     Stream Control Transmission Protocol (SCTP) Network Address Translation
                     draft-ietf-behave-sctpnat-00.txt

Status of this Memo

   By submitting this Internet-Draft, each author represents that any
   applicable patent or other IPR claims of which he or she is aware
   have been or will be disclosed, and any of which he or she becomes
   aware will be disclosed, in accordance with Section 6 of BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF), its areas, and its working groups. Note that
   other groups may also distribute working documents as Internet-
   Drafts.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   time. It is inappropriate to use Internet-Drafts as reference
   material or to cite them other than as "work in progress."

   The list of current Internet-Drafts can be accessed at
   http://www.ietf.org/ietf/1id-abstracts.txt.

   The list of Internet-Draft Shadow Directories can be accessed at
   http://www.ietf.org/shadow.html.

   This Internet-Draft will expire on April 25, 2009.

Abstract

   Stream Control Transmission Protocol [RFC4960] provides a reliable
   communications channel between two end-hosts in many ways similar to
   TCP [RFC0793]. With the widespread deployment of Network Address
   Translators (NAT), specialized code has been added to NAT for TCP
   that allows multiple hosts to reside behind a NAT and yet use only a
   single globally unique IPv4 address, even when two hosts (behind a
   NAT) choose the same port numbers for their connection. This
   additional code is sometimes classified as Network Address and Port
   Translation or NAPT. To date, specialized code for SCTP has NOT yet
   been added to most NATs so that only pure NAT is available. The end

   result of this is that only one SCTP capable host can be behind a
   NAT.

   This document describes an SCTP specific variant of NAT which
   provides similar features of NAPT in the single point and multi-point
   traversal scenario.

Table of Contents

1. Introduction

Stream Control Transmission Protocol [RFC4960] provides a reliable
communications channel between two end-hosts in many ways similar to
TCP [RFC0793]. With the widespread deployment of Network Address
Translators (NAT), specialized code has been added to NAT for TCP
that allows multiple hosts to reside behind a NAT and yet use only a
single globally unique IPv4 address, even when two hosts (behind a
NAT) choose the same port numbers for their connection. This
additional code is sometimes classified as Network Address and Port
Translation or NAPT. To date, specialized code for SCTP has NOT yet
been added to most NATs so that only true NAT is available. The end
result of this is that only one SCTP capable host can be behind a
NAT.

This document proposes an SCTP specific variant NAT that provides the
NAPT functionality without changing SCTP port numbers. The authors
feel it is possible and desirable to make these changes for a number
of reasons.

o It is desirable for SCTP internal end-hosts on multiple platforms
   to be able to share a NAT's public IP address, much as TCP does
   today.

o If a NAT does not need to change any data within an SCTP packet it
   will reduce the processing burden of NAT'ing SCTP by NOT needing
   to execute the CRC32c checksum required by SCTP.

o Not having to touch the IP payload makes the processing of ICMP
   messages in NATs easier.

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. Terminology

For this discussion we will use several terms, which we will define
and point out in a figure.

o Private-Address (Priv-Addr) - The private address that is known to
   the internal host.

o Internal-Port (Int-Port) - The port number that is in use by the
   host holding the Private-Address. Normally this is the port that
   will be translated by the NAPT to a different port number.

o Internal-VTag (Int-VTag) - The Verification Tag that the internal
   host has chosen for its communication. The VTag is a unique 32
   bit tag that must accompany any incoming SCTP packet for this
   association to the Private-Address.

o External-Address (Ext-Addr) - The address that an internal host is
   attempting to contact.

o External-Port (Ext-Port) - The port number of the peer process at
   the External-Address.

o External-VTag (Ext-VTag) - The Verification Tag that the host
   holding the External-Address has chosen for its communication.
   The VTag is a unique 32 bit tag that must accompany any incoming
   SCTP packet for this association to the External-Address.

o Public-Address (Pub-Addr) - The public address assigned to the NAT
   box which it uses as a source address when sending packets towards
   the External-Address.

```
   Internal Network      |           External Network
                         |
            Private      | Public                     External
 +---------+ Address     | Address     /--\/--\      Address +---------+
 |  SCTP   |             +-----+      /        \             |  SCTP   |
 |end point|=========| NAT |====== | Internet | ========= |end point|
 |    A    |             +-----+      \        /             |    B    |
 +---------+ Internal    |             \--/\--/      External +---------+
  Internal      Port     |                            Port    External
    VTag                 |                                      VTag
```
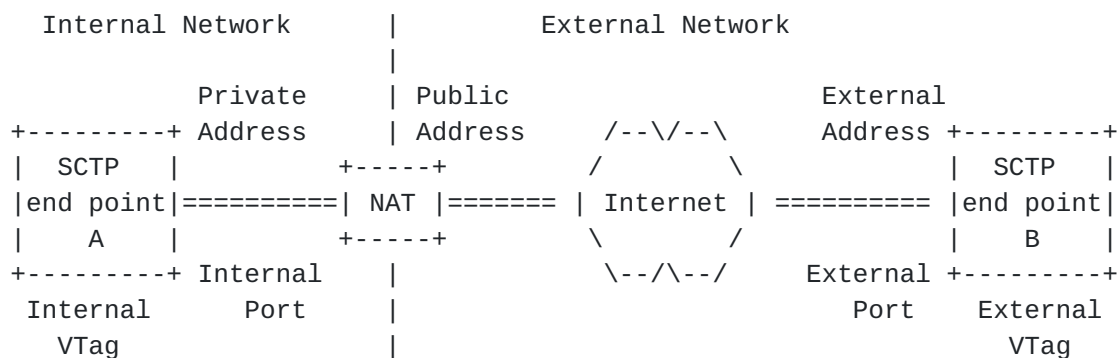
4. SCTP NAT Traversal Scenarios

4.1. Single Point Traversal

In this case, all packets in the SCTP association go through a single
NAT, as shown below:

```
   Internal Network       |         External Network
                          |
+---------+               |                 /--\/--\            +---------+
|  SCTP   |          +-----+               /        \          |  SCTP   |
|end point|=========| NAT |========= | Internet | ========|end point|
|    A    |          +-----+               \        /          |    B    |
+---------+               |                 \--/\--/            +---------+
                          |
```
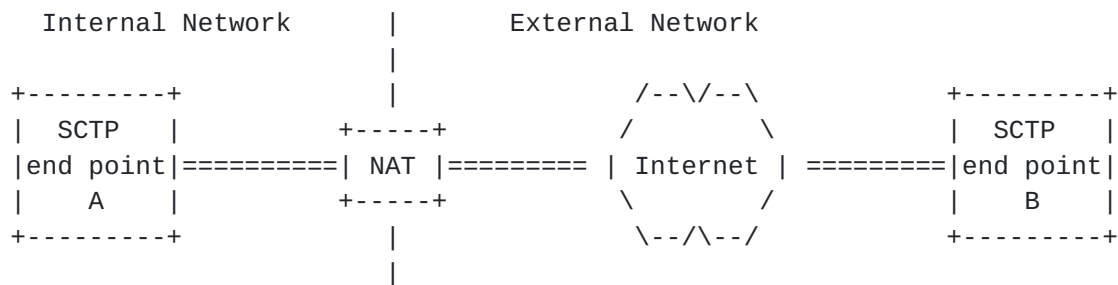
   A variation of this case is shown below, i.e., multiple NATs in a
   single path:

```
         Internal | External :: Internal | External
                  |          ::          |
+---------+       |          ::          |        /--\/--\    +---------+
|  SCTP   |  +-----+        ::       +-----+   /        \  |  SCTP   |
|end point|===| NAT |=======::======| NAT |==| Internet |==|end point|
|    A    |  +-----+        ::       +-----+   \        /  |    B    |
+---------+       |          ::          |        \--/\--/    +---------+
                  |          ::          |
```
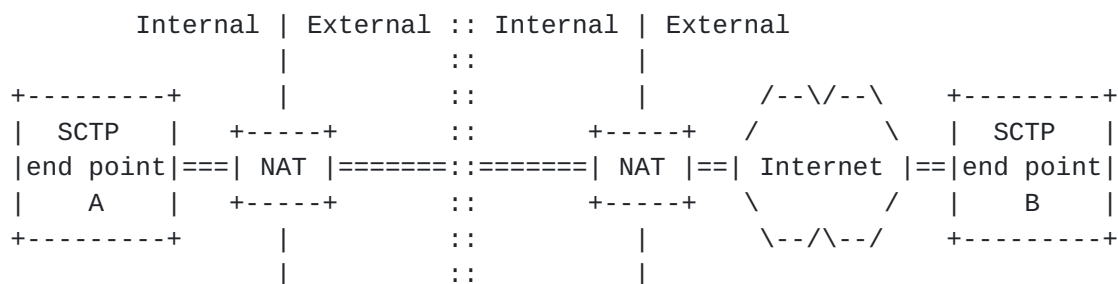
   The two SCTP endpoints in this case can be either single-homed or
   multi-homed. However, the important thing is that the NAT (or NATs)
   in this case sees ALL the packets of the SCTP association.

   In this single point traversal scenario, we must acknowledge that
   while one of the main benefits of SCTP multi-homing is redundant
   paths, the NAT function represents a single point of failure in the
   path of the SCTP multi-home association. However, the rest of the
   path may still benefit from path diversity provided by SCTP multi-
   homing.

4.2. Multi Point Traversal

   This case involves multiple NATs and each NAT only sees some of the
   packets in the SCTP association. An example is shown below:

```
          Internal        |       External
                   +------+              /---\/---\
+---------+  /=======|NAT A |=========\  /           \         +---------+
|  SCTP   | /        +------+          \/             \        |  SCTP   |
|end point|/      ...                   |   Internet   |=====|end point|
|    A    ||\                            \             /        |    B    |
+---------+ \        +------+           / \           /         +---------+
           \=======|NAT B |=========/   \---\/---/
                   +------+
                      |
```
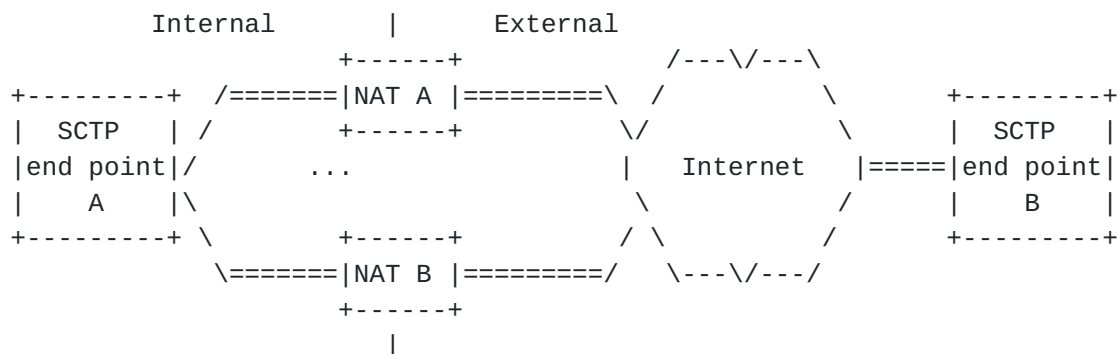
This case does NOT apply to a single-homed SCTP association (i.e., BOTH endpoints in the association use only one IP address). The advantage here is that the existence of multiple NAT traversal points can preserve the path diversity of a multi-homed association for the entire path. This in turn can improve the robustness of the communication.

To make this work, however, all the NATs involved must recognize the packets they see as belonging to the same SCTP association and perform address translation in a consistent way. This may require that a pre-defined table of ports and addresses were shared between the NATs. Other external management schemes that help multiple NATs coordinate a multi-homed SCTP association could be investigated.

5. The SCTP specific variant of NAT

In this section we assume that we have multiple SCTP capable hosts behind a NAT which has one Public-Address. Furthermore we are focusing in this section on the single point traversal scenario.

The modification of SCTP packets sent to the public Internet is easy. The source address of the packet has to be replaced with the Public-Address. It may also be necessary to establish some state in the NAT box to handle incoming packets, which is discussed later.

For SCTP packets coming from the public Internet the destination address of the packets has to be replaced with the Private-Address of the host the packet has to be delivered to. The lookup of the Private-Address is based on the External-VTag, External-Port, External-Address, Internal-VTag and the Internal-Port.

For the SCTP NAT processing the NAT box has to maintain a table of Internal-VTag, Internal-Port, Private-Address, External-VTag, External-Port and External-Address. An entry in that table is called a NAT state control block.

The processing of outgoing SCTP packets containing an INIT-chunk is described in the following figure. The scenario shown is valid for all message flows in this section.

```
                                    /--\/--\
 +--------+          +-----+       /        \          +--------+
 | Host A | <------> | NAT | <------> | Internet | <------> | Host B |
 +--------+          +-----+       \        /          +--------+
                                    \--/\---/
```
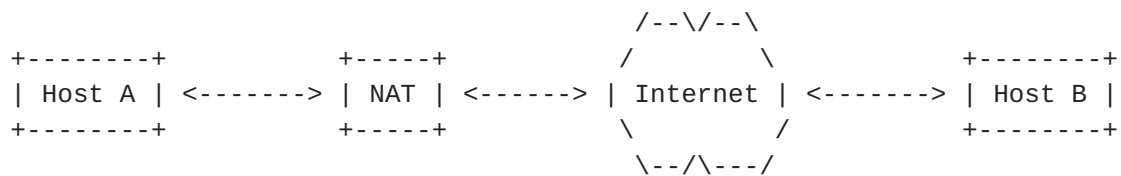
```
                INIT[Initiate-Tag]
 Priv-Addr:Int-Port ------> Ext-Addr:Ext-Port
                 Ext-VTag=0

                   Create(Initiate-Tag,Internal-Port,Private-Address,
                          0,External-Port,External-Address)
                   Returns(NAT-State control block)

            Translate To:

                     INIT[Initiate-Tag]
            Pub-Addr:Int-Port ------> Ext-Addr:Ext-Port
                        Ext-VTag=0
```

   It should be noted that normally a NAT control block will be created.
   However, it is possible that there is already a NAT control block
   with the same External-Address, External-Port, External-VTag,
   Internal-VTag but different Private-Address. In this case the INIT
   SHOULD be dropped and an ABORT MAY be sent back.

   The processing of outgoing SCTP packets containing no INIT-chunk is
   described in the following figure.

```
 Priv-Addr:Int-Port ------> Ext-Addr:Ext-Port
                 Ext-VTag


                          Translate To:

                          Pub-Addr:Int-Port ------> Ext-Addr:Ext-Port
                                        Ext-VTag
```

   The processing of incoming SCTP packets containing INIT-ACK chunks is
   described in the following figure.

```
                                          INIT-ACK[Initiate-Tag]
                            Pub-Addr:Int-Port <------ Ext-Addr:Ext-Port
                                          Int-VTag

            Lookup(Internal-VTag,Internal-Port,*,
                   0,External-Port,External-Address)
            Update(*, *, *, Initiate-Tag, *, *)

            Returns(NAT-State control block containing Private-Address)

                 INIT-ACK[Initiate-Tag]
    Priv-Addr:Int-Port <------ Ext-Addr:Ext-Port
                     Int-VTag
```

   In the case Lookup fails, the SCTP packet is dropped. The Update
   routine inserts the External-VTag (the Initiate-Tag of the INIT-ACK
   chunk) in the NAT state control block.

   The processing of incoming SCTP packets containing an ABORT or
   SHUTDOWN-COMPLETE chunk with the T-Bit set is described in the
   following figure.

```
                            Pub-Addr:Int-Port <------ Ext-Addr:Ext-Port
                                              Ext-VTag

              Lookup(0, Internal-Port, *,External-VTag,
                     External-Port, External-Address)

              Returns(NAT-State control block containing Private-Address)

    Priv-Addr:Int-Port <------ Ext-Addr:Ext-Port
                     Ext-VTag
```

   The processing of other incoming SCTP packets is described in the
   following figure.

```
                            Pub-Addr:Int-Port <------ Ext-Addr:Ext-Port
                                              Int-VTag

               Lookup(Internal-VTag, Internal-Port, *,
                      *, External-Port, External-Address)

               Returns(NAT-State control block containing Local-Address)

    Priv-Addr:Int-Port <------ Ext-Addr:Ext-Port
                     Int-VTag
```

   For an incoming packet containing an INIT-chunk a table lookup is

made only based on the addresses and port numbers. If an entry with
an Internal-VTag of zero is found, it is considered a match and the
Internal-VTag is updated.

This allows the handling of INIT-collision through NAT.

6. Handling of internal port number collisions

There is one drawback of the SCTP specific variant of NAT compared to
a NAPT solution like the ones available for TCP. Consider the case
where two hosts in the Private-Address space want to set up an SCTP
association with the same server running on the same host in the
Internet. This means that the External-Port and the External-Address
are the same. If they both choose the same Internal-Port the server
cannot distinguish both associations based on the address and port
numbers. For the server it looks like the association is being
restarted. To overcome this limitation the client sends a
NAT_SUPPORTED parameter in the INIT-chunk which is defined as
follows:

```
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 = 0xC007         |            Length=4           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

When the server receives this parameter it will also use the
verification tag to look up the association. However, this will make
it impossible to restart such associations.

7. Handling of internal port number and verification tag collisions

Consider the case where two hosts in the Private-Address space want
to set up an SCTP association with the same server running on the
same host in the Internet. This means that the External-Port and the
External-Address are the same. If they both choose the same
Internal-Port and Internal-VTag, the NAT box cannot distinguish
incoming packets anymore. But this is very unlikely. The Internal-
VTags are chosen at random and if the Internal-Ports are also chosen
from the ephemeral port range at random this gives a 46 bit random
number which has to match. In the TCP like NAPT case the NAT box can
control the 16 bit Natted Port.

However, in this unlikely event the NAT box MUST respond to the INIT
chunk by sending an ABORT chunk with the M-bit set. The source
address of the packet containing the ABORT chunk MUST be the

destination address of the SCTP packet containing the INIT chunk.
The sender of the packet containing the INIT chunk MAY start the
association setup procedure after choosing a new initiate tag.

The ABORT chunk defined in [RFC4960] is therefore extended by using
the following format:

```
 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 = 6    | Reserved  |M|T|           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                                                               \
/                   zero or more Error Causes                   /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The following error cause with cause code 0x00b0 (Colliding NAT table
entry) SHOULD be included in the ABORT chunk:

```
 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=0x00b0         |     Cause Length=Variable     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                         INIT chunk                            /
/                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

8. Handling of missing state

If the NAT box receives a packet for which the lookup procedure does
not find an entry in the NAT table, a packet containing an ERROR
packet is sent back with the M-bit set. The source address of the
packet containing the ERROR chunk MUST be the destination address of
the incoming SCTP packet. The verification tag is reflected.

The ERROR chunk defined in [RFC4960] is therefore extended by using
the following format:

```
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 = 9    | Reserved  |M|T|            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                                                               \
/                   zero or more Error Causes                   /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The following error cause with cause code 0x00b1 (Missing NAT table
entry) SHOULD be included in the ERROR chunk:

```
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=0x00b0         |     Cause Length=Variable     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                      Incoming Packet                          /
/                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

If an end-point receives a packet with this ERROR chunk it MAY send
an SCTP packet with an ASCONF chunk containing an Add IP Address
parameter followed by a vtag parameter:

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Parameter Type = 0xC007   |     Parameter Length = 12     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Internal Verification Tag                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   External Verification Tag                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

If the NAT box receives a packet for which it has no NAT table entry
and the packet contains an ASCONF chunk with a vtag parameter, the
NAT box MUST update its NAT table according to the verification tags
in the vtag parameter.

9. Multi Point Traversal considerations

If a multi-homed SCTP end-point behind a NAT connects to a peer, it
first sets up the association single-homed. Then it adds each IP
address using ASCONF chunks. The address to add is the wildcard
address and the lookup address also. The ASCONF chunks SHOULD also

contain a vtag parameter.

10. Handling of fragmented SCTP packets

A NAT box MUST support IP reassembly of received fragmented SCTP
packets. The fragments may arrive in any order.

When an SCTP packet has to be fragmented by the NAT box and the IP
header forbids fragmentation a corresponding ICMP packet SHOULD be
sent.

11. Simplification for small NATs

Small NAT boxes, i.e. NAT boxes which only have to support a small
number of concurrent SCTP associations, MAY not take the external
address into account when processing packets. Therefore the
External-Address could also be removed from the NAT table.

This simplification may make implementing a NAT box easier, however,
the collision probability is higher than using a mapping which takes
the external address into account.

12. Various examples of NAT traversals

12.1. Single-homed client to single-homed server

The internal client starts the association with the external server
via a four-way-handshake.

```
                                       /--\/--\
 +--------+               +-----+      /        \              +--------+
 | Host A | <-------> | NAT | <------> | Internet | <-------> | Host B |
 +--------+               +-----+      \        /              +--------+
                                       \--/\---/
        +---------+--------+----------+---------+--------+-----------+
 NAT    |  Int    |  Int   |   Priv   |   Ext   |  Ext   |   Ext     |
        |  VTag   |  Port  |   Addr   |   VTag  |  Port  |   Addr    |
        +---------+--------+--- -------+---------+--------+-----------+

 Host A sends INIT:

    INIT[Initiate-Tag = 1234]
 10.0.0.1:1 ------> 100.0.0.1:2
        Ext-VTtag = 0
```

```
        NAT creates entry:
        +---------+--------+----------+---------+--------+-----------+
NAT     |  Int    |  Int   |   Priv   |   Ext   |  Ext   |    Ext    |
        |  VTag   |  Port  |   Addr   |   VTag  |  Port  |    Addr   |
        +---------+--------+----------+---------+--------+-----------+
        |  1234   |   1    | 10.0.0.1 |    0    |   2    | 100.0.0.1 |
        +---------+--------+----------+---------+--------+-----------+

                           INIT[Initiate-Tag = 1234]
                101.0.0.1:1 -------------------------> 100.0.0.1:2
                                 Ext-VTtag = 0


                           INIT-ACK[Initiate-Tag = 5678]
                101.0.0.1:1 <------------------------- 100.0.0.1:2
                                 Int-VTag = 1234

    NAT updates entry:
        +---------+--------+----------+---------+--------+-----------+
NAT     |  Int    |  Int   |   Priv   |   Ext   |  Ext   |    Ext    |
        |  VTag   |  Port  |   Addr   |   VTag  |  Port  |    Addr   |
        +---------+--------+----------+---------+--------+-----------+
        |  1234   |   1    | 10.0.0.1 |   5678  |   2    | 100.0.0.1 |
        +---------+--------+----------+---------+--------+-----------+


 INIT-ACK[Initiate-Tag = 5678]
 10.0.0.1:1 <------ 100.0.0.1:2
         Int-VTag = 1234

         COOKIE-ECHO
 10.0.0.1:1 ------> 100.0.0.1:2
       Ext-VTag = 5678

                                  COOKIE-ECHO
                101.0.0.1:1 -------------------------> 100.0.0.1:2
                                 Ext-VTag = 5678


                                  COOKIE-ACK
                101.0.0.1:1 <------------------------- 100.0.0.1:2
                                 Int-VTag = 1234

           COOKIE-ACK
 10.0.0.1:1 <------ 100.0.0.1:2
         Int-VTag = 1234
```

12.2. Single-homed client to multi-homed server

   The internal client is single-homed whereas the external server is
   multi-homed. The server includes its addresses in the INIT-ACK
   chunk, which results in two NAT entries.

```
                                              +--------+
                                 /--\/--\      /-|Router 1| \
   +------+            +-----+      /        \    /  +--------+  \   +------+
   | Host | <------> | NAT | <-> | Internet | ==                  ==| Host |
   |   A  |            +-----+      \        /    \  +--------+  /   |   B  |
   +------+                          \--/\--/       \-|Router 2|-/   +------+
                                                      +--------+
```

```
         +---------+--------+----------+---------+--------+----------+
  NAT    |   Int   |  Int   |   Priv   |   Ext   |  Ext   |   Ext    |
         |   VTag  |  Port  |   Addr   |   VTag  |  Port  |   Addr   |
         +---------+--------+---  ------+---------+--------+----------+
```

```
  INIT[Initiate-Tag = 1234]
 10.0.0.1:1 ---> 100.0.0.1:2
       Ext-VTag = 0
```

       NAT creates entry:
```
         +---------+--------+----------+---------+--------+----------+
  NAT    |   Int   |  Int   |   Priv   |   Ext   |  Ext   |   Ext    |
         |   VTag  |  Port  |   Addr   |   VTag  |  Port  |   Addr   |
         +---------+--------+----------+---------+--------+----------+
         |  1234   |    1   | 10.0.0.1 |    0    |    2   | 100.0.0.1 |
         +---------+--------+----------+---------+--------+----------+
```

```
                       INIT[Initiate-Tag = 1234]
           101.0.0.1:1 -----------------------------> 100.0.0.1:2
                           Ext-VTag = 0
```

```
                 INIT-ACK[Initiate-tag = 5678, IP-Addr = 100.1.0.1]
           101.0.0.1:1 <----------------------------- 100.0.0.1:2
                           Int-VTag = 1234
```

       NAT updates first entry and creates entry for second address:
```
         +---------+--------+----------+---------+--------+----------+
  NAT    |   Int   |  Int   |   Priv   |   Ext   |  Ext   |   Ext    |
         |   VTag  |  Port  |   Addr   |   VTag  |  Port  |   Addr   |
         +---------+--------+----------+---------+--------+----------+
         |  1234   |    1   | 10.0.0.1 |  5678   |    2   | 100.0.0.1 |
         |  1234   |    1   | 10.0.0.1 |  5678   |    2   | 100.1.0.1 |
```

```
         +---------+--------+----------+----------+--------+-----------+



  INIT-ACK[Initiate-Tag = 5678]
   10.0.0.1:1 <--- 100.0.0.1:2
          Int-VTag = 1234


         COOKIE-ECHO
   10.0.0.1:1 ---> 100.0.0.1:2
         ExtVTag = 5678


                                    COOKIE-ECHO
                  101.0.0.1:1 ------------------------------> 100.0.0.1:2
                                    Ext-VTag = 5678


                                    COOKIE-ACK
                  101.0.0.1:1 <------------------------------ 100.0.0.1:2
                                    Int-VTag = 1234


           COOKIE-ACK
   10.0.0.1:1 <--- 100.0.0.1:2
          Int-VTag = 1234
```
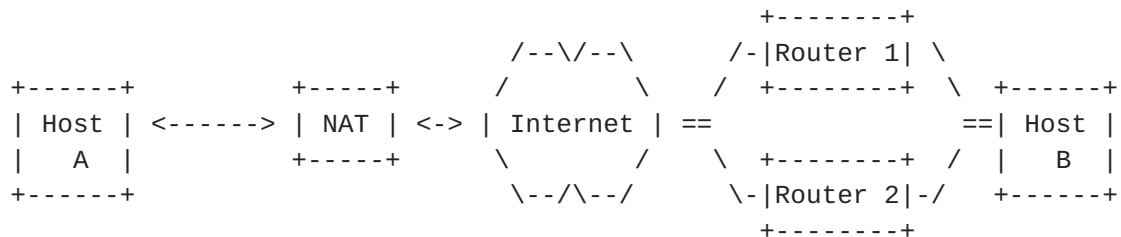
12.3. Multihomed client and server

```
                   +-------+
          /--------| NAT 1 |--------\        /--\/--\
 +------+    /         +-------+          \      /         \      +--------+
 | Host |====                             ====| Internet |====| Host B |
 |   A  |    \         +-------+          /      \         /      +--------+
 +------+     \--------| NAT 2 |--------/        \--/\--/
                   +-------+
```

```
      +---------+--------+----------+----------+--------+-----------+
NAT 1 |   Int   |  Int   |   Priv   |   Ext    |  Ext   |   Ext     |
      |  VTag   |  Port  |   Addr   |  VTag    |  Port  |   Addr    |
      +---------+--------+--- -------+----------+--------+-----------+
```

```
  INIT[Initiate-Tag = 1234]
 10.0.0.1:1 --------> 100.0.0.1:2
         Ext-VTag = 0

      NAT 1 creates entry:
      +---------+--------+----------+----------+--------+-----------+
NAT 1 |   Int   |  Int   |   Priv   |   Ext    |  Ext   |   Ext     |
      |  VTag   |  Port  |   Addr   |  VTag    |  Port  |   Addr    |
```

```
      +---------+--------+-----------+----------+--------+-----------+
      | 1234    |    1   | 10.0.0.1  |    0     |   2    | 100.0.0.1 |
      +---------+--------+-----------+----------+--------+-----------+


                            INIT[Initiate-Tag = 1234]
                  101.0.0.1:1 ------------------------> 100.0.0.1:2
                                   ExtVTag = 0


                  INIT-ACK[Initiate-Tag = 5678, IP-Addr = 100.1.0.1]
                  101.0.0.1:1 <------------------------ 100.0.0.1:2
                                   Int-VTag = 1234

NAT 1 updates first entry and creates complete entry for second address:
      +---------+--------+-----------+----------+--------+-----------+
NAT 1 |  Int    |  Int   |   Priv    |   Ext    |  Ext   |    Ext    |
      |  VTag   |  Port  |   Addr    |   VTag   |  Port  |    Addr   |
      +---------+--------+-----------+----------+--------+-----------+
      | 1234    |    1   | 10.0.0.1  |   5678   |   2    | 100.0.0.1 |
      | 1234    |    1   | 10.0.0.1  |   5678   |   2    | 100.1.0.1 |
      +---------+--------+-----------+----------+--------+-----------+


   INIT-ACK[Initiate-Tag = 5678]
 10.0.0.1:1 <---------100.0.0.1:2
          Int-VTag = 1234

         COOKIE-ECHO
 10.0.0.1:1 --------> 100.0.0.1:2
         Ext-VTag = 5678


                                   COOKIE-ECHO
                  101.0.0.1:1 ---------------------> 100.0.0.1:2
                                   Ext-VTag = 5678


                                   COOKIE-ACK
                  101.0.0.1:1 <--------------------- 100.0.0.1:2
                                   Int-VTag = 1234

         COOKIE-ACK
 10.0.0.1:1 <------- 100.0.0.1:2
          Int-VTag = 1234

Host A announces the second address

ASCONF [ADD-IP,INT-VTag=1234, Ext-VTag = 5678]
```

```
10.1.0.1:1 --------> 100.1.0.1:2
          Ext-VTag = 5678
```

```
     NAT 2 creates complete entry:
        +---------+--------+-----------+----------+--------+-----------+
NAT 2   |  Int    |  Int   |   Priv    |   Ext    |  Ext   |    Ext    |
        |  VTag   |  Port  |   Addr    |   VTag   |  Port  |    Addr   |
        +---------+--------+-----------+----------+--------+-----------+
        |  1234   |    1   | 10.1.0.1  |   5678   |    2   | 100.1.0.1 |
        +---------+--------+-----------+----------+--------+-----------+
```

```
              ASCONF [ADD-IP,Int-VTag=1234, Ext-VTag = 5678]
                 101.1.0.1:1 ------------------------> 100.1.0.1:2
                                        Ext-VTag = 5678

                                       ASCONF-ACK
                 101.1.0.1:1 <------------------------ 100.1.0.1:2
                                        Int-VTag = 1234

          ASCONF-ACK
10.1.0.1:1 <----- 100.1.0.1:2
          Int-VTag = 1234
```
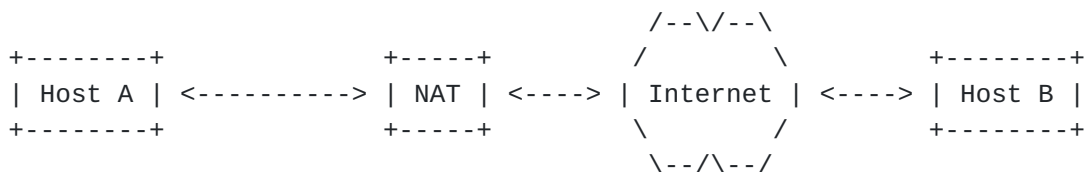
[12.4](#). **NAT loses its state**

   Assocation is already established between Host A and Host B.

```
                                      /--\/--\
+--------+                +-----+    /        \         +--------+
| Host A | <----------> | NAT | <----> | Internet | <----> | Host B |
+--------+                +-----+    \        /         +--------+
                                      \--/\--/

       +---------+--------+-----------+----------+--------+-----------+
NAT A  |  Int    |  Int   |   Priv    |   Ext    |  Ext   |    Ext    |
       |  VTag   |  Port  |   Addr    |   VTag   |  Port  |    Addr   |
       +---------+--------+-----------+----------+--------+-----------+
       |  1234   |    1   | 10.0.0.1  |   5678   |    2   | 100.0.0.1 |
       +---------+--------+-----------+----------+--------+-----------+
```

   The NAT loses its state and obtaines a new public address.

```
              DATA
 10.0.0.1:1 ----------> 100.0.0.1:2
          Ext-VTag = 5678
```

```
                           NAT cannot find entry: sends ERROR message
      ERROR [M-Bit, NAT state missing]
    10.0.0.1:1 <---------- 100.0.0.1:2
            Ext-VTag = 5678


   ASCONF [ADD-IP,DELETE-IP,Int-VTag=1234, Ext-VTag = 5678]
    10.0.0.1:1 ----------> 100.1.0.1:2
            Ext-VTag = 5678


        +---------+--------+----------+---------+--------+-----------+
 NAT A  |   Int   |  Int   |   Priv   |   Ext   |  Ext   |    Ext    |
        |   VTag  |  Port  |   Addr   |   VTag  |  Port  |   Addr    |
        +---------+--------+----------+---------+--------+-----------+
        |   1234  |    1   | 10.0.0.1 |   5678  |    2   | 100.0.0.1 |
        +---------+--------+----------+---------+--------+-----------+


        ASCONF [ADD-IP,DELETE-IP,Int-VTag=1234, Ext-VTag = 5678]
                      102.1.0.1:1 ----------------------> 100.1.0.1:2
                                    Ext-VTag = 5678

   Host B adds new source address and deletes all former entries.

                                          ASCONF-ACK
                      102.1.0.1:1 <---------------------- 100.1.0.1:2
                                          Int-VTag = 1234

                ASCONF-ACK
    10.1.0.1:1 <---------- 100.1.0.1:2
    Int-VTag = 1234                    1

                DATA
    10.0.0.1:1 ----------> 100.0.0.1:2
            Ext-VTag = 5678
                                          DATA
                      102.1.0.1:1 ----------------------> 100.1.0.1:2
                                    Ext-VTag = 5678
```
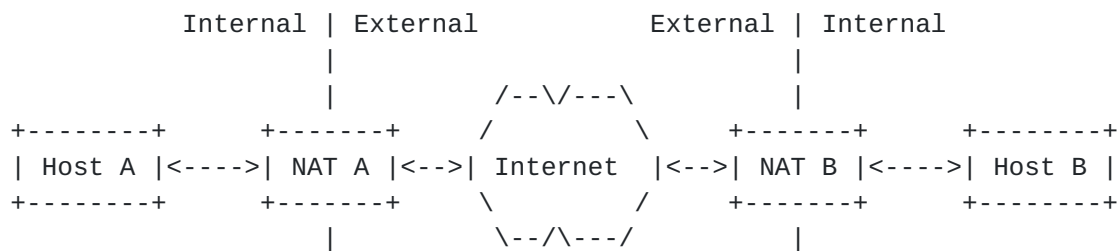
[12.5](#). **Peer-to-Peer Communication**

   If two hosts are behind NATs, they have to get knowledge of the
   peer's public address. This can be achieved with a so-called
   rendezvous server. Afterwards the destination addresses are public,
   and the association is set up with the help of the INIT collision.
   The NAT boxes create their entries according to their internal peer's
   point of view. Therefore, NAT A's Internal-VTag and Internal-Port
   are NAT B's External-VTag and External-Port, respectively. The
   naming of the verification tag in the packet flow is done from the

sending peer's point of view.

```
            Internal | External          External | Internal
                     |                            |
                     |          /--\/---\          |
 +--------+      +-------+     /         \    +-------+      +--------+
 | Host A |<---->| NAT A |<-->| Internet  |<-->| NAT B |<---->| Host B |
 +--------+      +-------+     \         /    +-------+      +--------+
                     |          \--/\---/          |
```

NAT-Tables

```
       +---------+--------+-----------+---------+--------+-----------+
NAT A  |   Int   |  Int   |   Priv    |   Ext   |  Ext   |    Ext    |
       |  VTag   |  Port  |   Addr    |  VTag   |  Port  |    Addr   |
       +---------+--------+--- -------+---------+--------+-----------+


       +---------+--------+-----------+---------+--------+-----------+
NAT B  |   Int   |  Int   |   Priv    |   Ext   |  Ext   |    Ext    |
       |  v-tag  |  port  |   addr    |  v-tag  |  port  |    addr   |
       +---------+--------+--- -------+---------+--------+-----------+
```

Host A sends INIT:

```
INIT[Initiate-Tag = 1234]
10.0.0.1:1 --> 100.0.0.1:2
      Ext-VTag = 0
```

NAT A creates entry:

```
       +---------+--------+-----------+---------+--------+-----------+
NAT A  |   Int   |  Int   |   Priv    |   Ext   |  Ext   |    Ext    |
       |  VTag   |  Port  |   Addr    |  VTag   |  Port  |    Addr   |
       +---------+--------+-----------+---------+--------+-----------+
       |  1234   |   1    | 10.0.0.1  |    0    |   2    | 100.0.0.1 |
       +---------+--------+-----------+---------+--------+-----------+

                  INIT[Initiate-Tag = 1234]
          101.0.0.1:1 ----------------> 100.0.0.1:2
                     Ext-VTag = 0


                              NAT B processes INIT

                              SCTP packet silently discarded

       +---------+--------+-----------+---------+--------+-----------+
NAT B  |   Int   |  Int   |   Priv    |   Ext   |  Ext   |    Ext    |
       |  VTag   |  Port  |   Addr    |  VTag   |  Port  |    Addr   |
       +---------+--------+-----------+---------+--------+-----------+
```

```
                                          Host B sends INIT:

                                            INIT[Initiate-Tag = 5678]
                                            101.0.0.1:1 <-- 10.1.0.1:2
                                                        Ext-VTag = 0

                             NAT B processes INIT:


        +---------+--------+-----------+----------+--------+-----------+
 NAT B  |  Int    |  Int   |   Priv    |   Ext    |  Ext   |    Ext    |
        |  VTag   |  Port  |   Addr    |   VTag   |  Port  |   Addr    |
        +---------+--------+-----------+----------+--------+-----------+
        |  5678   |    2   | 10.1.0.1  |    0     |    1   | 101.0.0.1 |
        +---------+--------+-----------+----------+--------+-----------+

                             NAT B forwards INIT:

                        INIT[Initiate-Tag = 5678]
                101.0.0.1:1  <--------------- 100.0.0.1:2
                              Ext-VTag = 0

                  NAT A processes INIT and updates entry:

                VTag != Int-VTag, but Ext-VTag == 0, find entry.
        +---------+--------+-----------+----------+--------+-----------+
 NAT A  |  Int    |  Int   |   Priv    |   Ext    |  Ext   |    Ext    |
        |  VTag   |  Port  |   Addr    |   VTag   |  Port  |   Addr    |
        +---------+--------+-----------+----------+--------+-----------+
        |  1234   |    1   | 10.0.0.1  |   5678   |    2   | 100.0.0.1 |
        +---------+--------+-----------+----------+--------+-----------+

                  NAT A forwards INIT:

   INIT[Initiate-tag = 5678]
  10.0.0.1:1 <-- 100.0.0.1:2
          Ext-VTag = 0

  Host A send INIT-ACK:

  INIT-ACK[Initiate-Tag = 1234]
  10.0.0.1:1 --> 100.0.0.1:2
        Ext-VTag = 5678


                    INIT-ACK[Initiate-Tag = 1234]
                101.0.0.1:1 ----------------> 100.0.0.1:2
                            Ext-VTag = 5678
```

                                             NAT B updates entry:

```
        +---------+--------+-----------+----------+--------+-----------+
NAT B   |   Int   |  Int   |   Priv    |   Ext    |  Ext   |    Ext    |
        |  VTag   |  Port  |   Addr    |   VTag   |  Port  |    Addr   |
        +---------+--------+-----------+----------+--------+-----------+
        |  5678   |   2    | 10.1.0.1  |   1234   |   1    | 101.0.0.1 |
        +---------+--------+-----------+----------+--------+-----------+
```

                                     INIT-ACK[Initiate-Tag = 1234]
                                         101.0.0.1:1 --> 10.1.0.1:2
                                                Ext-VTag = 5678

                                             COOKIE-ECHO
                                         101.0.0.1:1 <-- 10.1.0.1:2
                                                Ext-VTag = 1234

                             COOKIE-ECHO
                  101.0.0.1:1 <------------- 100.0.0.1:2
                             Ext-VTag = 1234

          COOKIE-ECHO
    10.0.0.1:1 <-- 100.0.0.1:2
          Ext-VTag = 1234

          COOKIE-ACK
    10.0.0.1:1 --> 100.0.0.1:2
          Ext-VTag = 5678

                             COOKIE-ACK
                  101.0.0.1:1 ----------------> 100.0.0.1:2
                             Ext-VTag = 5678

                                             COOKIE-ACK
                                         101.0.0.1:1 --> 10.1.0.1:2
                                                Ext-VTag = 5678

[13]. IANA Considerations

   TBD

[14]. Security considerations

   State maintenance within a NAT is always a subject of possible Denial
   Of Service attacks. This document recommends that at a minimum a NAT
   runs a timer on any SCTP state so that old association state can be

cleaned up.

15. Acknowledgments

The authors wish to thank Qiaobing Xie, Henning Peters, Bryan Ford, David Hayes, Alfred Hines, Dan Wing, and Jason But for their invaluable comments.

16. References

16.1. Normative References

[RFC0793]   Postel, J., "Transmission Control Protocol", STD 7,
            RFC 793, September 1981.

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

[RFC4960]   Stewart, R., "Stream Control Transmission Protocol",
            RFC 4960, September 2007.

16.2. Informative References

[RFC1918]   Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and
            E. Lear, "Address Allocation for Private Internets",
            BCP 5, RFC 1918, February 1996.

Authors' Addresses

   Randall R. Stewart
   The Resource Group
   1700 Pennsylvania Ave NW
   Suite 56
   Washington, DC 20006
   USA

   Phone:
   Email: randall.stewart@trgworld.com

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

Email: tuexen@fh-muenster.de

Irene Ruengeler
Muenster Univ. of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
Germany

Email: i.ruengeler@fh-muenster.de