Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 21, 2007

R. Stewart
Cisco Systems, Inc.
Q. Xie
Motorola, Inc.
M. Tuexen
Univ. of Applied Sciences Muenster
S. Maruyama
M. Kozuka
Kyoto University
June 19, 2007

Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration
draft-ietf-tsvwg-addip-sctp-22.txt

Status of this Memo

Copyright Notice

Abstract

A local host may have multiple points of attachment to the Internet,

giving it a degree of fault tolerance from hardware failures. Stream Control Transmission Protocol (SCTP) [I-D.ietf-tsvwg-2960bis] was developed to take full advantage of such a multi-homed host to provide a fast failover and association survivability in the face of such hardware failures. This document describes an extension to SCTP that will allow an SCTP stack to dynamically add an IP Addresses to an SCTP association, dynamically delete an IP addresses from an SCTP association, and to request to set the primary address the peer will use when sending to an endpoint.

Table of Contents

1. Introduction

A local host may have multiple points of attachment to the Internet, giving it a degree of fault tolerance from hardware failures. SCTP was developed to take full advantage of such a multi-homed host to provide a fast failover and association survivability in the face of such hardware failures. However, many modern computers allow for the dynamic addition and deletion of network cards (sometimes termed a hot-pluggable interface). Complicate this with the ability of a provider, in IPv6, to dynamically renumber a network, and there still is a gap between full fault tolerance and the currently defined SCTP protocol. No matter if a card is added or an interface is renumbered, in order to take advantage of this new configuration, the transport association must be restarted. For many fault tolerant applications this restart is considered an outage and is undesirable.

This document describes an extension to SCTP to attempt to correct this problem for the more demanding fault tolerant application. This extension will allow an SCTP stack to:

- Dynamically add an IP Addresses to an association.
- Dynamically delete an IP Addresses from an association.
- Request to set the primary address the peer will use when sending to an endpoint.

The dynamic addition and subtraction of IP addresses allows an SCTP association to continue to function through host and network reconfigurations. These changes, brought on by provider or user action, may mean that the peer would be better served by using the newly added address, however this information may only be known by the endpoint that had the reconfiguration occur. In such a case this extension allows the local endpoint to advise the peer as to what it thinks is the better primary address that the peer should be using.

One last thing this extension adds is a small 32 bit integer, called an adaptation indication, that can be exchanged at startup. This is useful for applications where there is one or more specific layers below the application, yet still above SCTP. In such a case the exchange of this indication can allow the proper layer to be enabled below the application.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Serial Number Arithmetic

It is essential to remember that the actual ASCONF Sequence Number space is finite, though very large. This space ranges from 0 to 2**32 - 1. Since the space is finite, all arithmetic dealing with ASCONF Sequence Numbers MUST be performed modulo 2**32. This unsigned arithmetic preserves the relationship of sequence numbers as they cycle from 2**32 - 1 to 0 again. There are some subtleties to computer modulo arithmetic, so great care should be taken in programming the comparison of such values. When referring to ASCONF Sequence Numbers, the symbol "=<" means "less than or equal"(modulo 2**32).

Comparisons and arithmetic on ASCONF sequence numbers in this document SHOULD use Serial Number Arithmetic as defined in [RFC1982] where SERIAL_BITS = 32.

ASCONF Sequence Numbers wrap around when they reach 2**32 - 1. That is, the next ASCONF Sequence Number an ASCONF chunk MUST use after transmitting ASCONF Sequence Number = 2**32 - 1 is 0.

Any arithmetic done on Stream Sequence Numbers SHOULD use Serial Number Arithmetic as defined in [RFC1982] where SERIAL_BITS = 16. All other arithmetic and comparisons in this document uses normal arithmetic.

4. Additional Chunks and Parameters

This section describes the addition of two new chunks and, seven new parameters to allow:

- Dynamic addition of IP Addresses to an association.
- Dynamic deletion of IP Addresses from an association.
- A request to set the primary address the peer will use when sending to an endpoint.

Additionally, this section describes three new error causes that support these new chunks and parameters.

4.1. New Chunk Types

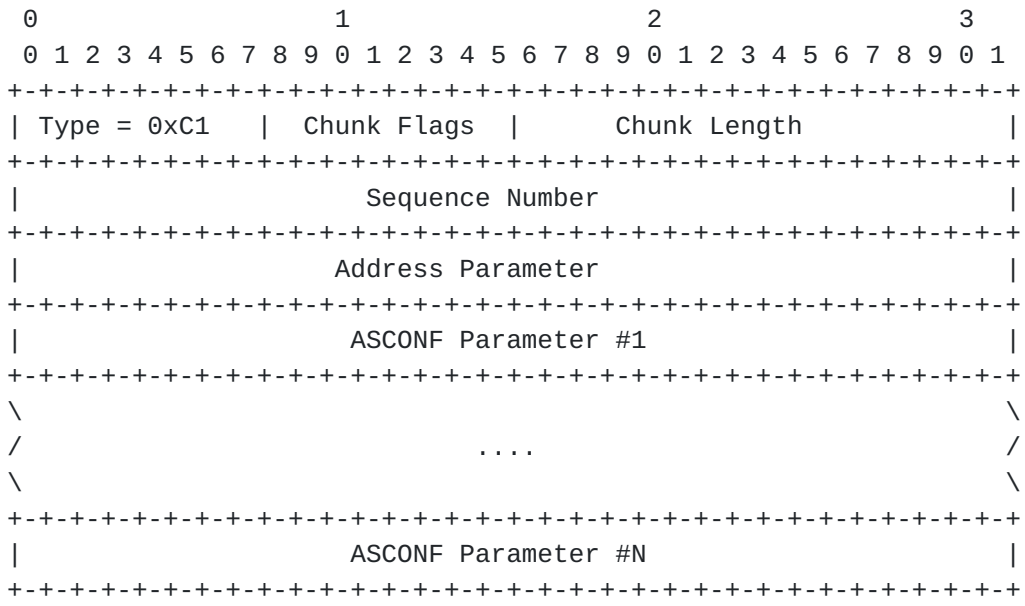
This section defines two new chunk types that will be used to transfer the control information reliably. Table 1 illustrates the two new chunk types.

Chunk Type	Chunk Name	
0xC1	Address Configuration Change Chunk	(ASCONF)
0x80	Address Configuration Acknowledgment	(ASCONF-ACK)

Table 1: Address Configuration Chunks

4.1.1. Address Configuration Change Chunk (ASCONF)

This chunk is used to communicate to the remote endpoint one of the configuration change requests that MUST be acknowledged. The information carried in the ASCONF Chunk uses the form of a Type-Length-Value (TLV), as described in "3.2.1 Optional/Variable-length Parameter Format" in [I-D.ietf-tsvwg-2960bis] for all variable parameters. This chunk MUST be sent in an authenticated way by using the mechanism defined in [I-D.ietf-tsvwg-sctp-auth]. If this chunk is received unauthenticated it MUST be silently discarded as described in [I-D.ietf-tsvwg-sctp-auth].



Sequence Number : 32 bits (unsigned integer)

This value represents a Sequence Number for the ASCONF Chunk. The valid range of Sequence Number is from 0 to 4294967295 (2**32 - 1). Sequence Numbers wrap back to 0 after reaching 4294967295.

Address Parameter : 8 or 20 bytes (depending on the address type)

This field contains an address parameter, either IPv6 or IPv4, from [I-D.ietf-tsvwg-2960bis]. The address is an address of the sender of the ASCONF Chunk, the address MUST be considered part of the

association by the peer endpoint (the receiver of the ASCONF Chunk). This field may be used by the receiver of the ASCONF to help in finding the association. If the address 0.0.0.0 or ::0 is provided the receiver MAY lookup the association by other information provided in the packet. This parameter MUST be present in every ASCONF message, i.e. it is a mandatory TLV parameter.

Note: the host name address MUST NOT be sent and MUST be ignored if received in any ASCONF message.

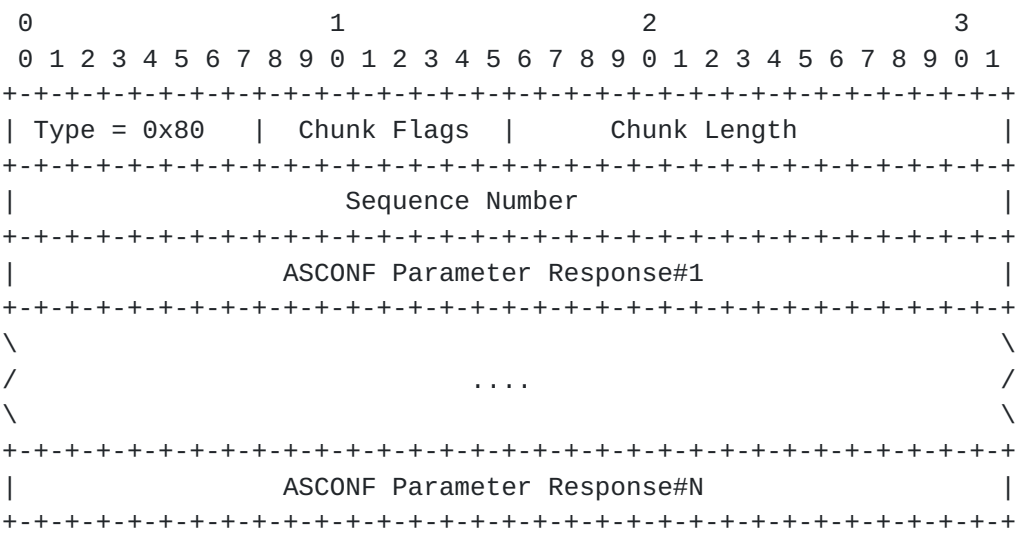
It should be noted that the ASCONF Chunk format requires the receiver to report to the sender if it does not understand the ASCONF Chunk. This is accomplished by setting the upper bits in the chunk type as described in [I-D.ietf-tsvwg-2960bis]. section 3.2. Note that the upper two bits in the ASCONF Chunk are set to one. As defined in [I-D.ietf-tsvwg-2960bis] section 3.2, when setting these upper bits in this manner the receiver that does not understand this chunk MUST skip the chunk and continue processing, and report in an Operation Error Chunk using the 'Unrecognized Chunk Type' cause of error. This will NOT abort the association but indicates to the sender that it MUST not send any further ASCONF chunks.

ASCONF Parameter: TLV format

Each Address configuration change is represented by a TLV parameter as defined in Section 4.2. One or more requests may be present in an ASCONF Chunk.

4.1.2. Address Configuration Acknowledgment Chunk (ASCONF-ACK)

This chunk is used by the receiver of an ASCONF Chunk to acknowledge the reception. It carries zero or more results for any ASCONF Parameters that were processed by the receiver. This chunk MUST be sent in an authenticated way by using the mechanism defined in [I-D.ietf-tsvwg-sctp-auth]. If this chunk is received unauthenticated it MUST be silently discarded as described in [I-D.ietf-tsvwg-sctp-auth].



Sequence Number : 32 bits (unsigned integer)

This value represents the Sequence Number for the received ASCONF Chunk that is acknowledged by this chunk. This value is copied from the received ASCONF Chunk.

ASCONF Parameter Response : TLV format

The ASCONF Parameter Response is used in the ASCONF-ACK to report status of ASCONF processing. By default, if a responding endpoint does not include any Error Cause, a success is indicated. Thus a sender of an ASCONF-ACK MAY indicate complete success of all TLVs in an ASCONF by returning only the Chunk Type, Chunk Flags, Chunk Length (set to 8) and the Sequence Number.

4.2. New Parameter Types

The seven new parameters added follow the format defined in section 3.2.1 of [I-D.ietf-tsvwg-2960bis]. Tables 2, 3 and 4 describe the parameters.

Address Configuration Parameters	Parameter Type
Set Primary Address	0xC004
Adaptation Layer Indication	0xC006
Supported Extensions	0x8008

Table 2: Parameters that can be used in INIT/INIT-ACK chunk

Address Configuration Parameters	Parameter Type
Add IP Address	0xC001
Delete IP Address	0xC002
Set Primary Address	0xC004

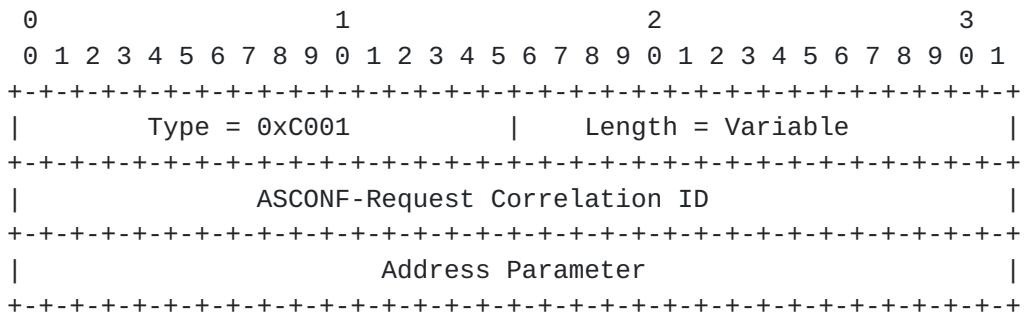
Table 3: Parameters used in ASCONF Parameter

Address Configuration Parameters	Parameter Type
Error Cause Indication	0xC003
Success Indication	0xC005

Table 4: Parameters used in ASCONF Parameter Response

Any parameter that appears where it is not allowed (for example a 0xC002 parameter appearing within an INIT or INIT-ACK) MAY be responded to with an ABORT by the receiver of the invalid parameter. If the receiver chooses NOT to abort, the parameter MUST be ignored. A robust implementation SHOULD ignore the parameter and leave the association intact.

4.2.1. Add IP Address



ASCONF-Request Correlation ID: 32 bits

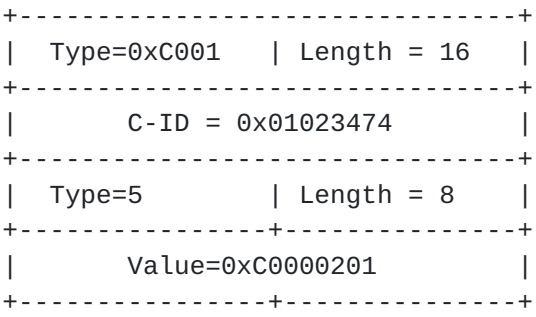
This is an opaque integer assigned by the sender to identify each request parameter. The receiver of the ASCONF Chunk will copy this 32 bit value into the ASCONF Response Correlation ID field of the ASCONF-ACK response parameter. The sender of the ASCONF can use this same value in the ASCONF-ACK to find which request the response is for. Note that the receiver MUST NOT change this 32 bit value.

Address Parameter: TLV

This field contains an IPv4 or IPv6 address parameter as described in

3.3.2.1 of [I-D.ietf-tsvwg-2960bis]. The complete TLV is wrapped within this parameter. It informs the receiver that the address specified is to be added to the existing association. This parameter MUST NOT contain a broadcast or multicast address. If the address 0.0.0.0 or ::0 is provided, the source address of the packet MUST be added.

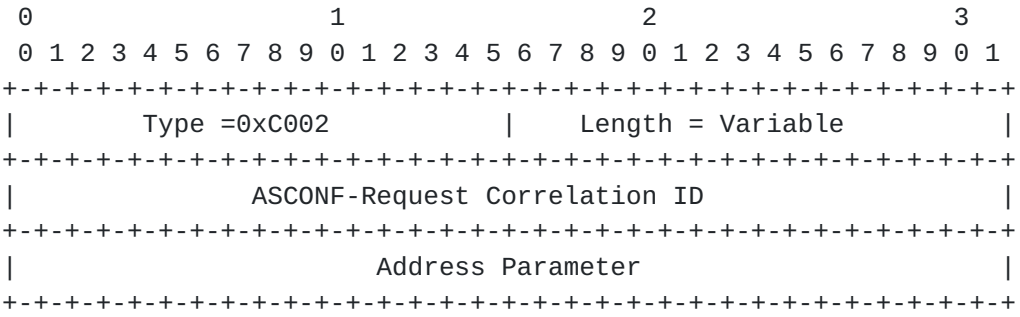
An example TLV requesting that the IPv4 address 192.0.2.1 be added to the association would look as follows:



Valid Chunk Appearance

The Add IP Address parameter may only appear in the ASCONF Chunk type.

4.2.2. Delete IP Address



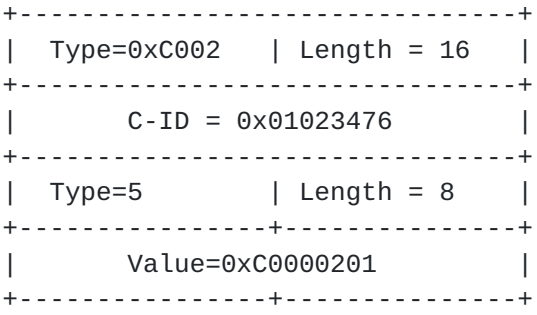
ASCONF-Request Correlation ID: 32 bits

This is an opaque integer assigned by the sender to identify each request parameter. The receiver of the ASCONF Chunk will copy this 32 bit value into the ASCONF Response Correlation ID field of the ASCONF-ACK response parameter. The sender of the ASCONF can use this same value in the ASCONF-ACK to find which request the response is for. Note that the receiver MUST NOT change this 32 bit value.

Address Parameter: TLV

This field contains an IPv4 or IPv6 address parameter as described in 3.3.2.1 of [I-D.ietf-tsvwg-2960bis]. The complete TLV is wrapped within this parameter. It informs the receiver that the address specified is to be removed from the existing association. This parameter MUST NOT contain a broadcast or multicast address. If the address 0.0.0.0 or ::0 is provided, all addresses of the peer except the source address of the packet MUST be deleted.

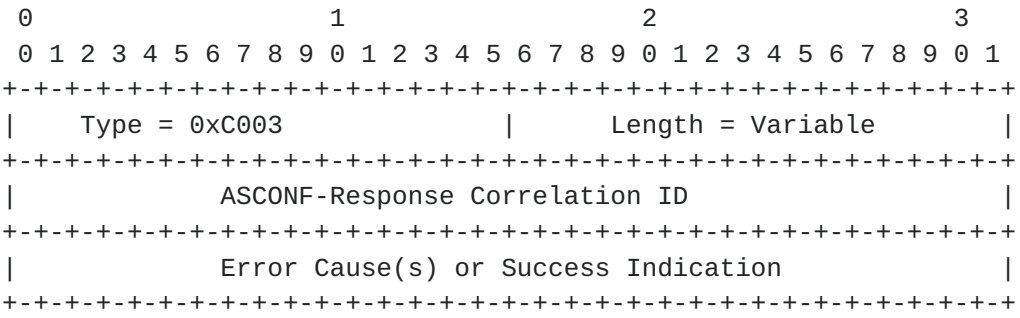
An example TLV deleting the IPv4 address 192.0.2.1 from an existing association would look as follows:



Valid Chunk Appearance

The Delete IP Address parameter may only appear in the ASCONF Chunk type.

4.2.3. Error Cause Indication



ASCONF-Response Correlation ID: 32 bits

This is an opaque integer assigned by the sender to identify each request parameter. The receiver of the ASCONF Chunk will copy this 32 bit value from the ASCONF-Request Correlation ID into the ASCONF Response Correlation ID field so the peer can easily correlate the request to this response. Note that the receiver MUST NOT change this 32 bit value.

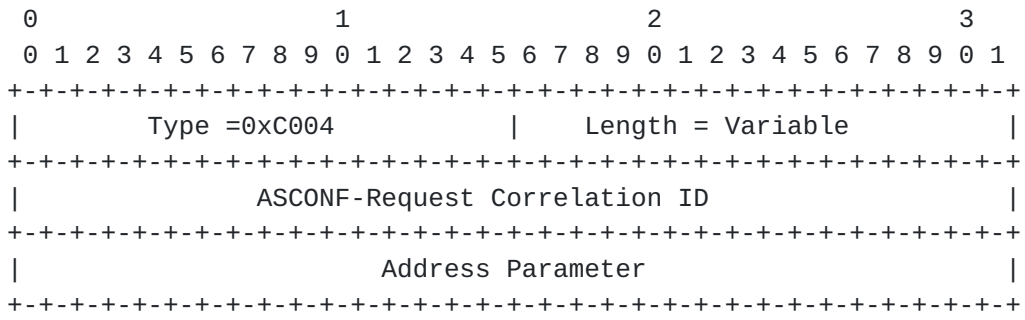
Error Cause(s): TLV(s)

When reporting an error this response parameter is used to wrap one or more standard error causes normally found within an SCTP Operational Error or SCTP Abort (as defined in [I-D.ietf-tsvwg-2960bis]). The Error Cause(s) follow the format defined in section 3.3.10 of [I-D.ietf-tsvwg-2960bis].

Valid Chunk Appearance

The Error Cause Indication parameter may only appear in the ASCONF-ACK Chunk type.

4.2.4. Set Primary IP Address



ASCONF-Request Correlation ID: 32 bits

This is an opaque integer assigned by the sender to identify each request parameter. The receiver of the ASCONF Chunk will copy this 32 bit value into the ASCONF Response Correlation ID field of the ASCONF-ACK response parameter. The sender of the ASCONF can use this same value in the ASCONF-ACK to find which request the response is for. Note that the receiver MUST NOT change this 32 bit value.

Address Parameter: TLV

This field contains an IPv4 or IPv6 address parameter as described in 3.3.2.1 of [I-D.ietf-tsvwg-2960bis]. The complete TLV is wrapped within this parameter. It requests the receiver to mark the specified address as the primary address to send data to (see section 5.1.2 of [I-D.ietf-tsvwg-2960bis]). The receiver MAY mark this as its primary upon receiving this request. If the address 0.0.0.0 or ::0 is provided, the receiver MAY mark the source address of the packet as its primary.

An example TLV requesting that the IPv4 address 192.0.2.1 be made the primary destination address would look as follows:

```
+---------------------------------+
|  Type=0xC004   | Length = 16    |
+---------------------------------+
|       C-ID = 0x01023479         |
+---------------------------------+
|  Type=5        | Length = 8     |
+----------------+----------------+
|       Value=0xC0000201          |
+----------------+----------------+
```

Valid Chunk Appearance

The Set Primary IP Address parameter may appear in the ASCONF, the INIT, or the INIT-ACK chunk type. The inclusion of this parameter in the INIT or INIT-ACK can be used to indicate an initial preference of primary address.

4.2.5. Success Indication

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Type = 0xC005          |      Length = 8               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               ASCONF-Response Correlation ID                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

By default if a responding endpoint does not report an error for any requested TLV, a success is implicitly indicated. Thus a sender of a ASCONF-ACK MAY indicate complete success of all TLVs in an ASCONF by returning only the Chunk Type, Chunk Flags, Chunk Length (set to 8) and the Sequence Number.

The responding endpoint MAY also choose to explicitly report a success for a requested TLV, by returning a success report ASCONF Parameter Response.

ASCONF-Response Correlation ID: 32 bits

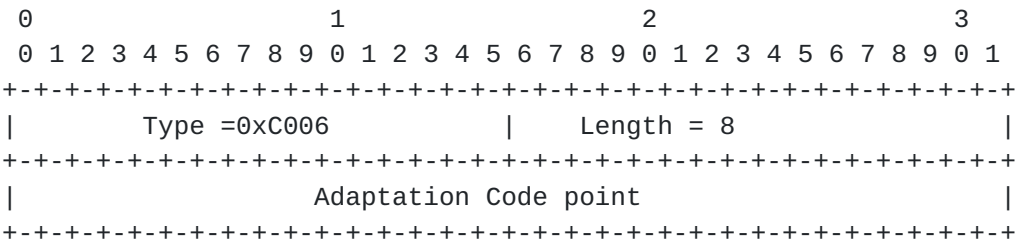
This is an opaque integer assigned by the sender to identify each request parameter. The receiver of the ASCONF Chunk will copy this 32 bit value from the ASCONF-Request Correlation ID into the ASCONF Response Correlation ID field so the peer can easily correlate the request to this response.

Valid Chunk Appearance

The Success Indication parameter may only appear in the ASCONF-ACK

chunk type.

4.2.6. Adaptation Layer Indication



This parameter is specified for the communication of peer upper layer protocols. It is envisioned to be used for flow control and other adaptation layers that require an indication to be carried in the INIT and INIT-ACK. Each adaptation layer that is defined that wishes to use this parameter MUST specify an adaptation code point in an appropriate RFC defining its use and meaning. This parameter SHOULD NOT be examined by the receiving SCTP implementation and should be passed opaquely to the upper layer protocol.

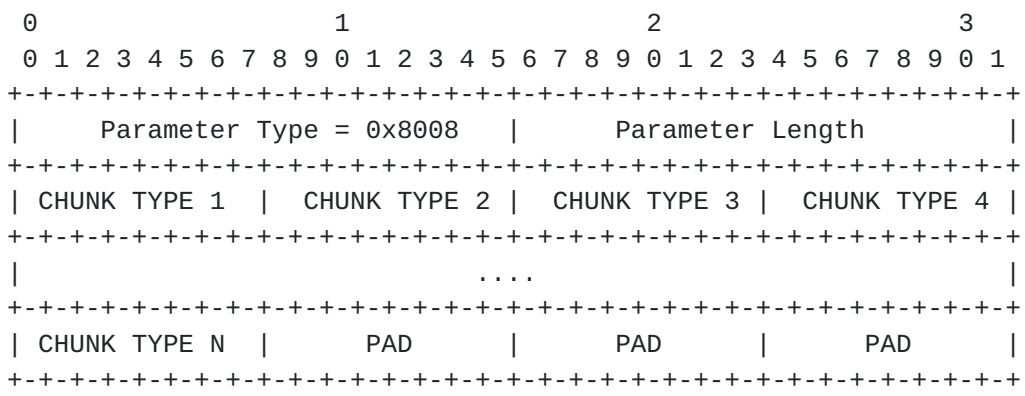
Note: this parameter is not used in either the addition or deletion of addresses but is for the convenience of the upper layer. This document includes this parameter to minimize the number of SCTP documents.

Valid Chunk Appearance

The Adaptation Layer Indication parameter may appear in INIT or INIT-ACK chunk and SHOULD be passed to the receivers upper layer protocol based upon the upper layer protocol configuration of the SCTP stack. This parameter MUST NOT be sent in any other chunks and if it is received in another chunk it MUST be ignored.

4.2.7. Supported Extensions Parameter

This parameter is used at startup to identify any additional extensions that the sender supports. The sender MUST support both the sending and the receiving of any chunk types listed within the Supported Extensions Parameter. An implementation supporting this extension MUST list the ASCONF,the ASCONF-ACK, and the AUTH chunks in its INIT and INIT-ACK parameters.



Parameter Type This field holds the IANA defined parameter type for Supported Extensions Parameter. The suggested value of this field for IANA is 0x8008.

Parameter Type Length This field holds the length of the parameter, including the Parameter Type, Parameter Length and any addition supported extensions. Note: the length MUST NOT include any padding.

CHUNK TYPE X This field(s) hold the chunk type of any SCTP extension(s) that are currently supported by the sending SCTP. Multiple chunk types may be defined listing each additional feature that the sender supports. The sender MUST NOT include multiple Supported Extensions Parameter within any chunk.

Parameter Appearance This parameter may appear in the INIT or INIT-ACK chunk. This parameter MUST NOT appear in any other chunk.

4.3. New Error Causes

Five new Error Causes are added to the SCTP Operational Errors, primarily for use in the ASCONF-ACK Chunk.

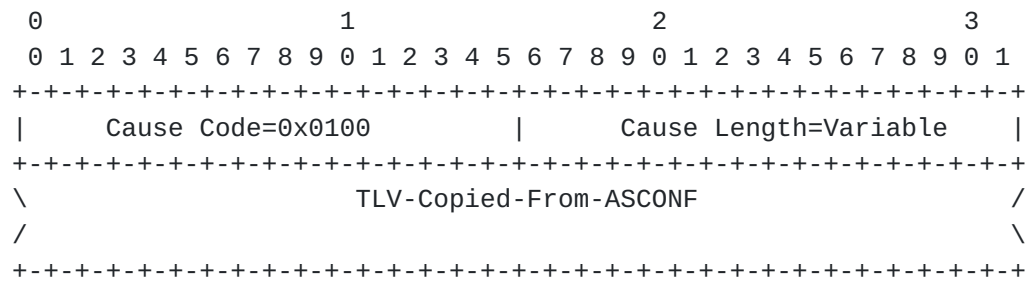
Cause Code Value	Cause Code
0x0100	Request to Delete Last Remaining IP Address.
0x0101	Operation Refused Due to Resource Shortage.
0x0102	Request to Delete Source IP Address.
0x0103	Association Aborted due to illegal ASCONF-ACK.
0x0104	Request refused - no authorization.

Table 5: New Error Causes

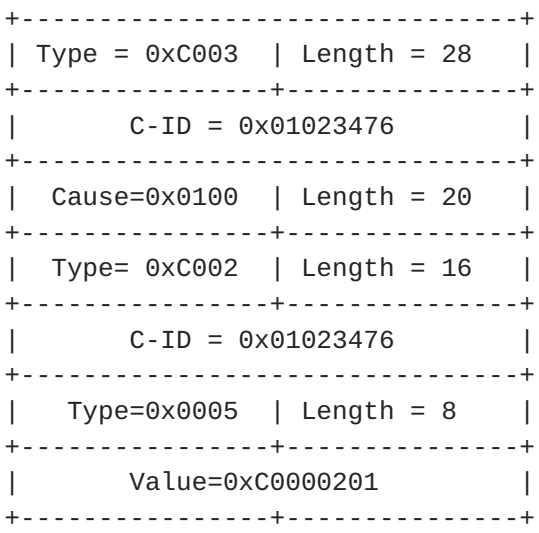
4.3.1. Error Cause: Request to Delete Last Remaining IP Address

Cause of error

Request to Delete Last Remaining IP address: The receiver of this error sent a request to delete the last IP address from its association with its peer. This error indicates that the request is rejected.



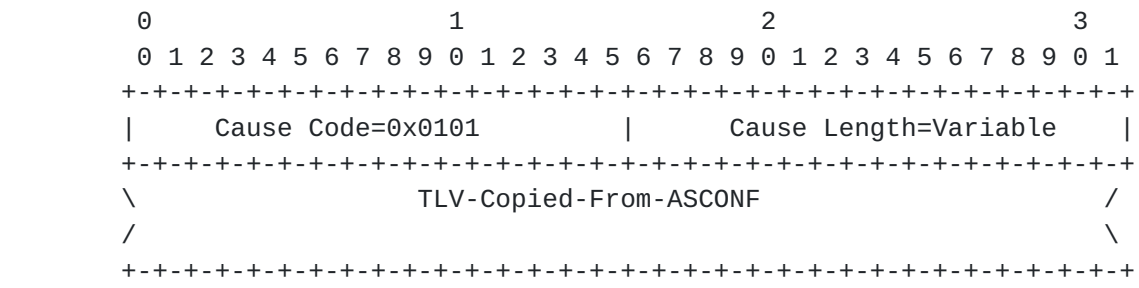
An example of a failed delete in an Error Cause TLV would look as follows in the response ASCONF-ACK message:



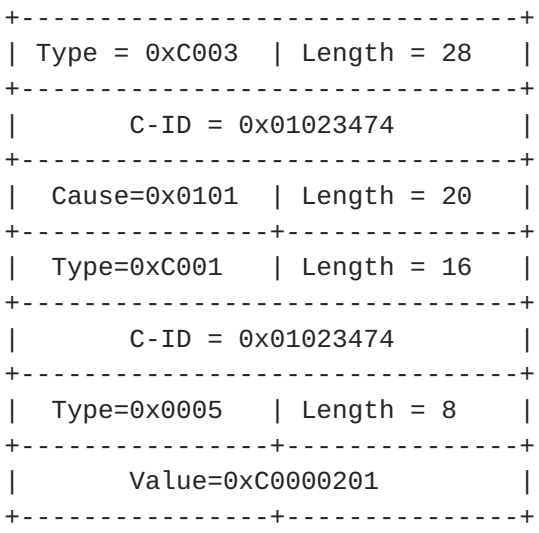
4.3.2. Error Cause: Operation Refused Due to Resource Shortage

Cause of error

This error cause is used to report a failure by the receiver to perform the requested operation due to a lack of resources. The entire TLV that is refused is copied from the ASCONF into the error cause.



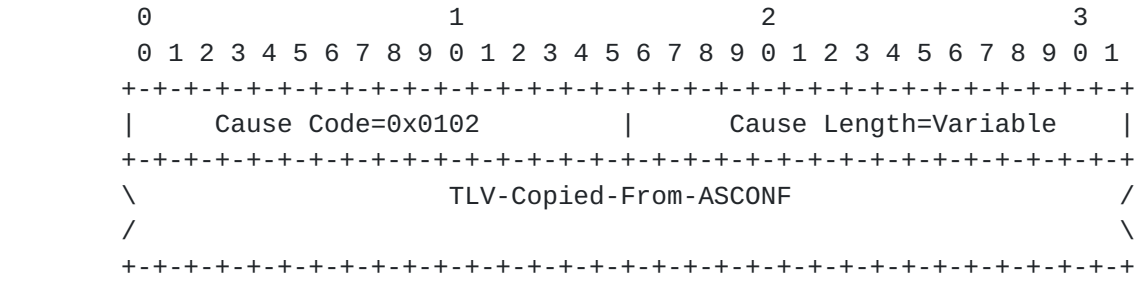
An example of a failed addition in an Error Cause TLV would look as follows in the response ASCONF-ACK message:



4.3.3. Error Cause: Request to Delete Source IP Address

Cause of error

Request to Delete Source IP Address: The receiver of this error sent a request to delete the source IP address of the ASCONF message. This error indicates that the request is rejected.



An example of a failed delete in an Error Cause TLV would look as follows in the response ASCONF-ACK message:

```
+---------------------------------+
| Type = 0xC003  | Length = 28    |
+---------------------------------+
|       C-ID = 0x01023476         |
+---------------------------------+
|  Cause=0x0102  | Length = 20    |
+----------------+----------------+
|  Type=0xC002   | Length = 16    |
+----------------+----------------+
|       C-ID = 0x01023476         |
+---------------------------------+
|   Type=0x0005  | Length = 8     |
+----------------+----------------+
|       Value=0xC0000201          |
+----------------+----------------+
```

IMPLEMENTATION NOTE: It is unlikely that an endpoint would source a packet from the address being deleted, unless the endpoint does not do proper source address selection.

4.3.4. Error Cause: Association Aborted due to illegal ASCONF-ACK

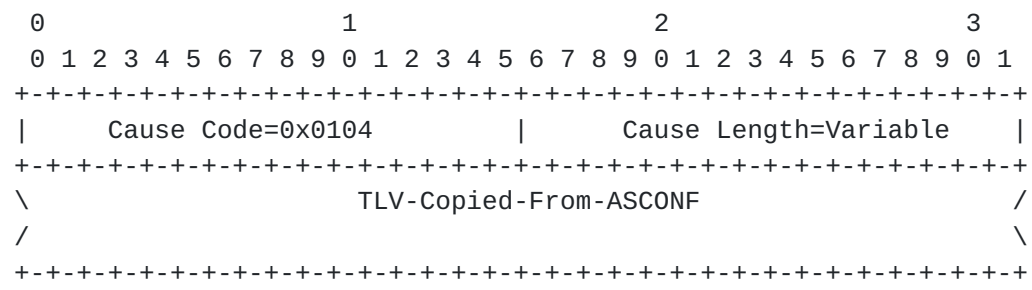
This error is to be included in an ABORT that is generated due to the reception of an ASCONF-ACK that was not expected but is larger than the current sequence number (see Section 5.3 Rule F0). Note: that a sequence number is larger than the last ACKed sequence number if it is either the next sequence or no more than 2**31-1 greater than the current sequence number. Sequence numbers smaller than the last acked sequence number are silently ignored.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Cause Code=0x0103         |      Cause Length=4           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.3.5. Error Cause: Request refused - no authorization.

Cause of error

This error cause may be included to reject a request based on local security policies.



5. Procedures

This section will lay out the specific procedures for address configuration change chunk type and its processing.

5.1. ASCONF Chunk Procedures

When an endpoint has an ASCONF signaled change to be sent to the remote endpoint it MUST do the following:

A1) Create an ASCONF Chunk as defined in Section 4.1.1. The chunk MUST contain all of the TLV(s) of information necessary to be sent to the remote endpoint, and unique correlation identities for each request.

A2) A sequence number MUST be assigned to the Chunk. The sequence number MUST be larger by one. The sequence number MUST be initialized at the start of the association to the same value as the Initial TSN and every time a new ASCONF Chunk is created it MUST be incremented by one after assigning the sequence number to the newly created chunk .

A3) If no SCTP packet with one or more ASCONF Chunk(s) is outstanding (un-acknowledged) with the remote peer, send the chunk and proceed to step A4. If an ASCONF chunk is outstanding, then the ASCONF chunk should be queued for later transmission and no further action should be taken until the previous ASCONF is acknowledged or a time out occurs.

A4) The sender MUST Start a T-4 RTO timer, using the RTO value of the selected destination address (normally the primary path; see [I-D.ietf-tsvwg-2960bis] section 6.4 for details).

A5) When the ASCONF-ACK that acknowledges the sequence number last sent arrives, the sender MUST stop the T-4 RTO timer, and clear the appropriate association and destination error counters as defined in [I-D.ietf-tsvwg-2960bis] section 8.1 and 8.2.

A6) The endpoint MUST process all of the TLVs within the ASCONF-ACK(s) to find out particular status information returned to the various requests that were sent. Use the Correlation IDs to correlate the request and the responses.

A7) If an error response is received for a TLV parameter, all TLVs with no response before the failed TLV are considered successful if not reported. All TLVs after the failed response are considered unsuccessful unless a specific success indication is present for the parameter.

A8) If there is no response(s) to specific TLV parameter(s), and no failures are indicated, then all request(s) are considered successful.

A9) If the peer responds to an ASCONF with an ERROR chunk reporting that it did not recognize the ASCONF Chunk type, the sender of the ASCONF MUST NOT send any further ASCONF Chunks and MUST stop its T-4 timer.

If the T-4 RTO timer expires the endpoint MUST do the following:

B1) Increment the error counters and perform path failure detection on the appropriate destination address as defined in [I-D.ietf-tsvwg-2960bis] section 8.1 and 8.2.

B2) Increment the association error counters and perform endpoint failure detection on the association as defined in [I-D.ietf-tsvwg-2960bis] section 8.1 and 8.2.

B3) Back-off the destination address RTO value to which the ASCONF chunk was sent by doubling the RTO timer value.

Note: The RTO value is used in the setting of all timer types for SCTP. Each destination address has a single RTO estimate.

B4) Re-transmit the ASCONF Chunk last sent and if possible choose an alternate destination address (please refer to [I-D.ietf-tsvwg-2960bis] section 6.4.1). An endpoint MUST NOT add new parameters to this chunk, it MUST be the same (including its sequence number) as the last ASCONF sent. An endpoint MAY, however, bundle an additional ASCONF with new ASCONF parameters with the next sequence number. For details see Section 5.5

B5) Restart the T-4 RTO timer. Note: that if a different destination is selected, then the RTO used will be that of the new destination address.

Note: the total number of re-transmissions is limited by B2 above. If the maximum is reached, the association will fail and enter into the CLOSED state (see [I-D.ietf-tsvwg-2960bis] section 6.4.1 for details).

5.1.1. Congestion Control of ASCONF Chunks

In defining the ASCONF Chunk transfer procedures, it is essential that these transfers MUST NOT cause congestion within the network. To achieve this, we place these restrictions on the transfer of ASCONF Chunks:

C1) One and only one SCTP packet holding ASCONF Chunk(s) MAY be in transit and unacknowledged at any one time. If a sender, after sending an ASCONF chunk, decides it needs to transfer another ASCONF Chunk, it MUST wait until the ASCONF-ACK Chunk returns from the previous ASCONF Chunk before sending a subsequent ASCONF. Note: this restriction binds each side, so at any time two ASCONF may be in-transit on any given association (one sent from each endpoint). However when an ASCONF Chunk is retransmitted due to a time-out, the additional held ASCONF Chunks can be bundled into the retransmission packet as described in Section 5.5.

C2) An ASCONF Chunk may be bundled with any other chunk type including other ASCONF Chunks. If bundled with other ASCONF Chunks, the chunks MUST appear in sequential order with respect to their Sequence Number.

C3) An ASCONF-ACK Chunk may be bundled with any other chunk type including other ASCONF-ACK Chunks. If bundled with other ASCONF-ACK Chunks, the chunks MUST appear in sequential order with respect to their Sequence Number.

C4) Both ASCONF and ASCONF-ACK Chunks MUST NOT be sent in any SCTP state except ESTABLISHED, SHUTDOWN-PENDING, SHUTDOWN-RECEIVED and SHUTDOWN-SENT.

C5) An ASCONF Chunk and an ASCONF-ACK Chunk SHOULD not be larger than the PMTU. If the PMTU is unknown, then the PMTU should be set to the minimum PMTU. The minimum PMTU depends on the IP version used for transmission, and is the lesser of 576 octets and the first-hop MTU for IPv4 [RFC1122] and 1280 octets for IPv6 [RFC2460].

An ASCONF sender without these restrictions could possibly flood the network with a large number of separate address change operations thus causing network congestion.

If the sender of an ASCONF Chunk receives an Operational Error indicating that the ASCONF Chunk type is not understood, then the sender MUST NOT send subsequent ASCONF Chunks to the peer. The endpoint should also inform the upper layer application that the peer endpoint does not support any of the extensions detailed in this document.

5.2. Upon reception of an ASCONF Chunk.

When an endpoint receives an ASCONF Chunk from the remote peer special procedures may be needed to identify the association the ASCONF Chunk is associated with. To properly find the association the following procedures SHOULD be followed:

D1) Use the source address and port number of the sender to attempt to identify the association (i.e., use the same method defined in [I-D.ietf-tsvwg-2960bis] used for all other SCTP Chunks). If found proceed to rule D4.

D2) If the association is not found, use the address found in the Address Parameter TLV combined with the port number found in the SCTP common header. If found proceed to rule D4.

D2-ext) If more than one ASCONF Chunks are packed together, use the address found in the ASCONF Address Parameter TLV of each of the subsequent ASCONF Chunks. If found, proceed to rule D4.

D3) If neither D1, D2 nor D2-ext locates the association, treat the chunk as an Out Of The Blue packet as defined in [I-D.ietf-tsvwg-2960bis].

D4) Follow the normal rules to validate the SCTP verification tag found in [I-D.ietf-tsvwg-2960bis].

D5) After the verification tag has been validated, normal chunk processing should occur. Prior to finding the ASCONF chunk the receiver MUST encounter an AUTH chunk as described in [I-D.ietf-tsvwg-sctp-auth]. If either authentication fails, or the AUTH chunk is missing, the receiver MUST silently discard this chunk and the rest of the packet.

After identification and verification of the association, the following should be performed to properly process the ASCONF Chunk:

E1)

If the value found in the sequence number of the ASCONF Chunk is equal to the ('Peer-Sequence-Number' + 1) and the Sequence Number of the ASCONF Chunk is the first in the SCTP Packet, the endpoint MAY clean any old cached ASCONF-ACK up to the 'Peer-Sequence-Number' and then proceed to rule E4.

E1-ext If the value found in the sequence number of the ASCONF Chunk is equal to the ('Peer-Sequence-Number' + 1) and the ASCONF chunk is NOT the first Sequence Number in the SCTP packet proceed to rule E4 but do NOT clear any cached ASCONF-ACK or state information.

E2)

If the value found in the sequence number is less than the ('Peer-Sequence-Number' + 1), simply skip to the next ASCONF, and include in the outbound response packet any previously cached ASCONF-ACK response that was sent and saved that matches the sequence number of the ASCONF. Note: it is possible that no cached ASCONF-ACK Chunk exists. This will occur when an older ASCONF arrives out of order. In such a case the receiver should skip the ASCONF Chunk and not include ASCONF-ACK Chunk for that chunk.

E3)

Then, process each ASCONF one by one as above while the Sequence Number of the ASCONF is less than the ('Peer-Sequence-Number' + 1).

E4) When the sequence number matches the next one expected, process the ASCONF as described below and after processing the ASCONF Chunk, append an ASCONF-ACK Chunk to the response packet and cache a copy of it (in the event it later needs to be retransmitted).

V1) Process the TLVs contained within the Chunk performing the appropriate actions as indicated by each TLV type. The TLVs MUST be processed in order within the Chunk. For example, if the sender puts 3 TLVs in one chunk, the first TLV (the one closest to the Chunk Header) in the Chunk MUST be processed first. The next TLV in the chunk (the middle one) MUST be processed second and finally the last TLV in the Chunk MUST be processed last.

V2) In processing the chunk, the receiver should build a response message with the appropriate error TLVs, as specified in the Parameter type bits for any ASCONF Parameter it does not understand. To indicate an unrecognized parameter, cause type 8 as defined in the ERROR in 3.3.10.8 of [I-D.ietf-tsvwg-2960bis] should be used. The endpoint may also use the response to carry rejections for other reasons such as resource shortages etc, using the Error Cause TLV and an appropriate error condition.

Note: a positive response is implied if no error is indicated by the sender.

V3)

All responses MUST copy the ASCONF-Request Correlation ID field received in the ASCONF parameter, from the TLV being responded to, into the ASCONF-Request Correlation ID field in the response parameter.

V4) After processing the entire Chunk, the receiver of the ASCONF MUST queue the response ASCONF-ACK Chunk for transmission after the rest of the SCTP packet has been processed. This allows the ASCONF-ACK Chunk to be bundled with other ASCONF-ACK Chunks as well as any additional responses e.g. a SACK Chunk.

V5) Update the 'Peer-Sequence-Number' to the value found in the sequence number field.

E5) Otherwise, the ASCONF Chunk is discarded since it must be either a stale packet or from an attacker. A receiver of such a packet MAY log the event for security purposes.

E6) When all ASCONF Chunks are processed for this SCTP packet, send back the accumulated single response packet with all of the ASCONF-ACK Chunks. The destination address of the SCTP packet containing the ASCONF-ACK Chunks MUST be the source address of the SCTP packet that held the ASCONF Chunks.

E7) While processing the ASCONF Chunks in the SCTP packet, if the response packet will exceed the PMTU of the return path, the receiver MUST stop adding additional ASCONF-ACKs into the response packet but MUST continue to process all of the ASCONF Chunks, saving ASCONF-ACK Chunk responses in its cached copy. The sender of the ASCONF Chunk will later retransmit the ASCONF Chunks that were not responded to, at which time the cached copies of the responses that would NOT fit in the PMTU can be sent to the peer.

Note: These rules have been presented with the assumption that the implementation is caching old ASCONF-ACKs in case of loss of SCTP packets in the ACK path. It is allowable for an implementation to maintain this state in another form it deems appropriate, as long as that form results in the same ASCONF-ACK sequences being returned to the peer as outlined above.

5.3. General rules for address manipulation

When building TLV parameters for the ASCONF Chunk that will add or delete IP addresses the following rules MUST be applied:

F0) If an endpoint receives an ASCONF-ACK that is greater than or equal to the next sequence number to be used but no ASCONF Chunk is outstanding the endpoint MUST ABORT the association. Note: that a sequence number is greater than if it is no more than 2^^31-1 larger than the current sequence number (using serial arithmetic).

F1) When adding an IP address to an association, the IP address is NOT considered fully added to the association until the ASCONF-ACK arrives. This means that until such time as the ASCONF containing the add is acknowledged the sender MUST NOT use the new IP address as a source for ANY SCTP packet except on carrying an ASCONF Chunk. The receiver of the add IP address request may use the address as a destination immediately. The receiver MUST use the path verification procedure for the added address before using that address. The receiver MUST NOT send packets to the new address except for the corresponding ASCONF-ACK Chunk or HEARTBEAT Chunks for path verification before the new path is verified. If the ASCONF-ACK is sent to the new address it MAY be bundled with the HEARTBEAT chunk for path verification.

F2) After the ASCONF-ACK of an IP address add arrives, the endpoint MAY begin using the added IP address as a source address for any type of SCTP chunk.

F3a) If an endpoint receives an Error Cause TLV indicating that the IP address Add or IP address Deletion parameters was not understood, the endpoint MUST consider the operation failed and MUST NOT attempt to send any subsequent Add or Delete requests to the peer.

F3b) If an endpoint receives an Error Cause TLV indicating that the IP address Set Primary IP Address parameter was not understood, the endpoint MUST consider the operation failed and MUST NOT attempt to send any subsequent Set Primary IP Address requests to the peer.

F4) When deleting an IP address from an association, the IP address MUST be considered a valid destination address for the reception of SCTP packets until the ASCONF-ACK arrives and MUST NOT be used as a source address for any subsequent packets. This means that any datagrams that arrive before the ASCONF-ACK destined to the IP address being deleted MUST be considered part of the current association. One special consideration is that ABORT Chunks arriving destined to the IP address being deleted MUST be ignored (see Section 5.3.1 for further details).

F5) An endpoint MUST NOT delete its last remaining IP address from an association. In other words if an endpoint is NOT multi-homed it MUST NOT use the delete IP address without an add IP address preceding the delete parameter in the ASCONF Chunk. Or if an endpoint sends multiple requests to delete IP addresses it MUST NOT delete all of the IP addresses that the peer has listed for the requester.

F6) An endpoint MUST NOT set an IP header source address for an SCTP packet holding the ASCONF Chunk to be the same as an address being deleted by the ASCONF Chunk.

F7) If a request is received to delete the last remaining IP address of a peer endpoint, the receiver MUST send an Error Cause TLV with the error cause set to the new error code 'Request to Delete Last Remaining IP Address'. The requested delete MUST NOT be performed or acted upon, other than to send the ASCONF-ACK.

F8) If a request is received to delete an IP address which is also the source address of the IP packet which contained the ASCONF chunk, the receiver MUST reject this request. To reject the request the receiver MUST send an Error Cause TLV set to the new error code 'Request to Delete Source IP Address' (unless Rule F5 has also been violated, in which case the error code 'Request to Delete Last Remaining IP Address' is sent).

F9) If an endpoint receives an ADD IP address request and does not have the local resources to add this new address to the association, it MUST return an Error Cause TLV set to the new error code 'Operation Refused Due to Resource Shortage'.

F10) If an endpoint receives an 'Out of Resource' error in response to its request to ADD an IP address to an association, it must either ABORT the association or not consider the address part of the association. In other words if the endpoint does not ABORT the association, it must consider the add attempt failed and NOT use this address since its peer will treat SCTP packets destined to the address as Out Of The Blue packets.

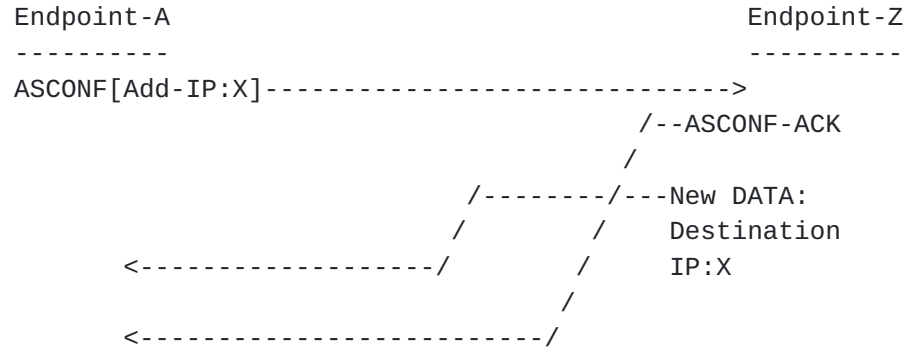
F11) When an endpoint receiving an ASCONF to add an IP address sends an 'Out of Resource' in its response, it MUST also fail any subsequent add or delete requests bundled in the ASCONF. The receiver MUST NOT reject an ADD and then accept a subsequent DELETE of an IP address in the same ASCONF Chunk. In other words, once a receiver begins failing any ADD or DELETE request, it must fail all subsequent ADD or DELETE requests contained in that single ASCONF.

F12) When an endpoint receives a request to delete an IP address that is the current primary address, it is an implementation decision as to how that endpoint chooses the new primary address.

F13) When an endpoint receives a valid request to DELETE an IP address the endpoint MUST consider the address no longer as part of the association. It MUST NOT send SCTP packets for the association to that address and it MUST treat subsequent packets received from that address as Out Of The Blue.

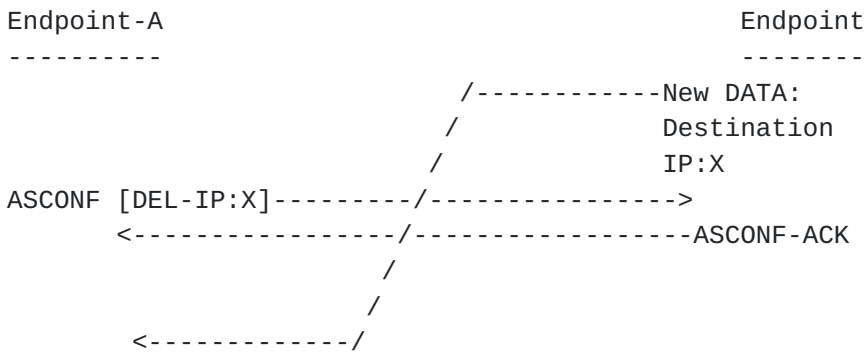
During the time interval between sending out the ASCONF and receiving the ASCONF-ACK it MAY be possible to receive DATA Chunks out of order. The following examples illustrate these problems:

F14) All addresses added by the reception of an ASCONF chunk MUST be put into the unconfirmed state and MUST have path verification performed on them before the address can be used as described in [I-D.ietf-tsvwg-2960bis] section 5.4.



In the above example we see a new IP address (X) being added to the Endpoint-A. However due to packet re-ordering in the network a new DATA chunk is sent and arrives at Endpoint-A before the ASCONF-ACK confirming the add of the address to the association.

A similar problem exists with the deletion of an IP address as follows:

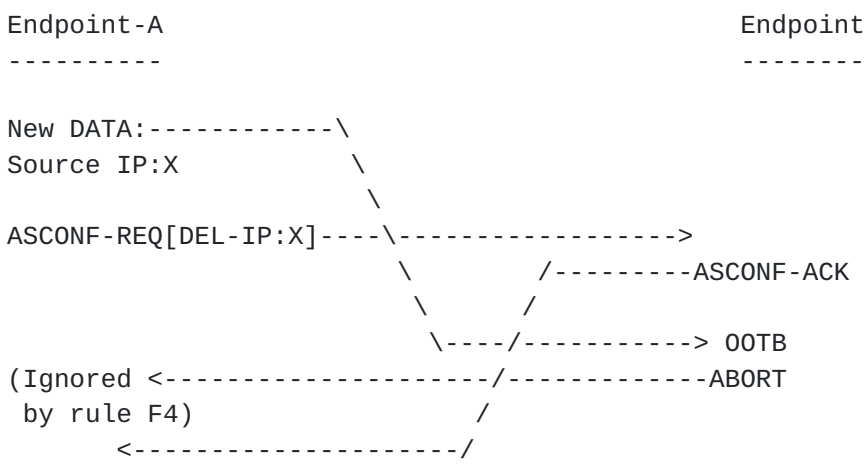


In this example we see a DATA chunk destined to the IP:X (which is about to be deleted) arriving after the deletion is complete. For the ADD case an endpoint SHOULD consider the newly adding IP address valid for the association to receive data from during the interval when awaiting the ASCONF-ACK. The endpoint MUST NOT source data from this new address until the ASCONF-ACK arrives but it may receive out of order data as illustrated and MUST NOT treat this data as an OOTB datagram (please see [I-D.ietf-tsvwg-2960bis] section 8.4). It MAY drop the data silently or it MAY consider it part of the association but it MUST NOT respond with an ABORT.

For the DELETE case, an endpoint MAY respond to the late arriving DATA packet as an OOTB datagram or it MAY hold the deleting IP address for a small period of time as still valid. If it treats the DATA packet as an OOTB the peer will silently discard the ABORT (since by the time the ABORT is sent the peer will have removed the IP address from this association). If the endpoint elects to hold the IP address valid for a period of time, it MUST NOT hold it valid longer than 2 RTO intervals for the destination being removed.

5.3.1. A special case for OOTB ABORT Chunks

Another case worth mentioning is illustrated below:



For this case, during the deletion of an IP address, an Abort MUST be ignored if the destination address of the Abort message is that of a destination being deleted.

5.3.2. A special case for changing an address.

In some instances the sender may only have one IP address in an association that is being renumbered. When this occurs, the sender may not be able to send to the peer the appropriate ADD/DELETE pair and use the old address as a source in the IP header. For this reason the sender MUST fill in the Address Parameter field with an address that is part of the association (in this case the one being deleted). This will allow the receiver to locate the association without using the source address found in the IP header.

The receiver of such a chunk MUST always first use the source address found in the IP header in looking up the association. The receiver should attempt to use the address found in the Address Parameter field only if the lookup fails using the source address from the IP header. The receiver MUST reply to the source address of the packet in this case which is the new address that was added by the ASCONF (since the old address is no longer a part of the association after processing).

5.4. Setting of the primary address

A sender of this option MAY elect to send this combined with a deletion or addition of an address. A sender MUST only send a set primary request to an address that is already considered part of the association. In other words if a sender combines a set primary with an add of a new IP address the set primary will be discarded unless the add request is to be processed BEFORE the set primary (i.e., it precedes the set primary).

A request to set primary MAY also appear in an INIT or INIT-ACK chunk. This can give advice to the peer endpoint as to which of its addresses the sender of the INIT or INIT-ACK would prefer to be used as the primary address.

The request to set an address as the primary path is an option the receiver SHOULD perform. It is considered advice to the receiver of the best destination address to use in sending SCTP packets (in the requester's view). If a request arrives that asks the receiver to set an address as primary that does not exist, the receiver SHOULD NOT honor the request, leaving its existing primary address unchanged.

5.5. Bundling of multiple ASCONFs

In the normal case a single ASCONF is sent in a packet and a single reply ASCONF-ACK is received. However, in the event of the loss of an SCTP packet containing either an ASCONF or ASCONF-ACK it is allowable for a sender to bundle additional ASCONFs in the retransmission. In bundling multiple ASCONFs the following rules MUST be followed:

1. Previously transmitted ASCONF Chunks MUST be left unchanged.
2. Each SCTP packet containing ASCONF Chunks MUST be bundled starting with the smallest ASCONF Sequence Number first in the packet (closest to the Chunk header) and preceding in sequential order from lowest to highest ASCONF Sequence Number.
3. All ASCONFs within the packet MUST be adjacent to each other i.e., no other chunk type must separate the ASCONFs.
4. Each new ASCONF lookup address MUST be populated as if the previous ASCONFs had been processed and accepted.

6. Security Considerations

The addition and or deletion of an IP address to an existing association does provide an additional mechanism by which existing associations can be hijacked. Therefore this document requires the use of the authentication mechanism defined in [I-D.ietf-tsvwg-sctp-auth] to limit the ability of an attacker to hijack an association.

Hijacking an association by using the addition and deletion of an IP address is only possible for an attacker who is able to intercept the initial two packets of the association setup when the SCTP-AUTH extension is used without pre-shared keys. If such a threat is considered a possibility, then the [I-D.ietf-tsvwg-sctp-auth] extension MUST be used with a preconfigured shared end-point pair key to mitigate this threat. For a more detailed analysis see

[I-D.ietf-tsvwg-sctp-auth].

When the address parameter in ASCONF chunks with Add, IP Delete IP, or Set Primary IP parameters is a wildcard, the source address of the packet is used. This address is not protected by SCTP-AUTH [I-D.ietf-tsvwg-sctp-auth] and an attacker can therefore intercept such a packet and modify the source address. Even if the source address is not one presently an alternate for the association, the identification of the association may rely on the other information in the packet (perhaps the verification tag, for example). An on-path attacker can therefore modify the source address to its liking.

If the ASCONF includes an Add IP with a wildcard address, the attacker can add an address of its liking, which provides little immediate damage but can set up later attacks.

If the ASCONF includes a Delete IP with a wildcard address, the attacker can cause all addresses but one of its choosing to be deleted from an association. The address supplied by the attacker must already belong to the association, which makes this more difficult for the attacker. However, the sole remaining address might be one that the attacker controls, for example, or can monitor, etc. The least result is the sender and the deceived receiver would have different ideas of what that sole remaining address would be. This will eventually cause the association to fail, but in the meantime, the deceived receiver could be transmitting packets to an address the sender did not intend.

If the ASCONF includes a Set Primary IP with a wildcard address, then the attacker can cause an address to be used as a primary address. This is limited to an address that already belongs to the association, so the damage is limited. At least, the result would be that the recipient is using a primary address that the sender did not intend. However, if both a wildcard Add IP and a wildcard Set Primary IP are used, then the attacker can modify the source address to both add an address to its liking to the association and make it the primary address. Such a combination would present the attacker with opportunity for more damage.

Note that all these attacks are from an on-path attacker. Endpoints that believe they face a threat from on-path attackers SHOULD NOT use wildcard addresses in ASCONF Add IP, Delete IP or Set Primary IP parameters.

If an SCTP endpoint that supports this extension receives an INIT that indicates that the peer supports the ASCONF extension but does NOT support the [I-D.ietf-tsvwg-sctp-auth] extension, the receiver of such an INIT MUST send an ABORT in response to such an INIT. Note:

that an implementation is allowed to silently discard such an INIT as an option as well but under NO circumstance is an implementation allowed to proceed with the association setup by sending an INIT-ACK in response.

An implementation that receives an INIT-ACK that indicates that the peer does not support the [I-D.ietf-tsvwg-sctp-auth] extension MUST NOT send the COOKIE-ECHO to establish the association. Instead the implementation MUST discard the INIT-ACK and report to the upper layer user that an association cannot be established destroying the TCB.

Other types of attacks, e.g. bombing, are discussed in detail in [I-D.ietf-tsvwg-sctpthreat]. The bombing attack, in particular, is countered by the use of a random nonce and is required by [I-D.ietf-tsvwg-2960bis].

An on-path attacker can modify the INIT and INIT-ACK Supported Extensions parameter (and authentication related parameters) to produce a denial of service. If the on-path attacker removes the [I-D.ietf-tsvwg-sctp-auth] related parameters from an INIT that indicates it supports the ASCONF extension, the association will not be established. If the on-path attacker adds a Supported Extensions parameter mentioning the ASCONF type to an INIT or INIT-ACK that does not carry any AUTH related parameters, the association will not be established. If the on-path attacker removes the Supported Extensions parameter (or removes the ASCONF type from that parameter) from the INIT or the INIT-ACK, then the association will not be able to use the ADD-IP feature. If the on-path attacker adds the Supported Extensions parameter listing the ASCONF type to an INIT-ACK that did not carry one (but did carry AUTH related parameters), then the INIT sender may use ASCONF where the INIT-ACK sender does not support it. This would be discovered later if the INIT sender transmitted an ASCONF, but the INIT sender could have made configuration choices at that point. As the INIT and INIT-ACK are not protected by the AUTH feature, there is no way to counter such attacks. Note however that an on-path attacker capable of modifying the INIT and INIT-ACK would almost certainly also be able to prevent the INIT and INIT-ACK from being delivered or modify the verification tags or checksum to cause the packet to be discarded, so the Supported Extensions adds little additional vulnerability (with respect to preventing association formation) to the SCTP protocol. The ability to prevent the use of this new feature is an additional vulnerability to SCTP but only for this new feature.

The Adaptation Layer Indication is subject to corruption, insertion or deletion from the INIT and INIT-ACK chunks by an on-path attacker. This parameter SHOULD be opaque to the SCTP protocol (see section

4.2.6), and so changes to the parameter will likely not affect the SCTP protocol. However, any adaptation layer that is defined SHOULD consider its own vulnerabilities in the security considerations section of the RFC that defines its adaptation code point.

The Set Primary IP Address parameter is subject to corruption, insertion or deletion by an on-path attacker when included in the INIT and INIT-ACK chunks. The attacker could use this to influence the receiver to choose an address to its own purposes (one over which it has control, one that would be less desirable for the sender, etc.). An on-path attacker would also have the ability to include or remove addresses for the association from the INIT or INIT-ACK, so it is not limited in the address it can specify in the Set Primary IP Address. Endpoints that wish to avoid this possible threat MAY defer sending the initial Set Primary request and wait until the association is fully established before sending a fully protected ASCONF with the Set Primary as its single parameter.

7. IANA considerations

This document defines the following new SCTP parameters, chunks and errors (http://www.iana.org/assignments/sctp-parameters):

- Two new chunk types,
- Six parameter types, and
- Five new SCTP error causes.

One of the two new chunk types must come from the range of chunk types where the upper two bits are one, we recommend 0xC1 but any other available code point with the upper bits set is also acceptable. The second chunk type must come from the range where only the upper bit is set to one. We recommend 0x80 but any other available code point with the upper bit set is also acceptable. The chunk types with there suggested values are shown below.

Chunk Type	Chunk Name	
0xC1	Address Configuration Change Chunk	(ASCONF)
0x80	Address Configuration Acknowledgment	(ASCONF-ACK)

All of the parameter types, with the exception of the supported parameters extension, must come from the range of types where the upper two bits are set, we recommend 0xC001 - 0xC006, as shown below. The supported parameters type extension must come from the range where only the upper bit is set, we recommend 0x8008. Note: that for

any of these values a different unique parameter type may be assigned by IANA as long as the upper bits correspond to the ones specified in this document. The suggested parameter types are listed below:

Parameter Type	Parameter Name
0x8008	Supported Extensions
0xC001	Add IP Address
0xC002	Delete IP Address
0xC003	Error Cause Indication
0xC004	Set Primary Address
0xC005	Success Indication
0xC006	Adaptation Layer Indication

The five new error causes can be any value, in this document we have used 0x0100-0x0104 in an attempt to separate these from the common ranges of error codes. Any other unassigned values are also acceptable. The suggested error causes are listed below:.

Cause Code Value	Cause Code
0x0100	Request to Delete Last Remaining IP Address.
0x0101	Operation Refused Due to Resource Shortage.
0x0102	Request to Delete Source IP Address.
0x0103	Association Aborted due to illegal ASCONF-ACK
0x0104	Request refused - no authorization.

This document also defines an Adaptation code point. The adaptation code point is a 32 bit integer that is assigned by IANA through an IETF Consensus action as defined in [RFC2434]. For this new registry no initial values are being added by this document, however draft-ietf-rddp-sctp will add the first entry.

8. Acknowledgments

The authors would like to express a special note of thanks to Michael Ramahlo and Phillip Conrad for their extreme efforts in the early formation of this draft.

The authors wish to thank Jon Berger, Mark Butler, Lars Eggert, Janardhan Iyengar, Greg Kendall, Seok Koh, Salvatore Loreto, Peter Lei, John Loughney, Sandy Murphy, Ivan Arias Rodriguez, Renee Revis,

Marshall Rose, Ronnie Sellars, Chip Sharp, and Irene Ruengeler for their invaluable comments.

The authors would also like to give special mention to Maria-Carmen Belinchon and Ian Rytina for there early contributions to this document and their thoughtful comments.

And a special thanks to James Polk, abstract writer to the few but lucky.

9. References

9.1. Normative References

[RFC1122] Braden, R., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.

[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, August 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[I-D.ietf-tsvwg-2960bis] Stewart, R., "Stream Control Transmission Protocol", draft-ietf-tsvwg-2960bis-05 (work in progress), June 2007.

[I-D.ietf-tsvwg-sctp-auth] Tuexen, M., "Authenticated Chunks for Stream Control Transmission Protocol (SCTP)", draft-ietf-tsvwg-sctp-auth-08 (work in progress), February 2007.

9.2. Informative References

[I-D.ietf-tsvwg-sctpthreat] Stewart, R., "Security Attacks Found Against SCTP and Current Countermeasures", draft-ietf-tsvwg-sctpthreat-05 (work in progress), June 2007.

Appendix A. Abstract Address Handling

A.1. General remarks

This appendix is non-normative. It is present to give the reader a concise mathematical definition of an SCTP endpoint. The following text provides a working definition of the endpoint notion to discuss address reconfiguration. It is not intended to restrict implementations in any way, its goal is to provide a set of definitions only. Using these definitions should make a discussion about address issues easier.

A.2. Generalized endpoints

A generalized endpoint is a pair of a set of IP addresses and a port number at any given point of time. The precise definition is as follows:

A generalized endpoint gE at time t is given by

gE(t) = ({IP1, ..., IPn}, Port)

where {IP1, ..., IPn} is a non empty set of IP addresses.

Please note that the dynamic addition and deletion of IP-addresses described in this document allows the set of IP-addresses of a generalized endpoint to be changed at some point of time. The port number can never be changed.

The set of IP addresses of a generalized endpoint gE at a time t is defined as

Addr(gE)(t) = {IP1, ..., IPn}

if gE(t) = ({IP1, ..., IPn}, Port) holds at time t.

The port number of a generalized endpoint gE is defined as

Port(gE) = Port

if gE(t) = ({IP1, ..., IPn}, Port) holds at time t.

There is one fundamental rule which restricts all generalized endpoints:

For two different generalized endpoints gE' and gE'' with the same port number Port(gE') = Port(gE'') the address sets Addr(gE')(t) and Addr(gE'')(t) must be disjoint at every point of time.

A.3. Associations

Associations consists of two generalized endpoints and the two address sets known by the peer at any time. The precise definition is as follows:

An association A between to different generalized endpoints gE' and gE'' is given by

A = (gE', S', gE'', S'')

where S'(t) and S''(t) are set of addresses at any time t such that S'(t) is a non-empty subset of Addr(gE')(t) and S''(t) is a non-empty subset of Addr(gE'')(t).

If A = (gE', S', gE'', S'') is an association between the generalized endpoints gE' and gE'' the following notion is used:

Addr(A, gE') = S' and Addr(A, gE'') = S''.

If the dependency on time is important the notion Addr(A, gE')(t) = S'(t) will be used.

If A is an association between gE' and gE'' then Addr(A, gE') is the subset of IP addresses of gE' which is known by gE'' and used by gE'.

Association establishment between gE' and gE'' can be seen as:

1. gE' and gE'' do exist before the association.
2. If an INIT has to be send from gE' to gE'' address scoping rules and other limitations are applied to calculate the subset S' from Addr(gE'). The addresses of S' are included in the INIT chunk.
3. If an INIT-ACK has to be send from gE'' to gE' address scoping rules and other limitations are applied to calculate the subset S'' from Addr(gE''). The addresses of S'' are included in the INIT-ACK chunk.
4. After the handshake the association A = (gE', S', gE'', S'') has been established.
5. Right after the association establishment Addr(A, gE') and Addr(A, gE'') are the addresses which have been seen on the wire during the handshake.

A.4. Relationship with RFC 4960

[I-D.ietf-tsvwg-2960bis] defines the notion of an endpoint. This subsection will show that these endpoints are also (special) generalized endpoints.

[I-D.ietf-tsvwg-2960bis] has no notion of address scoping or other address handling limitations and provides no mechanism to change the addresses of an endpoint.

This means that an endpoint is simply a generalized endpoint which does not depend on the time. Neither the Port nor the address list changes.

During association setup no address scoping rules or other limitations will be applied. This means that for an association A between two endpoints gE' and gE'' the following is true:

Addr(A, gE') = Addr(gE') and Addr(A, gE'') = Addr(gE'').

A.5. Rules for address manipulation

The rules for address manipulation can now be stated in a simple way:

1. An address can be added to a generalized endpoint gE only if this address is not an address of a different generalized endpoint with the same port number.
2. An address can be added to an association A with generalized endpoint gE if it has been added to the generalized endpoint gE first. This means that the address must be an element of Addr(gE) first and then it can become an element of Addr(A, gE). But this is not necessary. If the association does not allow the reconfiguration of the addresses only Addr(gE) can be modified.
3. An address can be deleted from an association A with generalized endpoint gE as long as Addr(A, gE) stays non-empty.
4. An address can be deleted from an generalized endpoint gE only if it has been removed from all associations having gE as a generalized endpoint.

These rules simply make sure that the rules for the endpoints and associations given above are always fulfilled.

Authors' Addresses

Randall R. Stewart
Cisco Systems, Inc.
4875 Forest Drive
Suite 200
Columbia, SC 29206
US

Phone:
Email: rrs@cisco.com

Qiaobing Xie
Motorola, Inc.
1501 W. Shure Drive, #2309
Arlington Heights, IL 60004
USA

Phone: +1-847-632-3028
Email: qxie1@email.mot.com

Michael Tuexen
Univ. of Applied Sciences Muenster
Stegerwaldstr. 39
48565 Steinfurt
Germany

Email: tuexen@fh-muenster.de

Shin Maruyama
Kyoto University
Yoshida-Honmachi
Sakyo-ku
Kyoto, Kyoto 606-8501
JAPAN

Phone: +81-75-753-7468
Email: mail@marushin.gr.jp

Masahiro Kozuka
Kyoto University
Yoshida-Honmachi
Sakyo-ku
Kyoto, Kyoto 606-8501
JAPAN

Phone: +81-75-753-7468
Email: ma-kun@kozuka.jp

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).