an API call which invokes the 'shutdown' request
- This puts the endpoint into one of two states: SHUTDOWN-PENDING or SHUTDOWN-SENT

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details II

- **Assume there is data enqueue on both sides for this discussion, and Endpoint-A initiates the shutdown**
- **The local Endpoint-A then enters the SHUTDOWN-PENDING state.**

    While in this state and through to completion of the shutdown, the local endpoint will reject any attempt to send new data from the upper layer

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details III

- **Endpoint-A continues with normal data transfer sending all queued data to the peer endpoint**

  Note the peer (Endpoint-Z) has no idea that Endpoint-A is in the SHUTDOWN-PENDING state

- **Once all data has been acknowledged, Endpoint-A:**

  Starts a Shutdown Timer and a Shutdown-Guard Timer

  Sends a SHUTDOWN to the peer

  Enters the SHUTDOWN-SENT state

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details IV

- The peer upon receiving the SHUTDOWN enters the SHUTDOWN-RECEIVED state and informs its upper layer. From here on, no new data will be accepted by the remote endpoint from its upper layer.
- What happens if the SHUTDOWN gets lost? The Shutdown Timer will expire, causing a resend of the SHUTDOWN chunk.
- Any received DATA will cause the Shutdown Timer to restart (not the Shutdown-Guard Timer).

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details V

- In fact every time a DATA chunk arrives the Endpoint-A will answer with at minimum a SHUTDOWN and possibly a SHUTDOWN bundled with a SACK.
- Note that the delayed SACK algorithm is disabled during the SHUTDOWN-SENT state.
- Eventually Endpoint-Z will dequeue all of its data in the SHUTDOWN-RECIEVED state.
- At that point it will send a SHUTDOWN-ACK and start a local Shutdown timer.

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details VI

Cisco.com

- Endpoint-Z will resend the SHUTDOWN-ACK until it receives a SHUTDOWN-COMPLETE.
- In both cases of Shutdown timer expiration for Endpoint-A or Endpoint-Z the error thresholds are also incremented so there is a limit to the number of SHUTDOWN's and SHUTDOWN-ACK's that will be sent.
- Once Endpoint-A receives the SHUTDOWN-ACK it will stop its two timers and send back a SHUTDOWN-COMPLETE.

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details VII

- **After sending the SHUTDOWN-COMPLETE it will destroy the local TCB.**
- **So what does the second shutdown timer do? This timer is known as the shutdown guard timer (its not in RFC2960). What it does is provide an overall guard in case the peer is malicious and does not stop sending new data. If it expires the TCB is immediately destroyed. Note this timer is usually set to at least 60 seconds.**

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details XIII

- So you may have noticed an issue, what happens if Endpoint-A gets the SHUTDOWN-ACK but the SHUTDOWN-COMPLETE is lost?
- This is where the special rules discussed previously come in.
- Endpoint-A would then receive a resend of the SHUTDOWN-ACK but it has no TCB.
- So instead it send a SHUTDOWN-COMPLETE with the T bit set.

© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Details IX

- This allows Endpoint-Z to recover from lost SHUTDOWN-COMPLETEs.
- One other isolated set of events is also handled by special rules. If, after destroying the TCB Endpoint-A sends a INIT at the same time the SHUTDOWN-COMPLETE is lost what happens?
- The normal rules of sending a T bit SHUTDOWN-COMPLETE still apply but Endpoint-Z can also send an ERROR message.

© 2003 Cisco Systems, Inc. All rights reserved.

ABORT Chunk

- One other state cleanup mechanism is the ABORT chunk.
- When an application issues a abortive close the TCB is destroyed immediately. In this case an ABORT is sent.
- The ABORT, in cases of application controlled abort, contains the proper V-tag and would cause an immediate destruction of the peers TCB upon receipt
- The ABORT chunk is not reliable however.

© 2003 Cisco Systems, Inc. All rights reserved.

ABORT Chunk II

- If an ABORT is lost, the next packet sent to the endpoint that destroyed its TCB will be treated as OOTB.
- The response would then be a ABORT with the T bit set. The V-Tag would be that of the incoming packet.
- In cases of system restart you would also receive an ABORT with T bit set in response to any message (such as a Heartbeat).

© 2003 Cisco Systems, Inc. All rights reserved.

Extensions to SCTP

- SCTP, as we have seen, is very extensible.
- To extend SCTP, both new chunk types and parameter types can be added through new RFC's.
- SCTP implementations use the upper bits to determine how to handle unknown chunks and parameters.
- When designing extensions, one should take this upper bit handling into account

© 2003 Cisco Systems, Inc. All rights reserved.

Extensions II

Cisco.com

- IANA assigns chunk and parameter values when a NEW RFC goes through the IETF standards process.
- Usually, the Internet Draft will contain a “suggested” parameter or chunk value taking into account current existing extension documents.
- PR-SCTP as just advanced as the first extension to RFC status – RFC 3758, others will follow the slow standards process I am sure :-D

© 2003 Cisco Systems, Inc. All rights reserved.

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.**

© 2003 Cisco Systems, Inc. All rights reserved.

PR-SCTP II

- **This "skip message" is called a FORWARD-TSN (FWD-TSN) chunk.**
- **The FWD-TSN 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.**

© 2003 Cisco Systems, Inc. All rights reserved.

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.

© 2003 Cisco Systems, Inc. All rights reserved.

Other Extensions

Cisco.com

- Two other extensions are under development as well.
- The ADD-IP draft allows dynamic changes to an address set of an endpoint without restart of the association.
- The AUTH draft allows selected chunks to be “wrapped” with a signature. The draft is in fluctuation right now but its final form will be an implementation of the PBK-Draft (PBK stands for Purpose Built Keys).

© 2003 Cisco Systems, Inc. All rights reserved.

Break

- **Questions?**

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP: Similarities

- **Both use a handshake to setup and terminate the state (communication) relationship between peers**
- **Both have an abortive method to terminate the state**
- **Both provide a “reliable ordered” service:**
  - **Lost data is retransmitted**
  - **Data is (or can be) delivered in the order it was sent**
- **Both follow an AIMD-based congestion control mechanism.**

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP

- **SCTP uses a four-way handshake to setup an association. TCP uses a three-way handshake to setup a connection.**
- **However, this does not mean that data can start to be sent more quickly (relative to the start of the connection) with TCP.**
- **SCTP can exchange data on the third and fourth leg of its handshake. TCP in practicality does not (due to socket API issues).**

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP

- **SCTP delivers messages, not a “byte stream”**
  - **An application using TCP must “frame” its own messages**
- **SCTP streams allows “partially ordered” transfers**
  - **Escapes head of line blocking, while preserving order within each stream**
- **An SCTP sender can send all messages in a single ordered stream to achieve the same behavior as TCP.**

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP

- SCTP also provides an “reliable un-ordered” service for applications

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP

- TCP is a singly-homed protocol so a single interface failure can shutdown a connection. SCTP is multi-homed and can take advantage of all interfaces, addresses on a host.
- SACK support:
  - Optional in TCP, fundamental to SCTP
  - TCP SACK has a very limited segment space for specifying out of order segments
  - SCTP has a much larger "gap ack" space so that many sets of segments can be reported

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP

- **SCTP does not allow a half-closed state**

  Half-closed state is when one side is no longer allowed to send data but the other side can.

- **SCTP does NOT have a timed-wait state that will hold a connection from being made again within a specified time.**

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP and TCP: Security Considerations

- **SCTP uses the four-way handshake and the signed state cookie to protect against SYN flooding attacks**
- **SCTP uses a 32-bit random nonce to protect its packets from blind attackers**

  I-G version 10 prevents these from ever being revealed after association setup.

  TCP does not have this and is more subject to various forms of blind data and control segment injection attacks as we have recently seen in the news

© 2003 Cisco Systems, Inc. All rights reserved.

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()

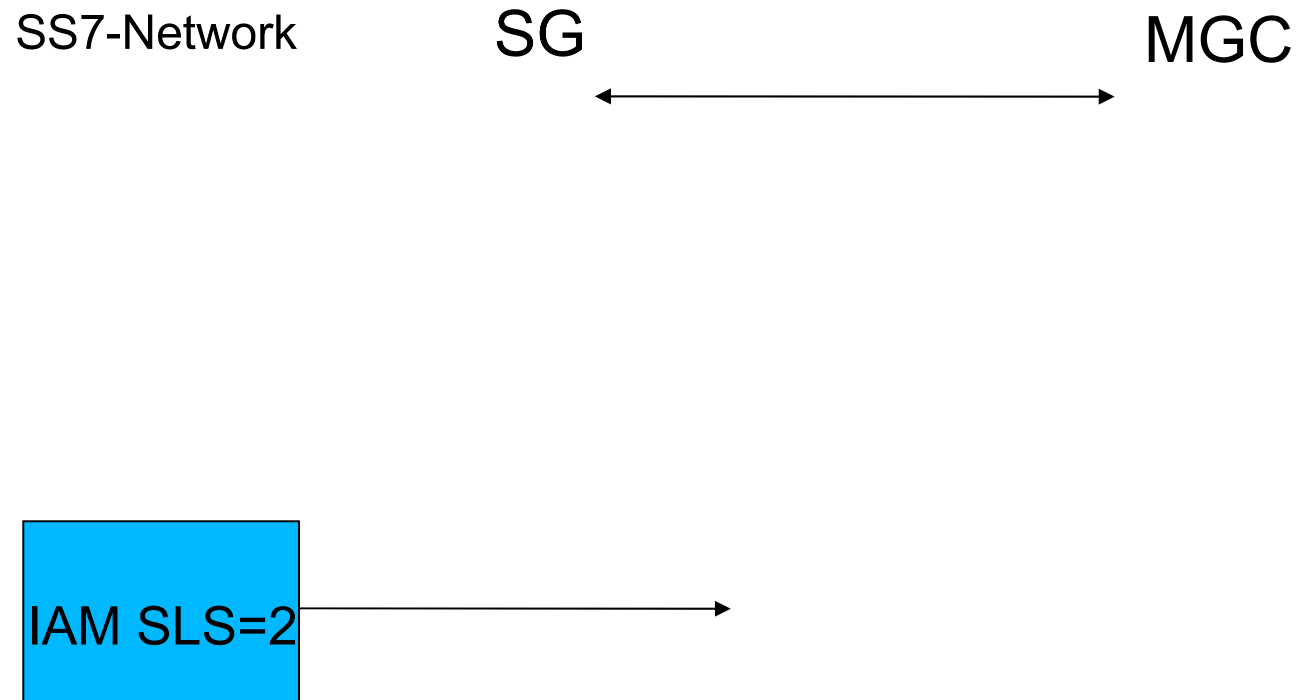
© 2003 Cisco Systems, Inc. All rights reserved.

More on Streams

- An example of using streams can be found in SS7 over IP (sigtran). Here various call messages will be routed to different streams so that a lost message on one call will not hold up another call. Usually the SLS index of SS7 is mapped onto a stream (SLS values range from 0 to 15 if I remember right :-D)
- A web client/server could use streams to display pictures in parallel instead of building multiple connections.

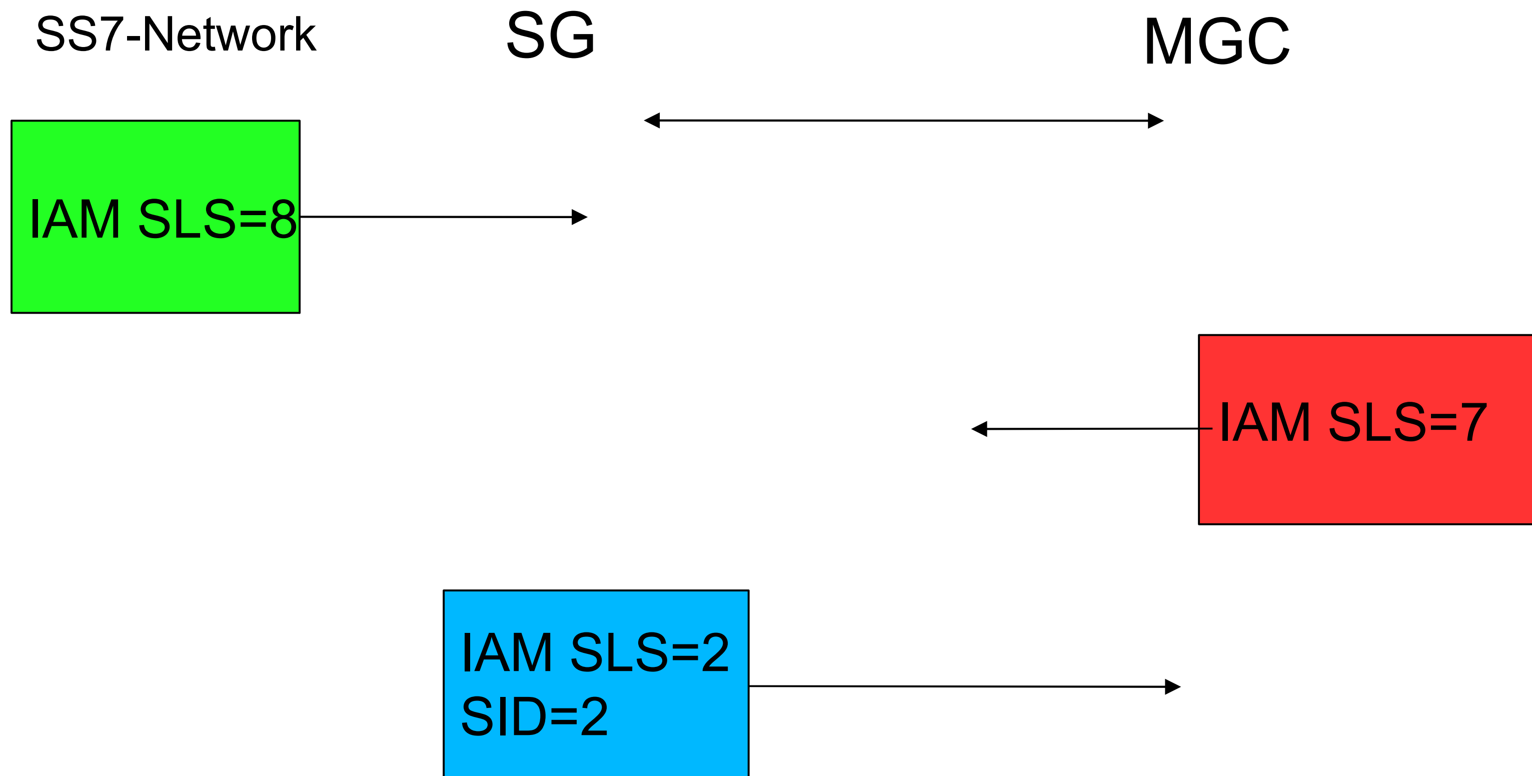
© 2003 Cisco Systems, Inc. All rights reserved.

A Stream Example



© 2003 Cisco Systems, Inc. All rights reserved.

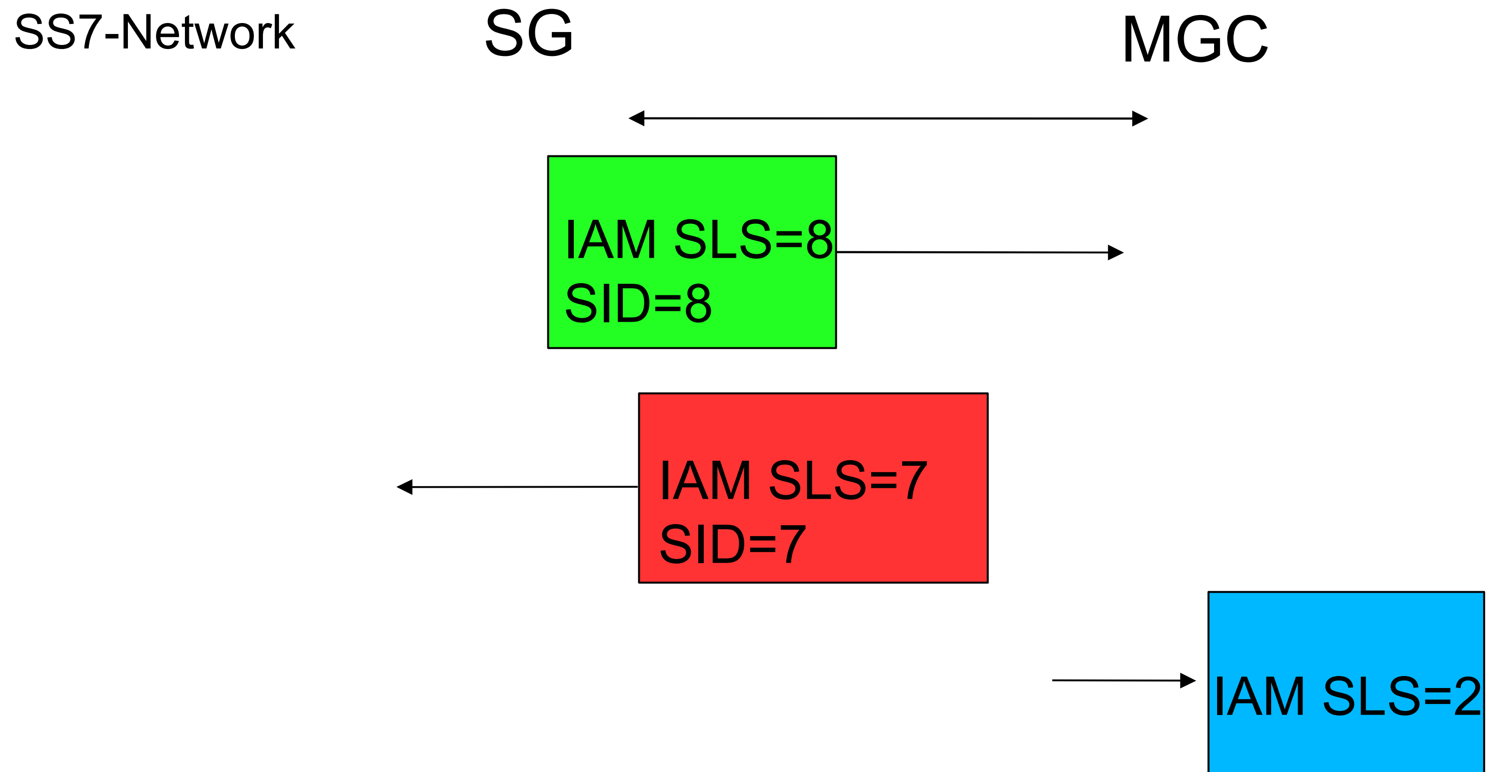
A Stream Example



© 2003 Cisco Systems, Inc. All rights reserved.

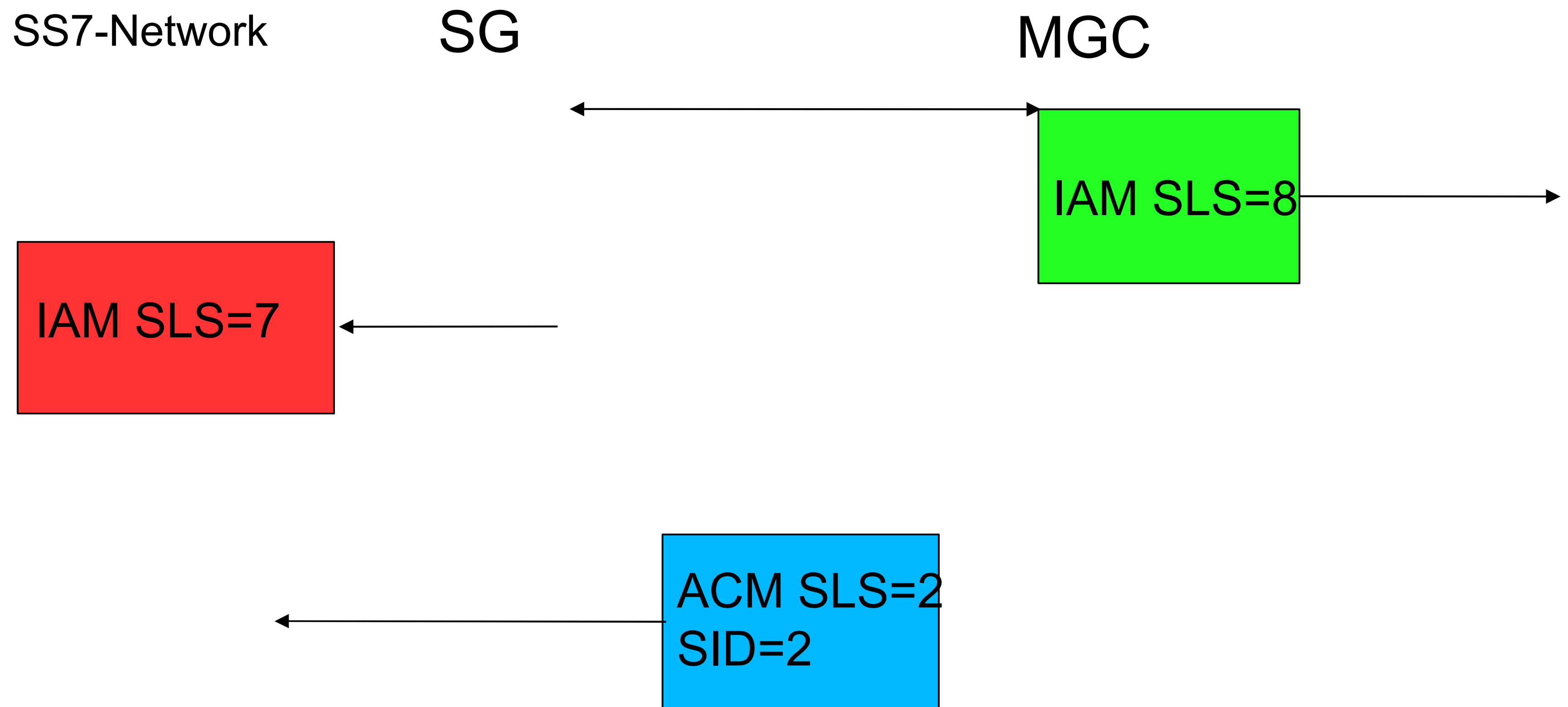
A Stream Example

Cisco.com



© 2003 Cisco Systems, Inc. All rights reserved.

A Stream Example



© 2003 Cisco Systems, Inc. All rights reserved.

Sockets API

Cisco.com

- Chapter 11 of the SCTP book discusses the socket API. This text is quite dated, but gives the reader a general idea how the socket API works.
- However, a better reference for SCTP socket API programming is the third revision of Stevens' Unix Network Programming.
- This new book has three comprehensive up-to-date chapters that detail the finer points of working with the SCTP socket API.

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Socket Types

- SCTP socket API comes in two forms: one-to-one and one-to-many.
- The one-to-many at one time was known by the “UDP style” socket. The one-to-one used to be called the a “TCP style” socket.
- So what is the purpose of each socket style and how can it be used?
- We will start with the one-to-one style.

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-One style

- The purpose of the one-to-one style socket is to provide a smooth transition mechanism for those applications running on TCP and wishing to move to SCTP.
- The same semantics used in TCP are used with this style.
- A server will typically open the socket, make a call to listen (to accept associations), and call accept, blocking upon the arrival of a new association.

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-One style

- The only notable difference between a TCP socket and a SCTP socket is the socket call uses IPPROTO_SCTP instead of IPPROTO_TCP (or 0).
- Two other common socket options that might be used in a TCP application have SCTP equivalents:

  TCP_NODELAY -> SCTP_NODELAY

  TCP_MAXSEG -> SCTP_MAXSEG

- SCTP has a host of other socket options as well which we will touch on further on.

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-One Style

Cisco.com

- Switching from TCP to SCTP becomes easy with this style of socket due to the few number of changes that have to be made.
- To give you an idea on this, note that I ported a version of mozilla with only two lines of change.
- Of course a quick change like I did in mozilla did not gain useage of SCTP streams but it does gain you the multi-homing aspects of SCTP.
- Other cavets of moving a TCP application are that there is NO half-close state, so if an application makes use of this, that code will need to be re-written.

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-One Style

- Another thing that MAY be an issue is that some TCP applications will write a 2 or 4 byte record length followed by that many bytes of data. If an application behaves in this way SCTP will make each write a single message.
- These two message would most likely be bundled together but overall this increases the overhead on the wire.
- So what does a typical application using the one-to-one style socket look like?

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-One Example Server


```
int sd, newfd, sosz;
struct sockaddr_in6 sin6;
sosz = sizeof(sin6);
sd = socket(AF_INET6, SOCK_STREAM, IPPROTO_SCTP);
listen(sd, 1);
while (1) {
    newfd = accept(sd, (struct sockaddr *)&sin6, &sosz)
    do_child_stuff(newfd, &sin6, sosz);
}
```


© 2003 Cisco Systems, Inc. All rights reserved.

One-to-Many style

- **A typical server using a one-to-many style socket will do a socket() call, followed by a listen() and recvfrom().**
- **A typical client will just sendto() the server of his choice.**
- **Note that the connect() and accept() call are not needed.**
- **The connect() call can be done by either side (server or client) but it is not needed.**

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-Many style

- **Note that this style is more like what a UDP client/server would look like thus the previous name.**
- **So what does a typical one-to-many style server look like?**

© 2003 Cisco Systems, Inc. All rights reserved.

One-to-many Example Server

Cisco.com

```
int sd, newfd, sosz, msg_flags;
struct sockaddr_in6 sin6;
struct sndrcvinfo snd_rcv;
char buf[8000];
sosz = sizeof(sin6);
sd = socket(AF_INET6, SOCK_SEQPKT, IPPROTO_SCTP);
listen(sd, 1);
while (1) {
    len = sctp_recvmsg(sd, buf, sizeof(buf), (sockaddr *)&sin6, &sosz,
                       &snd_rcv, &msg_flags);
    do_child_stuff(newfd, buf, len, &sin6, &snd_rcv, msg_flags);
}
```


© 2003 Cisco Systems, Inc. All rights reserved.

One-to-many Description

Cisco.com

- Note in the previous example we introduced the first of several new/extra calls sctp_recvmsg().
- This call is usually built as a library call (its not a true system call in most cases).
- It provides a convenince function that makes it easy to find out specific information about stream id and other auxilary information that SCTP can provide upon receiving messages.
- But before we get in to the details of all the extra calls we need to discuss notifications.

© 2003 Cisco Systems, Inc. All rights reserved.

SCTP Notifications

Cisco.com

- The SCTP stack, at times, has information it may wish to share with its application (or Upper Layer Protocol ... ULP).
- The ULP can turn off and on specific notifications via a socket options call.
- By default ALL notifications are off.
- So how does one get a notification?
- By reading data and looking at the msg_flags, if the message read is a notification, then "MSG_NOTIFICATION" is contained within the msg_flags argument upon return.

© 2003 Cisco Systems, Inc. All rights reserved.

More on Notifcations

- If the user does NOT use the sctp_recvmsg() call, then you can also gain access to this flag using the recvmsg() system call and look at the msg.msg_flags field (most library calls implementing sctp_recvmsg() use recvmsg() and copy the msg.msg_flags into the int* passed to sctp_recvmsg().
- So what do you get when you read a notification?
- A union is read in that looks as follows:

© 2003 Cisco Systems, Inc. All rights reserved.

Notification Union

Cisco.com

```
/* notification event */
union sctp_notification {
        struct sctp_tlv sn_header;
        struct sctp_assoc_change sn_assoc_change;
        struct sctp_paddr_change sn_paddr_change;
        struct sctp_remote_error sn_remote_error;
        struct sctp_send_failed         sn_send_failed;
        struct sctp_shutdown_event
sn_shutdown_event;
        struct sctp_adaption_event sn_adaption_event;
        struct sctp_pdapi_event sn_pdapi_event;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Deciphering Notifications

- **Every Notification uses a TLV format as illustrated below:**

```
struct sctp_tlv {
        u_int16_t sn_type;
        u_int16_t sn_flags;
        u_int32_t
sn_length;
};
```

- **So what type of notifications do you get?**

© 2003 Cisco Systems, Inc. All rights reserved.

Association change

- **SCTP_ASSOC_CHANGE - indicates that a change has occurred in regard to an association (e.g.a new association is now present on the socket or an association has went away/failed).**

```
struct sctp_assoc_change {
        u_int16_t sac_type;
        u_int16_t sac_flags;
        u_int32_t sac_length;
        u_int16_t sac_state;
        u_int16_t sac_error;
        u_int16_t
sac_outbound_streams;
        u_int16_t sac_inbound_streams;
        sctp_assoc_t sac_assoc_id;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

A Peer Address Change event

- **An SCTP_PEER_ADDR_CHANGE will indicate that something has occurred with the address (in-service, out-of-service, added, deleted etc).**

```
/* Address events */
struct sctp_paddr_change {
        u_int16_t spc_type;
        u_int16_t spc_flags;
        u_int32_t spc_length;
        struct sockaddr_storage
spc_aaddr;
        u_int32_t spc_state;
        u_int32_t spc_error;
        sctp_assoc_t spc_assoc_id;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

A Remote Error Event

- **An SCTP_REMOTE_ERROR will communciate a remote error sent by the peer, this will be in the form of a TLV and may indicate some internal stack debugging information as to why an association was closed.**

```
/* remote error events */
struct sctp_remote_error {
        u_int16_t sre_type;
        u_int16_t sre_flags;
        u_int32_t sre_length;
        u_int16_t sre_error;
        sctp_assoc_t
sre_assoc_id;
        u_int8_t  sre_data[4];
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Send Failure

- **An SCTP_SEND_FAILED will indicate that data queued was not acknowledged by the peer and will include the actual data that was attempted to be sent (within some limits). This may occur due to partial reliablility or right before an association comes down.**

```
/* data send failure event */
struct sctp_send_failed {
        u_int16_t ssf_type;
        u_int16_t ssf_flags;
        u_int32_t ssf_length;
        u_int32_t ssf_error;
        struct sctp_sndrcvinfo
ssf_info;
        sctp_assoc_t ssf_assoc_id;
        u_int8_t ssf_data[4];
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Shutdown Event

Cisco.com

- **An SCTP_SHUTDOWN_EVENT indicates that a graceful shutdown as occurred on an association.**

```
/* shutdown event */
struct sctp_shutdown_event {
        u_int16_t       sse_type;
        u_int16_t       sse_flags;
        u_int32_t       sse_length;
        sctp_assoc_t
sse_assoc_id;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Adaption Layer Event

Cisco.com

- **An SCTP_ADAPTION_INDICATION is a part of the add-ip extension and allows an upper layer to communicate an integer at startup informing the peer what type of ULP is being operated (iSCSI, RDMA, ?)**

```
/* Adaption layer indication stuff */
struct sctp_adaption_event {
        u_int16_t       sai_type;
        u_int16_t       sai_flags;
        u_int32_t       sai_length;
        u_int32_t
sai_adaption_ind;
        sctp_assoc_t    sai_assoc_id;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Partial Delivery Event

- **An SCTP_PARTIAL_DELIVERY_EVENT will indicate when something has went wrong on a partial delivery that has been begun (e.g. The association closed or the message was skipped via partial reliability).**

```
/* pdapi indications */
struct sctp_pdapi_event {
        u_int16_t       pdapi_type;
        u_int16_t       pdapi_flags;
        u_int32_t       pdapi_length;
        u_int32_t
pdapi_indication;
        sctp_assoc_t    pdapi_assoc_id;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Common to events is the assoc_id

- Note that all events include something called an assoc_id.
- This is a unique identifier to the association.
- Many of the extended SCTP calls can use this for sending and or configuring an association with socket options.
- An application that wishes to use assoc_id's needs to be aware of association id re-use and must pay close attention to failure and closing events.

© 2003 Cisco Systems, Inc. All rights reserved.

So how does one get notifications?

- **The socket option SCTP_EVENTS is used to turn on/off all of the various events by passing it the following structure:**

```
/* On/Off setup for subscription to events */
struct sctp_event_subscribe {
        u_int8_t sctp_data_io_event;
        u_int8_t sctp_association_event;
        u_int8_t sctp_address_event;
        u_int8_t sctp_send_failure_event;
        u_int8_t sctp_peer_error_event;
        u_int8_t sctp_shutdown_event;
        u_int8_t
sctp_partial_delivery_event;
        u_int8_t sctp_adaption_layer_event;
};
```


© 2003 Cisco Systems, Inc. All rights reserved.

Subscribing Part II

- **Placing a '1' in the respective event type field turns an event on.**
- **Placing a '0' in the respective event type field turns an event off.**
- **Note that these events are the standard ones so far, other events may be added as various extensions work their way through the IETF.**

© 2003 Cisco Systems, Inc. All rights reserved.

Socket Options

Cisco.com

- SCTP provides a host of socket options to perform a mirad of operations.
- Some have unique structures others just turn things on and off with boolean's or integers.
- SCTP_NODELAY – Turns on/off the nagel algorithm (or other delay) similar to TCP.
- SCTP_MAXSEG – Sets/Gets a value for the SCTP fragmentation point (an integer is passed). Note that its possible that the value the system uses is smaller than what you set.

© 2003 Cisco Systems, Inc. All rights reserved.

More Socket Options

- SCTP_ASSOCINFO – Retrieve or Set various information about an association. Note that not all fields in the structure are writeable.
- SCTP_AUTOCLOSE – Sets a idle time wherein an association will automatically close. For one-to-many style servers this can be used so that no connection state needs to be maintained by the application.
- SCTP_ADAPTION_LAYER – Set or Get the 32 bit adaption layer indication that will be sent with INIT's or INIT-ACK's.

© 2003 Cisco Systems, Inc. All rights reserved.

More Socket Options

Cisco.com

- SCTP_DEFAULT_SEND_PARAM – set or get the default sending parameters (stream number, ppid context and other fields in the sctp_sndrcvinfo structure).
- SCTP_DISABLE_FRAGMENTS – boolean that will disable SCTP fragmentation. Note that if fragmentation is disabled, sends larger than the fragment point will be rejected with an error return code.
- SCTP_EVENTS – we saw this one earlier, used to set what notification events we wish to see.

© 2003 Cisco Systems, Inc. All rights reserved.

More on Socket Options

- SCTP_GET_PEER_ADDR_INFO – get information on a peers address. The information returned includes the cwnd, srtt, rto and path mtu.
- SCTP_I_WANT_MAPPED_V4_ADDR – this boolean is normally on by default and makes it so an Ipv6 socket will map V4 address to V6. If this is turned off then V4 addresses will be received up a V6 socket.
- SCTP_INITMSG – Can be used to get or set the default INIT/INIT-ACK settings such as number of streams allowed in or requested out.

© 2003 Cisco Systems, Inc. All rights reserved.

Even More on socket options

Cisco.com

- SCTP_PEER_ADDR_PARAMS – allows an endpoint to get or set the heart beat interval and/or path maximum retransmist on a specific peer address.
- SCTP_PRIMARY_ADDR – Allows an application to specify a peers address has the "primary" address.
- SCTP_RTOINFO – get or set the RTO information RTO.min, RTO.max and RTO.initial.
- SCTP_SET_PEER_PRIMARY_ADDR – Allows an endpoint to request that the peer change its primary address to the one specified (note this will only suceed if the peer supports the ADD-IP extension).

© 2003 Cisco Systems, Inc. All rights reserved.

Final socket option page

Cisco.com

- SCTP_STATUS – allows an application to retrieve a number of various parameters and stats with respect to a specific association.
- As you can see there are a LOT of options. If you will there is a knob for about most things someone would want to do to a transport connection.
- The purpose of all of these knobs is to give the application better control of the transport.
- If you plan on using any of these options I would highly recommend getting the UNP 3rd edtion. This gives all the details you will need to use these effectively (with examples).

© 2003 Cisco Systems, Inc. All rights reserved.

Extended “system calls”.

- sctp_connectx – Allows a user to specify multiple address to attempt to connect too.
- sctp_bindx – Allows an application to bind a set of addresses instead of one or all addresses.
- sctp_opt_info – Some implementations do not support a getsockopt() call that allows data to be passed both ways (some of the calls need an association id to get information). Use this call to be compatible with all implementations.

© 2003 Cisco Systems, Inc. All rights reserved.

Extended “system calls”

- **sctp_getpaddrs – This call will return a block of memory holding the peers addresses currently part of the association.**
- **sctp_freepaddrs – This call is used to release the memory back that the sctp_getpaddrs call allocated.**
- **sctp_getladdrs – This call will return a block of memory holding the local addresses bound to an association.**
- **sctp_freeladdrs – This call should be used to release the memory allocated by sctp-getladdrs back to the system.**

© 2003 Cisco Systems, Inc. All rights reserved.

Extended “system calls”

Cisco.com

- **sctp_sendmsg – This call will allow the caller to specify on the command line things like the stream number and other SCTPish information to be sent with a message.**
- **sctp_send – This call has a similar purpose to sctp_sendmsg but instead of a large number of command line options, a sctp_sendrcvinfo structure is used to pass the relevant information.**
- **sctp_recvmsg – This call (as we saw previously) is used to receive a message but also a sctp_sendrcvinfo structure with details on the message (e.g. The stream number and stream sequence number).**

© 2003 Cisco Systems, Inc. All rights reserved.

Extended "system calls"

Cisco.com

- sctp_peeloff – this call is used to convert a single association that is part of a one-to-many socket into an individual new socket descriptor that is a one-to-one socket.
- [Phil::: Should we go throught each of these and put signatures??]
- [How do we end? A big example]

© 2003 Cisco Systems, Inc. All rights reserved.