Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: April 2, 2009

R. Despres

September 29, 2008

Stateless Address Mapping with A+P Extended IPv4 addressing (SAM)
draft-despres-sam-00

Status of this Memo

Abstract

Stateless Local Address Mapping (SAM) is a generic tool for global-address packets to traverse transit domains where routing is performed in different address spaces. To share IPv4 global addresses among several CPEs and/or hosts, port prefixes can be used as extensions of IPv4 global addresses. In this space (IPv4E), a node having an n-bits IPv4E prefix with n>32 may only use or delegate ports having its port prefix of length /32-n. Static Address Mappers can be placed in CPEs, in hosts, and/or in ISP Internet gateways. Applications include various IPv6 in IPv4 and IPv4E in IPv6 encapsulations.

Table of Contents

1. Introduction

This document introduces Stateless Local Address Mapping (SAM), a generic tool for global-address packets to traverse transit domains where routings are in different address spaces.

To statically share IPv4 global addresses among several CPEs and/or hosts, port prefixes are used as extensions of IPv4 global addresses. In this space (IPv4E), a node having an n-bits IPv4E prefix with n > 32 may only use, or delegate, ports that start with its port prefix (the n - 32 low order bits of the IPv4E prefix).

Mechanisms that have already been deployed for IPv6 packets to traverse IPv4 domains, in particular ISATAP, 6to4, and 6rd, are applications of SAM with specific parameter values.

Section 2 describes the general architecture of SAM configurations, with all their possible parameters. It also describes stateless mapping rules by which source and destination addresses of encapsulating packets are derived from those of packets to be tunneled.

In Section 3, detailed packet processing, including anti-spoofing checks, is presented in pseudo-code. Until some running code is written and tested, these algorithms are not claimed to be error proof. They should therefore be considered as provisional.

Section 4 indicates how ISATAP [RFC4214], 6to4 [RFC3056] and 6rd [I-D a]can be seen as specific applications of the general SAM model, with ad hoc parameter values.

A companion document, [I-D b], presents several configurations where SAM is used to provide global IPv4 connectivity to customer sites that have only shared global IPv4 addresses in a more scalable way than with NATs in ISP infrastructures, and with possible end-to-end network transparency to IPv4 packets in favorable configurations.

2. SAM operation

As shown on Figure 1, SAM concerns packets that traverse a "transit domain" situated between a "core domain" and a number of "branch domains".

Stateless Address Mappers (SAMs) are placed at borders between these domains. Being stateless, they can be duplicated any number of times for load sharing. Routes toward them are for this based on prefixes or on anycast addresses.

SAMs that are between branch domains and the transit domain are the "branch SAMs". They can receive all their parameters in DHCP (possibly DHCPv6). Those that are placed between the transit domain and the core domain are the "core SAMs". Their parameter settings would typically be less automatic.

The global Internet, in IPv4 and/or in IPv6, is accessible via the core domain, in which the address space is global.

Global packets that are exchanged between hosts of the branch domain ("branch hosts"), and hosts accessible via the core domain ("core hosts") are encapsulated to traverse the transit domain.

In each address family v (IPv4E or IPv6) in which a branch host X has an address, this address is structured as follows: Xv = Tv.Ivi.Sv, where Tv is the global prefix of the transit domain, Ivi is an infix that identifies the branch domain in the transit domain, and Sv is a suffix that identifies X in the branch domain. The infix is the same for both address families.

In an encapsulating packet of address family v that conveys a packet of family w toward or from branch host X, the address TXv that is derived from Xw, that of X, is structured as follows: TXv = Hv.Ivi.Sw.0/n, where Hv is a header that, in the transit domain, is at the beginning of all prefixes of branch domains, and where n is 32 for IPv4 encapsulating packets and 128 for IPv6 encapsulating packets [Figure 2]. Thus, although IPv4E addresses have 32 + 16 = 48 bits, packets can traverse the transit domain without routers having to route on more than 32 bits. (If k bits are necessary to identify branch domains, H4 should be taken equal to 32 - k.)

The address that, in encapsulating packets, corresponds to that of a core host Y is the anycast address Cv of core SAM gateways of the transit domain.

To be complete, the SAM model doesn't deal only with the transparent traversal of transit domains by global packets. It deals also with packets of branch host that have private IPv4 addresses and must be encapsulated in IPv6 to reach a NAT at the transit domain - core domain border (a Carrier grade NAT or CGN). The CGN can be IPv4 only as far as packet content is concerned, but they have to exercise their stateful address mapping with "composite" addresses at their transit side. The composite address of a host X that has XS as its private address is a combination of this address and of the encapsulating address derived from it. In the encapsulating packet of a CGN traversing packet, the core side address is the unicast IPv6 address N6 of the CGN in the transit domain.

```
                   BRANCH SAMs               CORE SAMs
               possible parameters:     possible parameters:
                 idem core SAM             T4, H4, C4
                + I4i, I6i, N4, N6         T6, H6, C6
                          |                     |
                          |                     V
                          |                     .----------------
                          V                     |  CORE domain
                          .---------------------|
                          |    TRANSIT domain   |
          ----------------|                     |
          BRANCH domain i |              C4 ---> <--- T4
                          |              C6 ---> <--- T6
                          |                     |
            0.0.0.0/0 ---> <--- B4i=H4.Ii       |
                0::/0 ---> <--- B6i=H6.Ii       |
                          |                     |
     <--- BRANCH host X   |                     |  CORE host Y --->
                          |  TRANSIT addresses  |
          addresses in    |   in encapsulating  |     addresses in
      encapsulated packets|        packets      | encapsulated packets
        <== [Xv, Yv] ==>  o  <== [TXv, Cv] ==>  o <== [Xv, Yv] ==>
        <== [Xv, X'v] ==> o  <== [TXv, TX'v] =. |
                          |                   | |
                          |                <==' |
                          |                     |
        <== [X4B, Y4] ==> o  <== [TX6, N6] ==>  |
                          |                     |
          ----------------|           N6 ---> [CGN] <--- T4b
                          |                     |
                          |                     |
                          '---------------------'
                                                |
                                                '----------------
          v    : address family 4 or 6 (for IPv4 or IPv6)
          Hv   : Header of all addresses in the transit domain
          Bvi  : prefix of Branch domain i in in the transit domain
                 (Bvi = Hvi.Ii)
          Ivi  : Infix of branch domain i (Ivi = Bvi - Hv)
          Cv   : anycast address of core domain gateways
          Nv   : unicast address of a CGN at the core domain border
          Tv   : prefix of the transit domain in the core domain
          TXv  : Transit address of branch host X

       ARCHITECTURE AND POSSIBLE PARAMETERS OF STATIC ADDRESS MAPPINGS

                                 Figure 1
```

```
                         glob. v4 add.    port
                       <------32------><---16-->
                      +-----.---------+-.------.
                      |  T4 |     Ii    |  S4  |
                      +-----'---------+-'------'
                      <--g--><----i-----><--s-->
                      Branch host IPv4E address (X4)


    <-------------------------------128---------------------------->
    +------------------.-----------.-------------------------------+
    |       T6         |     Ii    |              S6               |
    +------------------'-----------'-------------------------------+
    <-------g---------><----i-----><--------------s--------------->
                       Branch host IPv6 address (X6)


                           <------32------>
                           +---.-----------+
                           |H4 |     Ii    |
                           +---'-----------+
                           <-h-><----i----->
             IPv4 TRANSIT address (TX4) for a Branch host


    <-------------------------------128---------------------------->
    +-------------.-----------.------------------------------------+
    |      H6     |     Ii    |                  0                 |
    +-------------'-----------'------------------------------------+
    <-----h-------><----i-----><--s-->
              IPv6 TRANSIT address (TX6) for a Branch host
                  the global address of which is IPv4


     <-------------------------------128---------------------------->
    +-------------.-----------.------.-----------------------------+
    |      H6     |     Ii    |   Sv |             0               |
    +-------------'-----------'------'-----------------------------+
    <-----h-------><----i-----><--s-->
              IPv6 TRANSIT address (TX6) for a Branch host
                  the global address of which is IPv6


             GLOBAL TO TRANSIT ADDRESS MAPPINGs FOR BRANCH HOST

                                Figure 2
```

3. Detailed processing rules

Processing rules that result from the above description are detailed in Figure 6 to Figure 8. They include anti-spoofing tests whereby consistency between addresses of encapsulating packets and encapsulated packets are systematically verified.

In the pseudo-code, A and B are prefixes with B contained at the beginning of A, A - B stands for what follows B in A. In other words, with he dot as concatenation operator, A = B.(A - B). The pseudo-code notation is otherwise expected to be self explanatory.

```
CASE X4 = G4..
DO    CASE  Y4 NOT= G4..
      DO    CASE  C4 NOT= nil
            DO    TY4 <- C4
                  TX4 <- H4.(X4E-G4).0/32
                  Encapsulate 4/4
            CASE  C4 = nil & C6 NOT= nil
            DO    TY6 <- C6
                  TX6 <- H6.(X4E-G4).0/128
                  Encapsulate 4/6
            CASE  C4=nil & C6=nil & N4 NOT= nil
            DO    TY4 <- N4
                  TX4 <- H4.(X4E-G4).0/32
                  Encapsulate 4/4
            CASE  C4=nil & C6=nil & N4=nil
                  & N6 NOT= nil
            DO    TY6 <- N6
                  TX6 <- H6.(X4E-G4).0/128
                  Encapsulate 6/4
      CASE  Y4 = G4..
      DO    CASE  H4 NOT= nil
            DO    TY4 <- H4.(Y4E-G4).0/32
                  TX4 <- H4.(S4E-G4).0/32
                  Encapsulate 4/4
            CASE  H4 = nil & H6 NOT= nil
            DO    TY6 <- Y6.(Y4E-G4).0/128
                  TX6 <- H6.(X4E-G4).0/128
                  Encapsulate 4/6
CASE  X4 NOT= G4..
DO    Discard packet
```

BRANCH-SAM PROCESSING OF AN IPV4E CORE-BOUND PACKET

Figure 3

```
            CASE X6 = G6..
                  CASE  Y6 NOT= G6::
                  DO    CASE  H4 NOT= nil
                        DO    TY4 <- C4
                              TX4 <- H4.(X6-G6).0/32
                              Encapsulate 6/4
                        CASE  H4 = nil & H6 NOT= nil
                        DO    TY6 <- C6
                              TX6 <- H6.(X6-G6).0/128
                              Encapsulate 6/6
                        CASE  C4=nil & C6=nil & N4 NOT= nil
                        DO    TY4 <- N4
                              TX4 <- H4.(X6-G6).0/32
                              Encapsulate 4/4
                        CASE  C4=nil & C6=nil & N4=nil
                              & N6 NOT= nil
                        DO    TY6 <- N6
                              TX6 <- H6.(X6-G6).0/128
                              Encapsulate 6/4
                  CASE  Y6 = G6..
                  DO    CASE  H4 NOT= nil
                        DO    TY4 <- H4.(Y6-G6).0/32
                              TX4 <- H4.(X6-G6).0/32
                              Encapsulate 6/4
                        CASE  H4 = nil & H6 NOT= nil
                        DO    TY6 <- Y6.(Y6-G6).0/128
                              TX6 <- H6.(X6-G6).0/128
                              Encapsulate 6/6
            CASE  X6 NOT= G6..
            DO    Discard packet

              BRANCH-SAM PROCESSING OF AN IPV6 CORE-BOUND PACKET
```

Figure 4

```
           CASE Encapsulating packet is v4
                 CASE  Encapsulated packet is v4
                 DO    Decapsulate 4/4, getting X4 and Y4
                       IF    X4=G4.. & TX4 =  H4.(X4-G4).0/32
                             & [TY4=C4 OR TY4=N4
                                OR (Y4 = G4.. & TY4=H4.(Y4-G4)..]
                       DO    Forward decapsulated packet
                       ELSE  Discard packet
                 CASE  Encapsulated packet is v6
                 DO    Decapsulate 6/4, getting X4 and Y4
                       IF    X6=G6.. & TX4 =  H4.(X6-G6).0/32
                             & [TY4=C4 OR TY4=N4
                                OR (Y6 = G6.. & TY4=H4.(Y6-G6)..]
                       DO    Forward decapsulated packet
                       ELSE  Discard packet
           CASE Encapsulating packet is v6
                 CASE  Encapsulated packet is v4
                 DO    Decapsulate 4/6, getting X4 and Y4
                       IF    X4=G4.. & TX6 =  H6.(X4-G4).0/128
                             & [TY6=C6 OR TY6=N6
                                OR (Y4 = G4.. & TY6=H6.(Y4-G4)..]
                       DO    Forward decapsulated packet
                       ELSE  Discard packet
                 CASE  Encapsulated packet is v6
                 DO    Decapsulate 6/6, getting X6 and Y6
                       IF    X6=G6.. & TX6 =  H6.(X6-G6).0/128
                             & [TY6=C6 OR TY6=N6
                                OR (Y6 = G6.. & TY6=H6.(Y6-G6)..]
                       DO    Forward decapsulated packet
                       ELSE  Discard packet

             BRANCH-SAM PROCESSING OF A BRANCH-BOUND PACKET
```

Figure 5

```
CASE  Encapsulating packet is v4
      CASE  Encapsulated packet is v4
      DO    Decapsulate 4/4, getting X4 and Y4
            IF    X4 = G4.. & TX4 = H4.(X4-G4).0/32
                  & Y4 NOT= G4..
            DO    Forward decapsulated packet
            ELSE  Discard packet
      CASE  Encapsulated packet is v6
      DO    Decapsulate 6/4, getting X6 and Y6
            IF    X6 = G6.. & TX4 = H4.(X6-G6).0/32
                  & Y4 NOT= G4..
            DO    Forward decapsulated packet
            ELSE  Discard packet
CASE Encapsulating packet is v6
      CASE  Encapsulated packet is v4
      DO    Decapsulate 4/6, getting X4 and Y4
            IF    X4 = G4.. & TX6 = H6.(X4-G4).0/128
                  & Y4 NOT= G4..
            DO    Forward decapsulated packet
            ELSE  Discard packet
      CASE  Encapsulated packet is v6
      DO    Decapsulate 6/6, getting X6 and Y6
            IF    X6 = G6.. & TX6 = H6.(X6-G6).0/128
                  & [ Y6 NOT = G6..
            DO    Forward decapsulated packet
            ELSE  Discard packet

    CORE-SAM PROCESSING OF CORE-BOUND PACKET
```

Figure 6

```
CASE  Y4 NOT= G4..
DO    CASE  C4 NOT= nil
      DO    TY4 <- C4
            TX4 <- H4.(X4E-G4).0/32
            Encapsulate 4/4
      CASE  C4 = nil & C6 NOT= nil
      DO    TY6 <- C6
            TX6 <- H6.(X4E-G4).0/128
            Encapsulate 4/6
CASE  Y4 = G4..
DO    Discard packet
```

CORE-SAM PROCESSING OF AN IPV4 BRANCH-BOUND PACKET

Figure 7

```
CASE  Y6 NOT= G6..
DO    CASE  C4 NOT= nil
      DO    TY4 <- C4
            TX4 <- H4.(X6-G6).0/32
            Encapsulate 6/4
      CASE  C4 = nil & C6 NOT= nil
      DO    TY6 <- C6
            TX6 <- H6.(X6-G6).0/128
            Encapsulate 6/6
CASE  Y4 = G4..
DO    Discard packet
```

CORE-SAM PROCESSING OF AN IPV6 BRANCH-BOUND PACKET

Figure 8

4. Parameter values for ISATAP - 6to4 - 6rd

ISATAP [RFC4214], 6to4 [RFC3056], and 6rd [I-D a], are techniques that provide IPv6 connectivity via various IPv4 domains. They can be implemented as specific applications of the SAM architecture with the ad hoc parameter values shown in the following table.

	ISATAP	6to4	6rd
Branch domains	DS hosts	customer sites	customer sites
Transit domain	customer site	global IPv4 Internet *	ISP IPv4 infrastructure
Core domain	ISP IPv6 infrastructure	global IPv6 Internet	global IPv6 Internet
T6	Site v6 prefix	2002::/16	ISP v6 prefix **
H4	0.0.0.0/0	0.0.0.0/0	0.0.0.0/0
C4	CPE local Add.	192.88.99.1	192.88.99.2 ***
Ii length	32	32	32

* For full connectivity between 6to4 sites, the 2002 prefix must be routed from the global IPv6 Internet to the global IPv4 Internet
** A /28 prefix in the Iliad-Free deployment (initially a /32)
*** Value used in the Iliad-Free deployment. Any anycast address that is local to the ISP infrastructure can do.

SAM PARAMETERS OF EXISTING ENCAPSULATIONS OF IPv6 IN IPv4

Figure 9

5. Security considerations

With anti-spoofing checks in processing rules of Section 3, no security risk inherent to SAM has been identified.

6. IANA Considerations

To automate parameter settings of branch SAMs, DHCP and DHCPv6 option codes will have to be assigned.

7. Acknowledgements

So far, the SAM design has essentially been worked out by the author, with various intermediate stages like the so called Address Borrowing Protocol and the Global Address Protocol, without any sponsoring or company contract, and without seeking intellectual property protection. He therefore wishes to expresses its first acknowledgment to his wife: she accepted that traveling and other expenses be supported by the uni-personal enterprise of the author, the money of which cannot be distinguished from family money.

One important and recent progress of the approach has been the recognition that, with the flexibility of DHCP, no new protocol would be necessary to automate SAM parameter settings. Acknowledgment is due to Gabor Bajko and Teemu Savolainen for pointing it out at IETF 72.

8. Informative References

[I-D a] "IPv6 Rapid Deployment on IPv4 infrastructures (6rd) - Work in progress", September 2008.

[I-D b] "IPv4-IPv6 Coexistence Scenarios based on Stateless Address mapping - Work in progress", September 2008.

[RFC3056] Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds", RFC 3056, February 2001.

[RFC4214] Templin, F., Gleeson, T., Talwar, M., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", RFC 4214, October 2005.

Author's Address

Remi Despres
3 rue du President Wilson
Levallois,
France

Email: remi.despres@free.fr

Full Copyright Statement

Intellectual Property