Enhanced IPv4: IPv4 with 64 bit Address Space

This memo defines an experimental protocol for the Internet
community. It provides information of an experimental IPv4
extension, compatible with IPv4. This memo does not specify an
Internet standard of any kind. Discussion and suggestions for
improvement are requested. Distribution of this memo is unlimited.

1. Summary

This document is based on an article published February 2014 in IEEE
Computer magazine describing a solution to IPv4 address depletion,
Enhanced IP (EnIP) [1]. Some concepts will be repeated here for
clarity. Source code, documentation, and demonstration virtual
machines can be found at the following site:

http://www.enhancedip.org

An EnIP address is two IPv4 addresses concatenated together. It
looks like this:

    192.0.2.2.10.1.1.2

whereas an IPv6 address looks like this:

    2001:0101:c000:0202:0a01:0102::0

IPv6 faces a number of challenges to reach global deployment. Some
systems administrators have complained because IPv6 addresses are not
easy to remember. Also, IPv6 requires a complete upgrade to the
backbone of the network. It requires repeering via BGP on a large
scale. In contrast, EnIP packets can be sent over IPv4 backbones
without the need for repeering. Upgrades to CPE equipment (cable,
DSL, fiber modems) are another challenge for IPv6. EnIP also
requires upgrades to CPE equipment. There is a slight advantage for
EnIP because the EnIP CPE upgrade is much less complex. In a
demonstration system, a loadable kernel module is installed without
rebooting the CPE device. The demonstration kernel module is
presently 364 lines of code. This is simpler than the code required
to implement RFC 6204 (Basic Requirements for IPv6 Customer Edge
Routers). The complexity of RFC 6204 is such that getting an
accurate count of the lines of code is difficult.

EnIP does not replace IPv4 but builds upon it to maximize backward
compatibility. As a demonstration, EnIP is currently deployed via
Internet 2, between two nodes at the University of Maryland and the
University of Delaware. Applications such as ssh, http, and samba

operate between the two EnIP networks without modification of the
software or routers in the path.

2. Introduction

For various reasons, there is a large demand for IP addresses. It
would be useful to have a unique address for each Internet device to
allow, if desired, that any device could call another [2]. The
Internet of Things would also be able to make use of more routable
addresses if they were available. Currently, this is not possible
with IPv4. IPv6 has presented a potential solution to this problem
but has faced problems with global adoption. Carrier Grade Network
Address Translation and NAT (NAPT) have provided the solution thus
far.

Enhanced IP (EnIP) was designed to minimize impact on core and border
routers. DHCP, ARP, and existing IPv4 routing protocols are
compatible with EnIP. By leveraging private address space in a
similar manner of IP4+4 [3], EnIP increases available address space
by a factor of 17.9 million [1] or 17.9 million new addresses for
each current routable IP address.

This could allow the reassignment of small segments of unused address
blocks in /8 networks to registries with chronic shortages of IP
addresses.

EnIP packets carry additional address bits and state in an IP option,
eliminating routing table updates like IPv6. EnIP supports end-to-end
connectivity, a shortcoming of NAT, making it easier to implement
mobile networks. Host renumbering is also not required in EnIP as
has been the case with other 64-bit protocol proposals [3]. This is a
major topic of section 4.

Because current NATs are resource intensive, requiring that state be
maintained, this paper will call these NATs NATs, The stateless NAT
used in EnIP will be called an EnIP NAT.

2.1. Protocol Transitions

Several transitions in the Internet Protocol suite are discussed in
the IEEE paper [1]. These transitions include Network Control
Program (NCP) to TCP/IP, the evolution of IPv4, and IPv6. It also
refers to network address port translation (NAPT), usually called NAT
[4] Nat444 [5], Dual-Stack Lite [6], NAT64 [7] and RFC 6144 [8].

2.2. The Use of IP Options

EnIP uses IP options to increase address space. Over time, this
fixed-length IP option would need to be added to the fast path
implementations available on core routers [9]. Fast path

implementations are discussed by Hidell et. al. [10]. Fast path
allows core routers to optimize data paths to line cards based on
table lookups. The fast path does not usually process packets
containing IP options [9]. As a result, packets containing IP
options are forwarded to the slow path CPU for processing and
forwarding to the correct line card.

However, a very simple upgrade could be made to the core and border
routers to process EnIP packets in the fast path with the following
simple pseudo code [1]:

```
if (IgnoreOptions)
        FastPathOperations();
else if (IP_Option_Present AND Option_Byte1 == 0X9A)
        FastPathOperations() ;
else
        SlowPathOperations() ;
```

EnIP and IPv6 both require upgrades to the fast path implementations
of routers. EnIP's fast path upgrade has four advantages:

  (1)   The EnIP upgrade is the pseudo code above, instead of a large
        code base insertion.

  (2)   There is no equipment reconfiguration.

  (3)   Internet providers can independently upgrade their fast paths.

  (4)   Finally, there are no requirements to update the IPv4 routing
        tables.

2.3. Contents of this Paper

Section 3 describes EnIP. Section 4 is the heart of the paper,
describing the mechanics of EnIP. Section 5 describes the University
of Maryland and University of Delaware deployment. It also describes
tests between 2 nodes using EnIP and using normal IPv4 and gives the
results of the tests. The final section contains some concluding
remarks.

3. Introduction to EnIP (EnIP)

EnIP increases IP address space while maintaining compatibility with
existing IPv4 implementations. Current EnIP implementations use IP
option 26 to create a twelve byte extension to the IP header.

3.1. EnIP Addressing Example

The EnIP extension contains four bytes of overhead, and two four byte
fields used as additional storage for the EnIP source and destination
addresses. EnIP addresses are written as two IPv4 addresses

concatenated together. IPv6 and EnIP addressing schemes are shown
below:

        IPv6 address   2001:0101:c000:0202:0a01:0102::0
        EnIP address   192.0.2.2.10.1.1.2

In the above EnIP address, 192.0.2.2 is a public IPv4 address called
the site address; the 10.1.1.2 address is called the host address.
The host address is always a private IPv4 address assigned to a host
behind a lightweight, stateless EnIP-enabled NAT. It is the private
IPv4 address that identifies the host in the network behind that NAT
and is compatible with other normal IPv4 hosts. The site address
allows the packet to traverse public networks because its IPv4 Source
Host Number (or source address) is 192.0.2.2, a fully routable IPv4
address.

3.2. IPv4 Header with EnIP Option Header

The IPv4 header supporting EnIP is shown in Figure 1. It is an IPv4
header with additional option space used as follows:

Figure 1: EnIP Header

```
       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
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   1  |Version| IHL   |Type of Service|         Total Length          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   2  |        Identification         |Flags|     Fragment Offset     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   3  |  Time to Live |    Protocol   |         Header Checksum        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   4  |                     Source Host Number                        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   5  |                   Destination Host Number                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   6  |    EnIP ID    |    EnIP ID    |E|E|                           |
      |    (0X9A)     | Option Length |S|D|        (Reserved)         |
      |               |               |P|P|                           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   7  |                  Enhanced Source Address                      |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   8  |                Enhanced Destination Address                   |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.1. Fields of the EnIP Extension

The fields

Field	Bits	Description
Version	4	The Version field is identical to IPv4's.
IHL	4	Internet Header Length is identical to IPv4's and is set to the length of the EnIP header.
Type of Service	8	The ToS field is identical to IPv4's.
Total Length	16	The Total Length field is identical to that of IPv4 and is 32 octets.
Identification	16	The Identification field is identical to IPv4's.
Flags	3	The Flags field is identical to IPv4's.
Fragment Offset	13	The Fragment Offset field is identical to IPv4's.
Time to Live	8	The Time to Live field is identical to IPv4's.
Protocol	8	The Protocol field is identical to IPv4's.
Header Checksum	16	The Header Checksum field is to IPv4's.
EnIP specific		
Source Host Number	32	The Source Host Number field identifies the source host. The EnIP use is described in section 4
Destination Host Number	32	The Destination Host Number field identifies the destination host. The EnIP use is described in section 4
EnIP ID	8	The EnIP ID field equals to **0x9A** or **1 00 11010**. It's meaning is given at the end of this section.
EnIP Option Length	8	The EnIP Option Length field is **12** octets
ESP	1	This selector determines if an

		Enhanced Source Address is present
EDP	1	This selector determines if an
		EnIP Destination Address is present
Reserved	14	*Reserved for future use*
Enhanced Source Address	32	This is the host address used by
		EnIP. It is described in section 4
Enhanced Destination Address	32	This is the destination
		host address used by EnIP.
		It is described in section 4

NOTE: The protocol has 12 bytes of overhead per packet.

The meaning of the EnIP ID field, **1 00 11010** (0x9A hexadecimal)
is as follows:

If an EnIP packet traverses a router and must be fragmented because
of a link with a smaller MTU, the copy bit ensures that fragments
include the 12 byte IP option header in all fragmented packets. The
rest of the bits and their meaning are shown in Table 1.

Meaning of **1 00 11010** the **EnIP ID**		
1	**00**	**11010** (or 26 base 10)
copy bit set	**Class** is **Control**	**Option** is a new value

Table 1: EnIP Id Field Interpretation

Note that 26 is the IP option value used in the EnIP experiments.

4. EnIP Operation

The following scenarios describe the operation of EnIP. The scenarios
will show the NAT use case as well as the end to end 64 bit EnIP use
case.

4.1. IPv4 NAT Explanation using Figure 2

```
       +-----+                                          +-----+
       | IP1 |                                          | IP2 |
       +-----+                                          +-----+
       10.1.1.1                                         10.3.3.1
                          192.0.2.1  192.0.2.2
                          +--+-+            +-+--+
       +------+           | N1 |            | N2 |          +------+
       | EIP1 |           +-+--+            +-+--+          | EIP2 |
       +------+  10.1.1.254                   10.3.3.254    +------+
       10.1.1.2                                         10.3.3.2
```
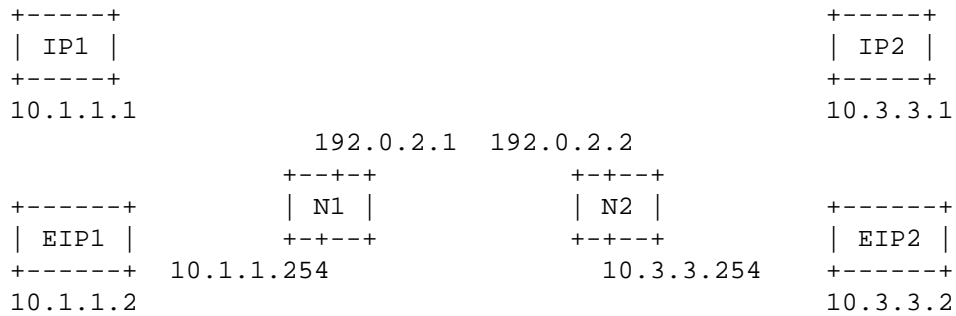
EIP1 and EIP2: machines supporting IPv4
                    and Enhanced IPv4
IP1 and IP2:   machines supporting only IPv4
N1 and N2:   NAT and EnIPNAT/translator devices

Figure 2 - Enhanced IP and Unenhanced IPv4

This section shows how two nodes, without EnIP, can communicate.
More complete explanations are available in reference 1 [1].

4.1.1. Typical Private IPv4 Host to a Typical Public IPv4 Host using NAT

IP1 (10.1.1.1), a typical IPv4 host, wants to reach N2 (192.0.2.2)
(See Figure 3). It uses NAT (as on Linux iptables) to masquerade as
the public IP address of N1 (192.0.2.1). N1 sets up a port
forwarding rule for IP1 for return packets from N2.

```
                192.0.2.1                 192.0.2.2
      +-----+           +--+-+            +-+--+
      | IP1 +--------+ N1 +------------+ N2 +
      +-----+           +-+--+            +--+-+
      10.1.1.1
```
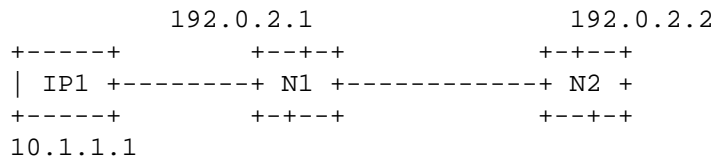
IP1:   machine supporting only IPv4
N1 and N2:   NAT and EnIPNAT/translator devices

**Figure 3 - Typical Private IPv4 Host to a
             Typical Public IPv4 Host NAT**

4.1.2. Typical Private IPv4 Host to a Typical Private IPv4 Host using NAT

The same IP1 (10.1.1.1) wants to reach tcp port 80 on IP2 (10.3.3.1),
another typical IPv4 host (see Figure 4). The packets from 10.1.1.1
use NAT on N1 to masquerade as source IP address 192.0.2.1. At N2
(192.0.2.2), the packets arrive. There they have a NAT port
forwarding rule setup on N2 to map tcp port 80 on 192.0.2.2 to

forward packets to the internal host IP2 (10.3.3.1).
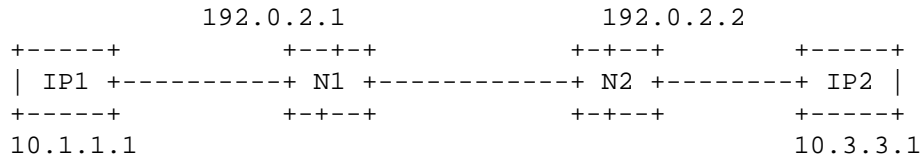
```
                  192.0.2.1                    192.0.2.2
      +-----+            +--+-+            +-+--+           +-----+
      | IP1 +----------+ N1 +------------+ N2 +--------+ IP2 |
      +-----+            +-+--+            +-+--+           +-----+
      10.1.1.1                                            10.3.3.1
```

IP1 and IP2:   machines supporting only IPv4
N1 and N2:   NAT and EnIP NAT/translator devices

Figure 4 - Typical Private IPv4 Host to a
Typical Private IPv4 Host using NAT

So packets move from IP1 to N1, taking N1's address as the source
address and move to N2. N2 has a mapping to IP2 and sends the packet
to IP2. When IP2 sends replies, the NATs used their table entries to
send the packet back to IP1, adjusting the addresses as necessary.

4.1.3. Private EnIP Host to a Typical Public IPv4 Host NAT

Suppose EIP1 (10.1.1.2) wants to reach N2 (192.0.2.2) as in Figure 5.
It is the same as an IPv4 node wanting to reach N2. So the
destination IP address EIP1 s used to communicate with N2
(192.0.2.2). The NAT device (N1) uses IPv4 NAT to translate the
source of the packets to come from 192.0.2.1. EIP1 behaves as though
it is an IPv4 host. It is fully compatible with normal IPv4
communications.

```
                  192.0.2.1                    192.0.2.2
      +------+           +--+-+            +-+--+
      | EIP1 +---------+ N1 +------------+ N2 |
      +------+           +-+--+            +-+--+
      10.1.1.2
```

EIP1:   machine supporting IPv4
                   and Enhanced IPv4
N1 and N2:   NAT and EnIP NAT/translator devices

Figure 5 - Private EnIP Host to a
Typical Public IPv4 Host using NAT

4.1.4. Private EnIP Host to a Typical Private IPv4 Host using NAT

EIP1 (10.1.1.2) initiates a tcp port 80 connection to IP2 (10.3.3.1)
(see Figure 6). Packets from EIP1 reach N1. As above, they are
masqueraded as IPv4 address 192.0.2.1. When the packets arrive at N2
(192.0.2.2) from N1 (192.0.2.1), there is a port forwarding entry for
port 80 setup to send the packets to IP2 (10.3.3.1).

```
                                                     +-----+
                         192.0.2.1  192.0.2.2       +----+ IP2 |
                        +--+-+              +-+--+    |   +-----+
         +------+    +----+ N1 +------------+ N2 +-----+   10.3.3.1
         | EIP1 +----+    +-+--+            +-+--+
         +------+
         10.1.1.2
```

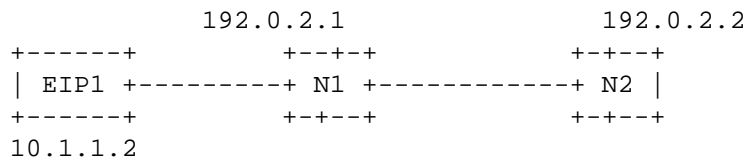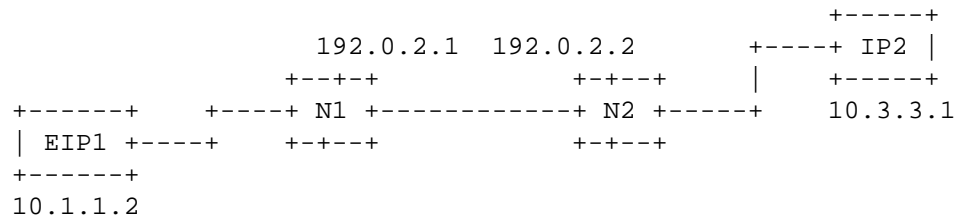|	
EIP1:	machine supporting IPv4 and Enhanced IPv4
IP2:	machine supporting only IPv4
N1 and N2:	NAT and EnIP NAT/translator devices

**Figure 6 - Private EnIP Host to a Typical
Private IPv4 Host using NAT**

**It is important to note that thus far
this paper has not demonstrated any usage of EnIP features,
just compatibility with current IPv4 implementations**

4.2. Enhanced IPv4 Explanation using an Example Figure 7

Suppose EIP1 wants to send packets to EIP2. Here both hosts are
running Enhanced IPv4 stacks and N1 and N2 support EnIP. Suppose EIP1
knows the address of EIP2 is 192.0.2.2.10.3.3.2. EIP1 knows its
internal IP address of 10.1.1.2 but is not aware of the external
address of N1 (192.0.2.1). The following is done (see Figure 7):
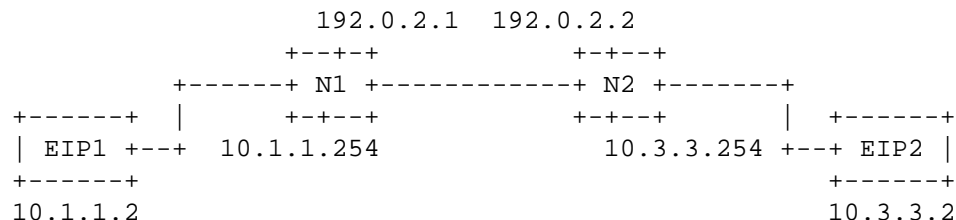
```
                         192.0.2.1  192.0.2.2
                        +--+-+              +-+--+
                 +------+ N1 +------------+ N2 +-------+
        +------+ |      +-+--+            +-+--+       | +------+
        | EIP1 +--+  10.1.1.254          10.3.3.254 +--+ EIP2 |
        +------+                                       +------+
        10.1.1.2                                       10.3.3.2
```

EIP1 and EIP2:	machines supporting IPv4 and Enhanced IPv4
N1 and N2:	NAT and EnIP NAT/translator devices

Figure 7 - Enhanced IP

4.2.1. EIP1 constructs the Packet

(1)   EIP1 sets the Source Host Number field (the source IPv4
       address) to 10.1.1.2;

   (2)   The EnIP ID field is set to 0X9A. the ESP bit is zeroed in the
          EnIP header.

   (3)   The Enhanced Source Address in the EnIP header is set to all
          ones since an EnIP source address is not currently present.

   (4)   The most significant 32 bits of the EnIP address is set by
          storing 192.0.2.2 in the IPv4 Destination Host Number.

   (5)   The least significant 32 bits of the EnIP address is set by
          storing 10.3.3.2 in the EnIP Destination Address field.

   (6)   Finally, EIP1 sets the EDP bit to 1.

4.2.2. EIP1 Transmits the Packet to N1

   The packet is routed to N1 using normal IPv4. N1 does the following:

   (1)   It notes the EnIP option (0X9A)is present.

   (2)   N1 reads 10.1.1.2 from the IPv4 Source Host Number field and
          places that in the Enhanced Source Address field. This field
          no longer contains 255.255.255.255.

   (3)   N1 sets the ESP bit to 1

   (4)   N1 makes 192.0.2.1 the IPv4 Source Host Number.

   (5)   N1 recomputes the IP checksum of the new packet. If the
          packet carries TCP or UDP, it recomputes these checksums as
          they have also changed.

4.2.3. Packet Arrives at N2 (192.0.2.2)

   When the packet Arrives at N2, N2 does the following:

   (1)   N2 identifies the packet as an EnIP packet (0X9A).

   (2)   It reads the Enhanced Destination Address, 10.3.3.2, and
          places this value as the IP header's Destination Host Number.

   (3)   N2 zeroes the EDP bit and the Enhanced Destination Address to
          zero.

   (4)   It recomputes the IP checksum. If the packet carries TCP or
          UDP, recomputes these checksums as they have changed as a
          result of a change to the IP destination address.

   (5)   N2 sends the packet to EIP2.

4.2.4. EIP2 Receives the Packet

   When the packet arrives at EIP2, EIP2 does the following:

   (1)   It computes the source address of the packet. The IPv4 Source
          Host Number (192.0.2.1) is concatenated with the Enhanced
          source address (10.1.1.2) The result is 192.0.2.1.10.1.1.2.

   (2)   The IPv4 Destination Host Number (address) is 10.3.3.2.

4.2.5. EIP2 Transmits a Packet to EIP1

   To construct a packet from EIP2 to EIP1, EIP2 does the following:

   (1)   EIP2 sets the Option ID field to 0X9A.

   (2)   It takes the IPv4 Source Host Number from the incoming packet,
          192.0.2.1, and sets it as the IPv4 Destination Host Number.

   (3)   It places the Enhanced Source Address (10.1.1.2) in the
          Enhanced Destination Address field.

   (4)   EIP2 sets the EDP field to 1 and the IPv4 Source Host Number
          to 10.3.3.2.

   (5)   It sets the Enhanced Source Address to all ones.

   (6)   It sets the ESP bit to 0.

4.2.6. Packet Arrives at N2

   When the packet Arrives at N2, N2 does the following:

   (1)   It writes 10.3.3.2 in the Enhanced Source Address field.

   (2)   The ESP bit is set to 1.

   (3)   N2 writes 192.0.2.2 in the IPv4 Source Host Number field and
          recomputes the IP checksum. If the packet carries TCP or UDP,
          it recomputes these checksums as well.

   (4)   N2 sends the packet to N1 (192.0.2.1).

4.2.7. Packet Arrives at N1

   When the packet arrives at N1, it does the following:

   (1)   N1 reads 10.1.1.2 from the Enhanced Destination Address

   (2)   It writes this value in the IPv4 Destination Host Number
          field.

   (3)   It zeroes the EDP bit and the Enhanced Destination Address
          field.

   (4)   N1 recomputes the IP checksum and if the protocol is TCP or
          UDP, recomputes these checksums as well.

   (5)   N1 sends the packet to EIP1.

4.3. Upgrading Servers to Support EnIP

   In order to maintain backward compatibility with old IPv4 systems, it
   is important that the EnIP Source Address is an allowed private
   address. Otherwise, that address becomes a routable address outside

of an autonomous system and there is a potential for routing
ambiguity.

With a kernel modification to support EnIP, an existing server
deployed using an IPv4 address can be upgraded. The example of a TCP
echo server [11] is shown in detail elsewhere [1]. However,
highlights are now presented here.

An echo server logs the source IP address of each data packet with
the *getpeername* function. However, the length of the IPv4 address
structure returned is currently 16 bytes for IPv4 (the size of a
struct sockaddr_in). For IPv6 the size of *struct sockaddr_in6* is 28
bytes. As an example of the Alpha implementation mentioned
previously, EnIP returns a new structure called *struct sockaddr_ein*
that is 26 bytes. It is necessary to use the length 26 to detect a
struct sockaddr_ein. This new *struct* has two values: sin addr1 and
sin addr2. These represent the IPv4 source address and the EnIP
Source Address. An implementation of *getpeername* could provide a
compatibility mode to treat EnIP addresses as IPv4 addresses. Once
the echo client software is upgraded, the *getpeername* implementation
could return *struct sockaddr_ein*.

4.4. DNS Operation

RFC 2928 provides 2001:0101 as the experimental IPv6 prefix [12].
EnIP lookups can use already standardized AAAA records with this
prefix. This prefix uses 32 bits of the 128 bit AAAA record. EnIP
uses only 64 bits of the remaining 96 bits. If 192.0.2.2.10.1.1.2 is
the EnIP address, the EnIP AAAA record is
2001:0101:c000:0202:0a01:0102::0. We call this an AA record. A new
record type is not needed and eliminates the need for DNS server
software upgrades on a massive scale.

4.5. Information Security

The same article shows the care taken to minimize changes in
IDS/Firewalls Operation. It also addresses the prevention of EnIP
NAT or hosts from forwarding illegal packets, and its operation with
most of IPSEC, except in the full tunnel mode AH+ESP scenario.

5. Tests and Comparisons of EnIP and Typical IPv4

The IEEE Computer Magazine article describes the two-node EnIP test
deployment over the Internet 2 network with the EnIP NATs. The
tests, along with their results, are given that compare typical IPv4
connections with EnIP connections [1]. EnIP was used across 7
routers on Internet 2 to demonstrate viability in environments that
need more addressing.

In brief, the test had a node at the University of Maryland running the Linux 2.6.38 kernel with patches to include the EnIP alpha code. Another similar node was setup at the University of Delaware. Each had an EnIP NAT. *http*, *samba*, and *ssh* were used without modification with Enhanced IP DNS calls used.

The results, in section 5.3 of the paper[1], show EnIP performance gains over traditional NAT, as there are no hash table lookups, as in NAT. It provided roughly a factor of 2 improvement.

EnIP has also had basic testing with a 4G Long Term Evolution (LTE) simulator along with an Android phone. However, these results have not been compiled.

6. Conclusion

This RFC describes the operation of IPv4 using IP options, called Enhanced IP or EnIP. EnIP provides a solution to IPv4 address scarcity. Because the EnIP design criteria is to extend IPv4 instead of replacing it, it reduces impact on routers and information security mechanisms already in place. The operation of two EnIP nodes at the University of Maryland and the University of Delaware, running http, samba, and ssh without modification of the software or routers in the path demonstrates the feasibility of EnIP on a small but realistic scale that spanned seven routers.

Tests were conducted and documented, that show expected performance improvement over the old NAT currently used, and the new EnIP NAT. It provided roughly a factor of 2 improvement [1].

7. References

[1] W. Chimiak, S. Patton, and S. Janansky, "Enhanced IP: IPv4 with 64-Bit Addresses," IEEE Computer Magazine, Vol. 47, Issue 2, pp. 62 - 69 , Feb 2014.

[2] G. Lee, et al., "The Internet of Things - Concept and Problem Statement," draft-lee-iot-problem-statement-05 (work in progress), July 2012.

[3] Z. Turanyi and A. Valko, "IPv4+4," Proceedings of the 10th IEEE International Conference on Network Protocols, 2002.

[4] T. E. Paul F. Tsuchiya, "Extending the ip Internet through address reuse," ACM SIGCOMM Computer Communication Review, vol. 23, pp. 16-33, Jan 1993.

[5] J. Yamaguchi, Y. Shirasaki, et al., "Nat444 addressing models," Request for Comments, draft-shirasaki-nat444-isp-shared-addr-08, Internet Engineering Task Force, 2012.

[6] A. Durand, R. Droms, J. Woodyatt, and Y. Lee, "Dual-Stack Lite
    Broadband Deployments Following IPv4 Exhaustion." RFC 6333
    (Proposed Standard), Aug. 2011.

[7] M. Bagnulo, P. Matthews, and I. van Beijnum, "Stateful NAT64:
    Network Address and Protocol Translation from IPv6 Clients to
    IPv4 Servers." RFC 6146 (Proposed Standard), Apr. 2011.

[8] F. Baker, X. Li, C. Bao, and K. Yin, "Framework for IPv4/IPv6
    Translation." RFC 6144 (Informational), Apr. 2011.

[9] J. Aweya, "IP router architecture: An overview," tech. rep.,
    University of Virginia, 1999.

[10]M. Hidell, P. Sjodin, and O. Hagsand, "Router architectures:
    Tutorial at Networking 2004." http://web.ict.kth.se/\
    ~mahidell/pubs/networking04_tutorial_final.pdf.

[11] J. Postel, "Echo Protocol." RFC 862 (Standard), May 1983.

[12] R. Hinden, S. Deering, R. Fink, and T. Hain, "Initial IPv6 Sub-
    TLA ID Assignments." RFC 2928 (Informational), Sept. 2000.

8. Acknowledgments

9. Author's Addresses

William J. Chimiak
Laboratory for Telecommunication Sciences
8080 Greenmead Drive
College Park, MD 20740
+1-301-422-5217

mailto:w.chimiak@ieee.org

Table of contents

EnIP Table of Contents