                A Compact Representation of IPv6 Addresses

Status of this Memo

1. Abstract

   IPv6 addresses, being 128 bits long, need 32 characters to write in
   the general case, if standard hex representation, is used, plus more
   for any punctuation inserted (typically about another 7 characters,
   or 39 characters total).  This document specifies a more compact
   representation of IPv6 addresses, which permits encoding in a mere 20
   bytes.

2. Introduction

   It is always necessary to be able to write in characters the form of
   an address, though in actual use it is always carried in binary.  For
   IP version 4 (IP Classic) the well known dotted quad format is used.
   That is, 10.1.0.23 is one such address.  Each decimal integer
   represents a one octet of the 4 octet address, and consequently has a
   value between 0 and 255 (inclusive).  The written length of the
   address varies between 7 and 15 bytes.

   For IPv6 however, addresses are 16 octets long [IPv6], if the old
   standard form were to be used, addresses would be anywhere between 31
   and 63 bytes, which is, of course, untenable.

   Because of that, IPv6 had chosen to represent addresses using hex
   digits, and use only half as many punctuation characters, which will
   mean addresses of between 15 and 39 bytes, which is still quite long.
   Further, in an attempt to save more bytes, a special format was
   invented, in which a single run of zero octets can be dropped, the
   two adjacent punctuation characters indicate this has happened, the
   number of missing zeroes can be deduced from the fixed size of the
   address.

   In most cases, using genuine IPv6 addresses, one may expect the
   address as written to tend toward the upper limit of 39 octets, as
   long strings of zeroes are likely to be rare, and most of the other

groups of 4 hex digits are likely to be longer than a single non-zero
digit (just as MAC addresses typically have digits spread throughout
their length).

This document specifies a new encoding, which can always represent
any IPv6 address in 20 octets.  While longer than the shortest
possible representation of an IPv6 address, this is barely longer
than half the longest representation, and will typically be shorter
than the representation of most IPv6 addresses.

3. Current formats

[AddrSpec] specifies that the preferred text representation of IPv6
addresses is in one of three conventional forms.

The preferred form is x:x:x:x:x:x:x:x, where the 'x's are the
hexadecimal values of the eight 16-bit pieces of the address.

Examples:

     FEDC:BA98:7654:3210:FEDC:BA98:7654:3210  (39 characters)

     1080:0:0:0:8:800:200C:417A  (25 characters)

The second, or zero suppressed, form allows "::" to indicate multiple
groups of suppressed zeroes, hence:

     1080:0:0:0:8:800:200C:417A

may be represented as

     1080::8:800:200C:417A

a saving of just 5 characters from this typical address form, and
still leaving 21 characters.

In other cases the saving is more dramatic, in the extreme case, the
address:

     0:0:0:0:0:0:0:0

that is, the unspecified address, can be written as

     ::

This is just 2 characters, which is a considerable saving.  However
such cases will rarely be encountered.

The third possible form mixes the new IPv6 form with the old IPv4
form, and is intended mostly for transition, when IPv4 addresses are
embedded into IPv6 addresses.  These can be considerably longer than
the longest normal IPv6 representation, and will eventually be phased
out.  Consequently they will not be considered further here.

4. The New Encoding Format

   The new standard way of writing IPv6 addresses is to treat them as a
   128 bit integer, encode that in base 85 notation, then encode that
   using 85 ASCII characters.

4.1. Why 85?

   2^128 is 340282366920938463463374607431768211456.  85^20 is
   387595310845143558731231784820556640625, and thus in 20 digits of
   base 85 representation all possible 2^128 IPv6 addresses can clearly
   be encoded.

   84^20 is 305904398238499908683087849324518834176, clearly not
   sufficient, 21 characters would be needed to encode using base 84,
   this wastage of notational space cannot be tolerated.

   On the other hand, 94^19 is just
   3086236607781508759287901645469541990904, also insufficient to encode
   all 2^128 different IPv6 addresses, so 20 characters would be needed
   even with base 94 encoding.  As there are just 94 ASCII characters
   (excluding control characters, space, and del) base 94 is the largest
   reasonable value that can be used.  Even if space were allowed, base
   95 would still require 20 characters.

   Thus, any value between 85 and 94 inclusive could reasonably be
   chosen.  Selecting 85 allows the use of the smallest possible subset
   of the ASCII characters, enabling more characters to be retained for
   other uses, eg, to delimit the address.

4.2. The Character Set

   The character set to encode the 85 base85 digits, is defined to be,
   in ascending order:

               '0'..'9', 'A'..'Z', 'a'..'z', '!', '#', '$', '%', '&', '(',
               ')', '*', '+', '-', ';', '<', '=', '>', '?', '@', '^', '_',
               '`', '{', '|', '}', and '~'.

   This set has been chosen with considerable care.  From the 94
   printable ASCII characters, the following nine were omitted:

’"’ and "’", which allow the representation of IPv6 addresses to
be quoted in other environments where some of the characters in
the chosen character set may, unquoted, have other meanings.

’,’ to allow lists of IPv6 addresses to conveniently be written,
and ’.’ to allow an IPv6 address to end a sentence without
requiring it to be quoted.

’/’ so IPv6 addresses can be written in standard CIDR
address/length notation, and ’:’ because that causes problems when
used in mail headers and URLs.

’[’ and ’]’, so those can be used to delimit IPv6 addresses when
represented as text strings, as they often are for IPv4,

And last, ’\’, because it is often difficult to represent in a way
where it does not appear to be a quote character, including in the
source of this document.

5. Converting an IPv6 address to base 85.

The conversion process is a simple one of division, taking the
remainders at each step, and dividing the quotient again, then
reading up the page, as is done for any other base conversion.

For example, consider the address shown above

    1080:0:0:0:8:800:200C:417A

In decimal, considered as a 128 bit number, that is
21932261930451111902915077091070067066.

As we divide that successively by 85 the following remainders emerge:
51, 34, 65, 57, 58, 0, 75, 53, 37, 4, 19, 61, 31, 63, 12, 66, 46, 70,
68, 4.

Thus in base85 the address is:

    4-68-70-46-66-12-63-31-61-19-4-37-53-75-0-58-57-65-34-51.

Then, when encoded as specified above, this becomes:

    4)+k&C#VzJ4br>0wv%Yp

This procedure is trivially reversed to produce the binary form of
the address from textually encoded format.

6. Additional Benefit

   Apart from generally reducing the length of an IPv6 address when
   encode in a textual format, this scheme also has the benefit of
   returning IPv6 addresses to a fixed length representation, leading
   zeroes are never omitted, thus removing the ugly and awkward variable
   length representation that has previously been recommended.

7. Implementation Issues

   Many current processors do not find 128 bit integer arithmetic, as
   required for this technique, a trivial operation.  This is not
   considered a serious drawback in the representation, but a flaw of
   the processor designs.

   It may be expected that future processors will address this defect,
   quite possibly before any significant IPv6 deployment has been
   accomplished.

8. Security Considerations

   By encoding addresses in this form, it is less likely that a casual
   observer will be able to immediately detect the binary form of the
   address, and thus will find it harder to make immediate use of the
   address.  As IPv6 addresses are not intended to be learned by humans,
   one reason for which being that they are expected to alter in
   comparatively short timespan, by human perception, the somewhat
   challenging nature of the addresses is seen as a feature.

   Further, the appearance of the address, as if it may be random
   gibberish in a compressed file, makes it much harder to detect by a
   packet sniffer programmed to look for bypassing addresses.

9. References

    [IPv6]          Internet Protocol, Version 6 (IPv6) Specification,
                    S. Deering, R. Hinden, RFC 1883, January 4, 1996.

    [AddrSpec]      IP Version 6 Addressing Architecture,
                    R. Hinden, S. Deering, RFC 1884, January 4, 1996.

10. Author's Address

    Robert Elz
    Computer Science
    University of Melbourne
    Parkville, Victoria, 3052
    Australia

    EMail: kre@munnari.OZ.AU