The IPtX Dynamic Host Configuration Protocol; DHCPvIPtX-MX

‘draft-terrell-iptx-mx-dhcp-specification-01’

Status of this Memo

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."

"This document may not be modified, and derivative works of it may not be created, except to publish it as an RFC and to translate it into languages other than English."

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.

Intellectual Property Rights (IPR) Statement

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.

Requirements Terminology

The keywords Must, Must Not, Required, Shall, Shall Not, Should, Should Not, Recommended, May, and Optional, when they appear in this document, are to be interpreted as described in [RFC-2119].

Conventions

Please note, the mathematical operators that cannot be represented in the 'txt' file format, which represent; the '^' Carrot sign for ‘NESTED’ Super-Script, and the ‘v’ sign is used for a ‘NESTED’ Sub-Script.

This Internet-Draft will expire on March 3rd, 2008.

E Terrell

Internet Draft

The IPtX-MX Dynamic Host Configuration Protocol

March 3rd, 2008
Abstract

This document defines the IPtX Specification for the 'Dynamic Host Configuration Protocol'; IPtX / IPtX-MX DHCP (DHCPvIPtX-MX), which provides Backwards Compatibility with the IPv4 Specification without compromise or change to current DHCP Server and Client Configuration and / or Operational requirements. And more importantly, because the IPtX / IPtX-MX Specification represents a 3 State Binary IP Addressing Specification, there are 2 IP Address Band Specifications; Mobile IP Address Pool and a Stationary IP Address Pool, with a 3 Dimensional Locator, which represents a 3 IP Address Coordinate System that uses an EMERGENCY Broadcast [e911] to establish a Synchronized LINK with 3 different [KNOWN] Router Locations and the MAC Address, to Triangulate the Location of any Node Connected to the Network -

[Given that - The 3 IP Address Coordinate System uses the CIDR Network Descriptors, '/0000:00', '+/0000:00', '-/0000:00' to differentiate the IP Address Broadcasting Node's Location to the 3 different [KNOWN] Router Locations, the differentiated IP Addresses however, must use the same IPtX-MX MAC Address to identify the Node's Hardware.]
Table of Contents

Abstract

Introduction

IANA Considerations

I. The DHCPv4 and DHCPv6 Header Design Specification
II. The IPv4 Compatibility DHCPvIPtX-MX 32 Bit Design Specification
III. The 'Preferred' DHCPvIPtX-MX 32 / 64 Bit Design Specification
IV. IPtX / IPtX-MX Mobile IP Addressing Specification
V. IPtX / IPtX-MX Subnet ID

Special IANA Considerations

Security Considerations

References

Work(s) in Progress

Normative References

Informative References
Introduction

The DHCPv4 Header, which was derived from the Bootp Protocol (RFC 951 Bootstrap Protocol), other than Commands, has not changed since 1985. And clearly, the purpose or functional use of DHCPv4 not is obsolete, because there are several viable reasons not to assign a Static IP Address to a Client. Especially when the Client is only a Guest of the Network. In other words, if any improvement in Performance or Use necessitates Change, then the DHCPvIPtX-MX Specification prescribes the logically viable reason(s) for making the changes.
IANA Consideration

I. The DHCPv4 and DHCPv6 Header Design Specification
## DHCPv4 32 Bit Header

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
<td>31</td>
<td></td>
</tr>
</tbody>
</table>

1. Opcode 8 Bits | Hardware type | Hardware address length | Hop count |
2. Transaction ID |
3. Number of seconds | Flags |
4. Client IP address |
5. Your IP address |
6. Server IP address |
7. Gateway IP address |
8. Client hardware address :: |
9. Server host name :: |
10. Boot filename :: |
11. Options :: |

---

E Terrell                                           Internet Draft                                                       6

The IPtX-MX Dynamic Host Configuration Protocol       March 3rd, 2008
IPv6 32 Bit Header

```
   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 2
   +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  1. |Ver|Traffic Class|       Flow Label       |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  2. |   Payload Length   |   Next Header   |   HOP LIMIT   |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  3. |   SOURCE ADDRESS   |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  4. |   SOURCE ADDRESS   |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  5. |   SOURCE ADDRESS   |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  6. |   SOURCE ADDRESS   |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  7. |   DESTINATION ADDRESS|
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  8. |   DESTINATION ADDRESS|
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  9. |   DESTINATION ADDRESS|
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
 10. |   DESTINATION ADDRESS|
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
 11. |   DATA               |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
 12. |   DATA               |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
 13. |   DATA               |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
 14. |   DATA               |
       +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Note: IPv6 Header Bit-Map Length = 14 x 4 Octets = 56 Octets
14 x 32 Bits = 56 Octets = IPv6 Header 448 Bits
DHCPv6 32 Bit Header

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Message | Data / Options* (variable length) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Data / Options Information Fields*

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| msg-type | transaction-id |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Options |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| (variable (16 octets) : :: : ) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| hop-count | link-address (16 octets) : :: : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| peer-address (16 octets) : :: : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| options (variable number and length) (16 octets) : :: : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| options (variable number and length) (16 octets) : :: : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
1 | hardware type (16 bits) |
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>time (32 bits)</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>link-layer address (variable length) (16 octets) ::::</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
</tbody>
</table>

2 | enterprise-number |
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>enterprise-number (contd)</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
</tbody>
</table>

3 | hardware type (16 bits) |
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>link-layer address (variable length) (16 octets) ::::</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
</tbody>
</table>

OPTION_IA_PD | option-length |
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IAID (4 octets)</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>T1</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>T2</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>IA_PD-options</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
</tbody>
</table>

OPTION_IAPREFIX | option-length |
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>preferred-lifetime</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>valid-lifetime</td>
</tr>
<tr>
<td>prefix-length</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>IAPREFIX-options</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
</tbody>
</table>
OPTION NIS_SERVERS | option-len | NIS server (IPv6 address) (16 octets) :::::

OPTION NIS_DOMAIN_NAME | option-len | nis-domain-name :::::

OPTION NISP_DOMAIN_NAME | option-len | nisp-domain-name :::::

OPTION_SNTP_SERVERS | option-len | SNTP server (IPv6 address) (16 octets) :::::

OPTION information-refresh-time

Additional Options :::::
| OPTION_BCMCS_SERVER_A | option-len |
| BCMCS Control server-1 (IPv6 address) (16 octets) :::: |
| BCMCS Control server-2 (IPv6 address) (16 octets) :::: |
| Additional Options :::: |

| OPTION_SUBSCRIBER_ID | option-len |
| subscriber-id |

| OPTION_FQDN | option-len |
| flags | domain-name (16 octets) :::: |

| OPTION_NEW_POSIX_TIMEZONE | option-len |
| TZ POSIX String (16 octets) :::: |

| OPTION_NEW_TZDB_TIMEZONE | option-len |
| TZ Name |
II. The IPtX / IPtX-MX 32/64 Bit Header Design Specification

IPtX / IPtX-MX 32 / 64 Bit Header

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
/ 32 Bit Header Scale /
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
/ 64 Bit Header Scale /

IPtX 32 / 64 Bit Header Information Fields

1 | IPtX Version = 2E21/53 = 21/53 Bits | Parity Notify Bit* |
| ++ + + + + + + + + + + + + + + + + + + + + + + + + + |
2 | Prefix | DESTINATION ADDRESS Exponent = 2E 14 / 46 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
3 | DESTINATION ADDRESS Exponential Decimal String = 2E22/54 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
4 | TTL / HOP LIMIT | Option Section FLAGS = 16 / 32 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
5 | IPtX Version = 2E21/53 = 21/53 Bits | Parity Notify Bit* |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
6 | Prefix | SOURCE ADDRESS Exponent = 2E 14 / 46 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
7 | SOURCE ADDRESS Exponential Decimal String = 2E 22 / 54 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
8 | 2E10.12 Bits = Option Section = 2E24.30 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |
9 | 2E10.12 Bits = DATA = 2E24.30 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + |

Note*: The 'Parity Notification Bit' defines the 'PREFIX' as either a Character (1 Bit), or an Integer (0 Bit).
DHCPvIPtX-MX 32 Bit Header

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Parity Bit | Message CALL Flags 32 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Parity Bit | Option CALL Flags 32 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Authorization Transaction ID = 2E24 = 24 Bits | TTL/HOP LIMIT |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| IPtX Version = 2E24 = 24 Bits | Parity Notify Bit* |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
5. | Prefix | Server ADDRESS Exponent = 2E14 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
6. | Server ADDRESS Exponential Decimal String = 2E22 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
7. | IPtX Version = 2E24 = 24 Bits | Parity Notify Bit* |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
8. | Prefix | Gateway ADDRESS Exponent = 2E14 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
9. | Gateway ADDRESS Exponential Decimal String = 2E22 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
10. | Message Section Exponent = 2E54 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
11. | Message Section Exponential Decimal String = 2E22 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
12. | Option Section Exponent = 2E54 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
13. | Option Section Exponential Decimal String = 2E22 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
14. | Requesting Client’s |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
15. | IPtX / IPtX-MX MAC Address and Hardware Info = 64 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
16. | 2EX.0000... = 2E4,194,304 - 2EQ.0000... = 2E4,194,304 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
17. | Client’s Network Account Info / DATA = 2E10.12 Bits |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
18. |---------------------------------------------------------|

Note: Client’s MAC Address is used as SOURCE Address when Requesting Client is on the Backbone of the DHCP Server’s Network.
In the structure of the Header noted above, for example, where each numbered Line defines a 32 Bit Field in a Transmission Sequence having the format of a Sentence, defines the 'Message and Option CALL Flag Fields' as a Set of Pointers interfacing with the 'Message and Options Section Fields', which defines a Set of Instructions ENCODED by the 'DCE Unit' that Performs a DHCP Task - as given below:

<table>
<thead>
<tr>
<th></th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>...</th>
<th>32</th>
<th>...</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Message Call Flags 64 Bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>+ + + + + + + + + + + + + + + + + + + + + + + + + + + + +</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

01 - SOLICIT.  
02 - ADVERTISE.  
03 - REQUEST.  
04 - CONFIRM.  
05 - RENEW.  
06 - REBIND.  
07 - REPLY.  
08 - RELEASE.  
09 - DECLINE.  
10 - RECONFIGURE.  
11 - INFORMATION-REQUEST.  
12 - RELAY-FORW.  
13 - RELAY-REPL.  
14 - Undefined.
| 01 - OPTION_CLIENTID. | 02 - OPTION_SERVERID. | 03 - OPTION_IA_NA. | 04 - OPTION_IA_TA. | 05 - OPTION_IAADDR. | 06 - OPTION_ORO. | 07 - OPTION_PREFERENCE. | 08 - OPTION_ELAPSED_TIME. |
33 - BCMCS Controller Domain Name list.

34 - BCMCS Controller IPtX address list.

35 - undefined.

36 - OPTION_GEOCONF_CIVIC.

37 - OPTION_REMOTE_ID.

38 - Relay Agent Subscriber-ID.

39 - FQDN, Fully Qualified Domain Name.

40 - OPTION_PANA_AGENT.

41 - OPTION_NEW_POSIX_TIMEZONE.

42 - OPTION_NEW_TZDB_TIMEZONE.

43 - OPTION_EPO.

44 - OPTION_LQ_QUERY.

45 - OPTION_CLIENT_DATA.

46 - OPTION_CLT_TIME.

47 - OPTION_LQ_RELAY_DATA.

48 - OPTION_LQ_CLIENT_LINK.

*** 49 - Undefined.

*** 50 - Undefined.

*** 51 - Undefined.

*** 52 - Undefined.

*** 53 - Undefined.

*** 54 - Undefined.

*** 55 - Undefined.

*** 56 - Undefined.

*** 57 - Undefined.

*** 58 - Undefined.

*** 59 - Undefined.

*** 60 - Undefined.

*** 61 - Undefined.

*** 62 - Undefined.

*** 63 - Undefined.

*** 64 - Undefined.
RECALL: \[ \text{Using the 'Data Compression' Ratio: '2EX : 1', or } 2^X \]

Example of Encoding the Bit-Map of the Equation for the 'Message and Option Section Fields'

Example of Text to encode...

'I went to the store today,.'

\[
\begin{align*}
I &= 01001001 = 73 = 2EX \sim 2E8 \\
\text{went} &= 0111011101100101011011110100 = 2,003,136,116 = 2EX \\
&\quad \sim 2E32 \\
\text{to} &= 01110100001101111 = 29,807 = 2EX \sim 2E16 \\
\text{the} &= 011101000110100001100101 = 7,628,901 = 2EX \sim 2E24 \\
\text{store} &= 01110011011011000110111101001001100101 = 495,874,699,877 \\
&\quad \sim 2EX \sim 2E40 \\
\text{today} &= 01110100011011110110010001000011101001001 = 500,085,055,865 \\
&\quad \sim 2EX \sim 2E40 \\
', ' &= 00101110 = 46 = 2EX \sim 2E8
\end{align*}
\]
The Equivalent Binary Numerical Conversion to be Transmitted:

'I went to the store today.'  'Iwenttothestoretoday.'

| 0100100100100000011101111011001 | 01001001011101110111011010101111 |
| 0101101110011101000010000000111 | 100111010001111010111101111110111 |
| 010001101110100000001110100001 | 0100011010000110010101111001101 |
| 1011000110010110010000011100111 | 110100001101111011110110010011001 |
| 01110100001101110110010011001 | 011101000110111110111010010011100 |
| 010010000001110100011011110111 | 01001110100010011110 |
| 01000110000010111011001001110 |

208 Bits  168 Bits

In other words, everything is counted, which includes the Blank SPACES Separating every word the sentence contains -

168 Bit Sentence '6 Words' = 'I went to the store today.'

Blank Space ' ' separating Words

00100000 = 8 BITS

Now... 'Taking it Away' yields;

'I went to the store today.'

\[
\begin{align*}
\text{I} & \quad = 01001001 \sim 2F8 = 73 \\
\text{Blank Space ' '} & \quad = 00100000 \sim 2F8 = 32 \\
\text{went} & \quad = 0111001110110010101011110011001 \sim 2H32 = 2,093,136,116 \\
\text{Blank Space ' '} & \quad = 00100000 \sim 2F8 = 32 \\
\text{to} & \quad = 01110010001110111110 \sim 2E16 = 29,807 \\
\text{Blank Space ' '} & \quad = 00100000 \sim 2F8 = 32 \\
\text{the} & \quad = 0111010010011000011001010 \sim 2E21 = 7,628,901 \\
\text{Blank Space ' '} & \quad = 00100000 \sim 2F8 = 32 \\
\text{store} & \quad = 011100111011010010001101110111101001 \sim 2E40 = 495,874,699,877 \\
\text{Blank Space ' '} & \quad = 00100000 \sim 2F8 = 32 \\
\text{today} & \quad = 0111010001101110110010100011011101100111010 \sim 2E40 = 500,085,055,065 \\
\end{align*}
\]

- No Blank Space Separating the 'WORD' and the 'Period'

\[
\text{.} = 00101111 \sim 2F8 = 46 \text{ (No Blank Space or 'Carriage Return' after the Period.)}
\]
And... 'Putting it Together' yields:

'I + went + to + the + store + today + .' 

I = 01001001 - 73 +
Blank Space = 00100000 = 32 +
went - 011101111101110101110111001110100 = 2,003,136,116 +
Blank Space = 00100000 = 32 +
to - 01110100011011110 = 29,807 +
Blank Space = 00100000 = 32 +
the - 0111010001101000011101010 = 7,628,901 +
Blank Space = 00100000 = 32 +
store - 01110011011101001101110100110011010 = 495,874,699,877 +
Blank Space = 00100000 = 32 +
today - 01110100011011110110011000011000011011110010 = 500,085,085,865 +
No Blank Space = Zero
'
'. ' - 00101110 - 46

Assembling (Joining) the Data Stream yields;

I(73) + Blank(32) + went(2,003,136,116) + Blank(32) + to(29,807) +
Blank(32) + the(7,628,901) + Blank(32) + store(495,874,699,877) +
Blank(32) + today(500,085,055,865) + Period(46)

= 73 + 32 + 2003136116 + 32 + 29807 + 32 + 7628901 + 32 +
495874699877 + 32 + 500085055865 + 46 = 60 Digit Number

= 733,220,031,361,163,229,807,327,628,913,324,958,746,
998,773,250,008,505,586,546 = 2E198.868003799...

= 2 E 198 . 868003799 ...
= 11 01000101 11000110 . 110011101111010101111111010111
= 2E198.868003799... = 48 Bit-Mapped Displacement
[ ' . ' = 8 Bits = 00101110 = 46 ]
48 - 56 Bits vs 208 Bits - 6 - 7 octets vs 26 octets
And this is equivalent to 26 Bytes, or approximately 208 Bits.

- Or -

\[
2E198.868003799... \sim 2E208 = \text{an approximate Bit-Mapped Displacement of 20 Bits (4 + 8 + 8). Or 20 Bits vs. 208 Bits; represents the difference between Bit-Mapping the 'Data Stream', as compared to Bit-Mapping the Equation of the 'Data Stream'.}
\]

Note: The Bit Mapped example used above follows from the Current Binary Translation, which includes the Askew Error!

And more importantly, the Compression Ratio becomes even greater, by some Exponential factor, as the amount of Data, which is to be Compressed increases. - e.g. 100Mbyte (800 MBit \sim 100,000,000 Octets) Document is compressed to \(2E800,000,000\), or \((4 + 8 + 30)\) 42 Bits \(\sim 6\) Octets \([\text{Approximating a '20,000,000 to 1' Bit-Mapped Compression Ratio}].\)

Furthermore, it should be readily concluded, since each of the numbered Line in the DHCPv1Ptx-MX Header defines a 32 Bit Field in a Transmission Sequence having the format of a Sentence, also defines the 'Message and Option CALL Flag Fields' as a Set of Pointers interfacing with the 'Message and Option Section Fields'. Where the 'Message and Options Section Fields' contains the Set of Instructions ENCODED by the 'DCE Unit', can Perform any assigned DHCP Task.
III. The IPtX / IPtX-MX 64 Bit Header Design Specification

**DHCPvIPtX-MX 64 Bit Header**

<table>
<thead>
<tr>
<th>0</th>
<th>2</th>
<th>4</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

1. | Message CALL Flags 64 Bits |

2. | Option CALL Flags 64 Bits |

3. | Authorization Transaction ID = 2E24 = 24 Bits | TTL/Hop LIMIT |

4. | IPtX Version = 2E24 = 24 Bits | Parity Notify Bit* |

5. | Prefix | Server ADDRESS Exponent = 2E46 Bits |

6. | Server ADDRESS Exponential Decimal String = 2E54 Bits |

7. | IPtX Version = 2E24 = 24 Bits | Parity Notify Bit* |

8. | Prefix | Server ADDRESS Exponent = 2E46 Bits |

9. | Server ADDRESS Exponential Decimal String = 2E54 Bits |

10. | Message Section Exponent = 2E18 Bits |

11. |

12. | Message Section Exponential Decimal String = 2E54 Bits |

13. | Option Section Exponent = 2E18 Bits |

14. |

15. | Option Section Exponential Decimal String = 2E54 Bits |

16. | IPtX / IPtX-MX MAC Address and Hardware Info = 64 Bits |

17. | Client’s Network Account Info / DATA = 2E10.12 Bits |

---

**Note:** Client’s MAC Address is used as SOURCE Address when Requesting Client is on the Backbone of the DHCP Server’s Network.
IV. IPtX / IPtX-MX Mobile IP Addressing Specification

There is a far greater growth potential, which expands the IPtX IP Addressing Protocol Family Specification, well beyond the results from the use of a Single IP Address Band Specification. That is, when adding the use of the 'Bar E' (Ě) notation to the 'DCE Unit' \{2ĚQ\} (given that the Members of the 'Real Number Set' represents every possible Numeral, denoting an Infinite Set), the IP Address Pool Total defined by the IPtX Specification increases to an amount equal to 'Bit-Mapping' every Element, or Member defined by the 'Set of Real' Numbers. In other words, the IPtX/IPtX-MX Specification defines a Logical 3 State Binary (2 Band) IP Addressing Specification, defining a Stationary and a Mobile IP Addressing Bands in a 3 Dimensional Space. – As given by;

\[
\begin{align*}
\text{Stationary Band} &= 0000:2\overline{E}.0000... \\
\text{Mobile Band} &= 0000:2\overline{E}.0000...
\end{align*}
\]

And more importantly, with each of these Address Band Specification there is a corresponding ‘MAC Address’ Specification – as given by;

\[
\begin{align*}
\text{Stationary Band 'MAC Address' Specification} &= 2\overline{E}.0000... \\
\text{Mobile Band 'MAC Address' Specification} &= 2\overline{E}.0000...
\end{align*}
\]
IPtx / IPtx-MX 32 / 64 Bit Mobile Header

0  1  2  3
1  2  3  4  5  6  7  8  9  0  1  2  3  4  5  6  7  8  9  0  1  2
/                                               / 32 Bit Header Scale
/ 0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5  6  7  8  9  0  1  2
/                                               / 64 Bit Header Scale
/                                               /
/ IPtx 32 / 64 Bit Header Information Fields
/
/
1 | IPtx Version = \text{2E}21/53 = 21/53 Bits | Parity Notify Bit* |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
2 | Prefix | DESTINATION ADDRESS Exponent = \text{2E} 14 / 46 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
3 | DESTINATION ADDRESS Exponential Decimal String = \text{2E}22/54 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
4 | TTL / HOP LIMIT | Option Section FLAGS = 16 / 32 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
5 | IPtx Version = \text{2E}21/53 = 21/53 Bits | Parity Notify Bit* |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
6 | Prefix | SOURCE ADDRESS Exponent = \text{2E} 14 / 46 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
7 | SOURCE ADDRESS Exponential Decimal String = \text{2E} 22 / 54 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
8 | \text{2E}10.12 Bits = Option Section = \text{2E}24.30 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
9 | \text{2E}10.12 Bits = DATA = \text{2E}24.30 Bits |
| + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |

Note*: The 'Parity Notification Bit' defines the 'PREFIX' as either a Character (1 Bit), or an Integer (0 Bit).
V. IPtX / IPtX-MX Subnet ID

It is extremely important to note, the general procedures for Subnetting, or allocating IP Address to a Sub-Division of the Network remain unchanged. That is, while the Subnet Mask has changed, Subnetting or allocating IP Address to a smaller Sub-Division of the Network remains unchanged, because it provides an easy method to account for every Node in the Structure of Network Hierarchal Scheme. However, because of the number of available IP Address in the IPtX / IPtX-MX IP Address Pool, Supernetting, as it were, is no longer a viable procedure or useful concept, especially since the IPtX / IPtX-MX Specification Sequentially counts every available IP Address.

Note: The Subnet Mask, now defined as the Subnet ID for the Stationary and Mobile IP Address Bands is given by;

\[
\text{Stationary Band Subnet ID} = 0000:\text{DCE E} \text{ Unit.0000...} \\
\text{Mobile Band Subnet ID} = \ 0000:\text{DCE E} \text{ Unit.0000...}
\]
Special IANA Considerations

Clearly, further exploitation of the 'DCE Unit'; since it has been shown that the Binary Exponential Base 2 Algorithm, '2EX', sequentially count using successive additions of "1's". The Preferred Design of the 'Message CALL Flags Field' and the 'Option CALL Flags Field' in the DHCPvIPtX-MX 32 / 64 Bit Header Specification, is given by:

Note: The using the 'DCE Unit' to redefine the 32 and 64 Bit Scales to represent a 'One to One' Correspondence with the Set of Integers, Bit-Maps each Flag as the Incremental Progression from 1 thru 32, or 64. And while this defines the Flags Progression in each Field, the Sequence Order of the Integer(s) representing the Bit Mapped Flag(s) is Function Governed. Hence, from pages 17 thru 20, the procedure for converting the first '3' 'Bit-Mapped Flag(s)', which represents the Sequence 1, 2, and 3, is given by:

Given that - the example of the Text to encode...

'1, 2, 3'  
'123'

0011000100110000100000001100 001100010011001000110011
1000101100001000000000110011

56 Bits 24 Bits

Recalling that everything is counted, which includes the 'COMMA(s)' and the 'BLANK SPACE(s)' Separating every Numerical the Sequence contains -

Hence, the Numerical Sequence represents a '56 Bit Sentence';

BLANK SPACE = 00100000 = 8 Bits
COMMA = 00101100 = 8 Bits

Note: The Bit Mapped example used above follows from the Current Binary Translation, which includes the Askew Error!
<table>
<thead>
<tr>
<th>'Taking it Away' yields:</th>
<th>'Putting it Together' yields:</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot; 1, 2, 3 &quot;</td>
<td>&quot; 1 + , + ' + 2 , + ' + 3 &quot;</td>
</tr>
</tbody>
</table>

| '1' = 00110001 = 49 = 2EX ~ 2E8 | '1' = 00110001 = 49 |
| 'Comma' = 00101100 = 44 = 2EX ~ 2E8 | 'Comma' = 00101100 = 44 |
| 'Space' = 00100000 = 32 = 2EX ~ 2E8 | 'Space' = 00100000 = 32 |
| '2' = 00110010 = 50 = 2EX ~ 2E8 | '2' = 00110010 = 50 |
| 'Comma' = 00101100 = 44 = 2EX ~ 2E8 | 'Comma' = 00101100 = 44 |
| 'Space' = 00100000 = 32 = 2EX ~ 2E8 | 'Space' = 00100000 = 32 |
| '3' = 00110011 = 51 = 2EX ~ 2E8 | '3' = 00110011 = 51 |

**Note:**

Encoding and Decoding the 'Binary Assembler', '2EX', representing the 'Assembled Data Stream' defining the 'DCE Unit', uses the "Punctuation" (Semi Colon, Commas, Spaces, Carriage Return... etc - in particular, the 'Blank Space') as 8 Bit HOOKS, to Decode or Encoded the 'Set of Instructions' - Given that; an individual Character or Numeral equals 8 Bits (2E8, or 28), where the HOOKS are used to define the Boundary Length of A Word or Numerical Sequence that is equal to 1 or more 8 Bit Octets, and contained in the Sentence defining any combination of Characters, Words, Numerals, and / or Punctuations. And clearly, once the 'Blank Spaces' has been identified, Decoding in a Right to Left or Left to Right 8 Bit Pattern would easily identify the remaining 8 Bit (Individual) Characters, Digits, and Punctuation(s) the Sentence contains.
And Assembling or Joining the Number Stream, \( '1, 2, 3' \), yields:

\[
1(49) + \text{Comma}(44) + \text{Space}(32) + 2(50) + \text{Comma}(44) + \text{Space}(32) + 3(51)
\]

\[
= 49 + 44 + 32 + 50 + 44 + 32 + 51
\]

\[
= 49,443,250,443,251
\]

\[
= 14 \text{ Digit Number}
\]

\[
\sim 2E45.4598888888...
\]

\[
= 2 \quad E \quad 45 \quad . \quad 4598888888 \quad ...
\]

\[
= 11 \quad 01000101 \quad 00101101 \quad 1000100100001111011110011111011000
\]

\[
= 2E45.4598888888... \quad = \quad 51 \text{ Bit-Mapped Displacement}
\]
DHCPvIPtX-MX 32 Bit Header

0   1   2   3
-------
1 2 3 4 5 6 7 8 9 0 1 2
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
1.   | Message CALL Flags = 2E10.12 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
2.   | Option CALL Flags = 2E10.12 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++

3.   | Authorization Transaction ID = 2E24 = 24 Bits| TTL/HOP LIMIT
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
4.   | IPtX Version = 2E24 = 24 Bits          | Parity Notify Bit*
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
5.   | Prefix | Server ADDRESS Exponent = 2E14 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
6.   | Server ADDRESS Exponential Decimal String = 2E22 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
7.   | IPtX Version = 2E24 = 24 Bits          | Parity Notify Bit*
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
8.   | Prefix | Gateway ADDRESS Exponent = 2E14 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
9.   | Gateway ADDRESS Exponential Decimal String = 2E22 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
10.  | Message Section Exponent = 2E54 Bits
11.  |
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
12.  | Message Section Exponential Decimal String = 2E22 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
13.  | Option Section Exponent = 2E54 Bits
14.  |
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
15.  | Option Section Exponential Decimal String = 2E22 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
16.  | Requesting Client’s
| IPtX / IPtX-MX MAC Address and Hardware Info = 64 Bits
17.  | 2EX.0000... = 2E4,194,304 - 2EQ.0000... = 2E4,194,304
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++
18.  | Client’s Network Account Info / DATA = 2E10.12 Bits
|++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ ++ +++

---

Note: Client’s MAC Address is used as SOURCE Address when Requesting Client is on the Backbone of the DHCP Server’s Network.
DHCPvIPtX-MX 64 Bit Header

<table>
<thead>
<tr>
<th>0</th>
<th>2</th>
<th>4</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. | Message CALL Flags = \(2^{E24}.30\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
2. | Option CALL Flags = \(2^{E24}.30\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
3. | Authorization Transaction ID = \(2^{E24} = 24\) Bits | TTL/HOP LIMIT |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
4. | IPtX Version = \(2^{E24} = 24\) Bits | Parity Notify Bit* |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
5. | Prefix | Server ADDRESS Exponent = \(2^{E46}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
6. | Server ADDRESS Exponential Decimal String = \(2^{E54}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
7. | IPtX Version = \(2^{E24} = 24\) Bits | Parity Notify Bit* |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
8. | Prefix | Server ADDRESS Exponent = \(2^{E46}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
9. | Server ADDRESS Exponential Decimal String = \(2^{E54}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
10. | Message Section Exponent = \(2^{E118}\) Bits |
11. |
12. | Message Section Exponential Decimal String = \(2^{E54}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
13. | Option Section Exponent = \(2^{E118}\) Bits |
14. |
15. | Option Section Exponential Decimal String = \(2^{E54}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
   | Requesting Client’s |
16. | IPtX / IPtX-MX MAC Address and Hardware Info = 64 Bits |
   | 2EX.0000... = \(2^{E4},194,304\) - \(2^{E4},0000... = 2^{E4},194,304\) |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |
17. | Client’s Network Account Info / DATA = \(2^{E10.12}\) Bits |
   | + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + |

**Note:** Client’s MAC Address is used as SOURCE Address when Requesting Client is on the Backbone of the DHCP Server’s Network.
Example of a 3D Grid Locator Scheme using a 3 IP Address Coordinate System in conjunction with a 'Longitude and Latitude' Rectangular Grid Overlay -

Noting more specifically that:

IANA EMERGENCY BROADCAST IP ADDRESS PROTOCOL

001-256 | 001-256:| All: | - - - - e .911 | All | IANA/Emer | 7/07

Representing on the Backbone, the IPtX-MX IP Address Mask given by:

0000:2E911.0000...

Prefix: | Zone IP: | IP Area Code: | Emergency Response Address | /XA

0000: 2 E 911 . 0000...
11111111 11 11001010 1110001111 . 1111...

Note: The Exponential Decimal String, '1111...', is used to Derive / Assign the additional IP Addresses on / for the 'e911 Emergency Response Network'.
Note - IANA \ IEEE Special Consideration -

- The Method used for Electronic Signal Propagation to Distinguish between Zero = ‘0’, and Binary 1 = ‘00’ -

1) A Single Position Binary 2 State Switch - Single Position 2 State Switch [X,Y] - Yielding: |X| = 0 or |1| = 1

   a) Single Position / 2 State Switch - True Or False

   b) 2 States defines the Choice of either a ‘0’ or a ‘1’

   c) ‘0’ defines a No Electronic Signal Pattern ‘*‘ = 0

   d) ‘1’ defines a True electronic Signal Pattern ‘1’ = 1

2) A Double Position Binary 3 State Switch - Double Position 3 State Switch [X,Y,Z] - Yielding: |X| = 0, |0,0| = 1, |0,1| = 2, |1,0| = 3, and |1,1| = 4

   a) Double Position / 3 State Switch - True, False, or No Response

   b) 3 States defines the Choice or Combination of either ‘*0’, ‘00’, ‘01’

   c) ‘0’ defines No Electronic Signal Pattern ‘*‘ = 0

   d) ‘00’ defines a True electronic Signal Pattern ‘00’ = 1

   e) ‘01’ defines a True electronic Signal Pattern ‘01’ = 2
Security Considerations

There are No Security Considerations presented in this document.
Work(s) in Progress;

These drafts represent the twelve chapters of the Networking Bible, designing a Network IP Addressing Specification that maintains a 100 Percent backward compatibility with the IPv4 Specification. In other words, this is a design specification developed from the Theory of the Expansion of the IPv4 IP Addressing Specification, which allowed the representation of the Network for the entire World on paper, and the possibility of an Infinite IP Address Pool. Nevertheless, the Internet-Drafts listed below, “Cited as Work(s) in Progress”, explain the design Specification for the development of the IPTX (IP Telecommunications Specification) Protocol Addressing System and the correction of the Mathematical Error in the Binary System.

Computer Science / Internet Technology:

   (Foundational Theory for the New IPTX family IP Addressing Specification, and the Binary Enumeration error discovery after the correction.) - "Work(s) in Progress"

   (The 2nd proof for the existence of another Binary System, resulting from the Error Correction.)
   - "Work(s) in Progress"

   (Argument against the Machine dependant IPv6 deployment.)
   - "Work(s) in Progress"

   (The foundation of the New IPTX Addressing Spec compared to the Telephone Numbering System.)
   - "Work(s) in Progress"

   (The IPTX Addressing Specification Address Space / IP Address Allocation Table; establishes the visual perspective that actually represents Networking Schematic Networking the entire World on Paper.) - "Work(s) in Progress"

   (Re-Defines CIDR) [Classes Inter-Domain Routing Architecture] and introduces the Network Descriptor for the IPTX Addressing Standard.) - "Work(s) in Progress"

   (The 3rd Proof for the New Binary System, correcting the error in Binary Enumeration.)
   - "Work(s) in Progress"

   (Defining the GWEBS – The Global Wide Emergency Broadcast System)
   - "Work(s) in Progress"

   (The development of the DHCP {Dynamic Host Configuration Protocol} for the IPTX IPSpec)
   - "Work(s) in Progress"
   (Derived the Binary System from the proof of "Fermat's Last Theorem", and Developed the Ternary
   Logic for the Binary System) - "Work(s) in Progress"

   - "Work(s) in Progress"
   (An application of Quantum Scale Theory, the $2^x : 1$ Compression Ratio, the Expansion derived from
   the 'CIDR Network Descriptor, and the Mathematics of Quantification provided the foundation for
   the development of the "Intelligent Quantum Tunneling Worm Protocol"; A Routable Mathematical
   Exponential Expression, Backend IP Addressing Protocol that provides an (nearly) Unlimited IP
   Address Space using the Compression Ratio $2^x : 1$.)

   (The development of the IPtX / IPtX-MX DNS {Domain Name Service} for IPtX IP
   Addressing Spec) 'Work(s) in Progress'

   (The development of the IPtX / IPtX-MX DHCP {Dynamic Host Configuration Protocol} for
   IPtX IP Addressing Spec) 'Work(s) in Progress'

Note: These Drafts has Expired at www.ietf.org Web Site. However, you can still find copies posted
at Web Sites all over the World. {Suggestion; Perform Internet search using “Yahoo”
or “Google”, Key word: “ETT-R&D Publications”}. 
Normative References:

Pure Mathematics:

1. The Proof of Fermat’s Last Theorem; The Revolution in Mathematical Thought {Nov 1979}
   Outlines the significance of the need for a thorough understanding of the Concept of
   Quantification and the Concept of the Common Coefficient. These principles, as well many others,
   were found to maintain an unyielding importance in the Logical Analysis of Exponential
   Equations in Number Theory.

2. The Rudiments of Finite Algebra; The Results of Quantification {July 1983}
   Demonstrates the use of the Exponent in Logical Analysis, not only of the Pure Arithmetic
   Functions of Number Theory, but Pure Logic as well. Where the Exponent was utilized in the
   Logical Expansion of the underlining concepts of Set Theory and the Field Postulates. The results
   yield another Distributive Property that is Conditional, which supports the existence of a Finite
   Field (i.e. Distributive Law for Exponential Functions) and emphasized the possibility of an
   Alternate View of the Entire Mathematical field.

3. The Rudiments of Finite Geometry; The Results of Quantification {June 2003}
   Building upon the preceding works from which the Mathematics of Quantification was derived.
   Where by it was logically concluded that there existed only 2 mathematical operations; Addition
   and Subtraction. In other words, the objectives this treatise maintained, which was derived from
   the foundation of the Mathematics of Quantification; involves not only the clarification of the
   misconceptions concerning Euclid’s Fifth Postulate, and the logical foundation of his work, or the
   existence of ‘Infinity in a Closed Bound Finite Space’. But, the logical derivation of the
   Foundational Principles that are consistence with the foundation presented by Euclid, which would
   establish the logical format for the Unification of all the Geometries presently existing.

4. The Rudiments of Finite Trigonometry; The Results of Quantification {July 2004}
   The development of the concepts for Finite Trigonometry from the combined foundations derived
   from numbers 3 and 5, and the Mathematics of Quantification.

5. The Mathematics of Quantification and the Metamorphosis of \( \pi : \tau \) {October 2004}
   The logical derivation of the exact relationship between the Circumference and the Diameter of
   the Circle, which defines the measurement of the exact length of the Circle’s Circumference, \( \tau \)
   when the Radius is equal to ‘1’.

6. Squaring the Circle? First! What is the Circle's Area? {January 2005}
   The Rhind Papyrus Tale, and the 10,000 year old quest involving “Squaring the Circle”;
   Derivation of the equation resolving the Area of the Circle. An illusion perplexing the Sight
   and Mind of the greatest mathematicians for about 10,000 years, which maintains an elementary
   algebraic solution: \( (\pi r^2) ÷ 2) = \text{Area of Circle} \).

Physics:

7. The Mathematics of Quantification & The Rudiments of Finite Physics
   The Analysis of Newton’s Laws of Motion…the Graviton’ {December 2004}
   Through the use of Finite Algebra, Geometry, Trigonometry, and # 5, investigation of the
   Laws of Classical Physics were found to be erroneous. This allowed the presentation of the
   initial work, which correct the flaws in Classical Physics, and establishes the foundation upon
   which there exist the possibility of a Grand Unified Field Theory for the Natural Sciences.
Informative References

1. G Boole (Dover publication, 1958) "An Investigation of The Laws of Thought" On which is founded The Mathematical Theories of Logic and Probabilities; and the Logic of Computer Mathematics.


3. R Carnap (Dover Publications, 1958) "Introduction to Symbolic Logic and its Applications"

4. Regis Desmeules (Cisco Press, April 24, 2003) "Cisco Self-Study: Implementing Cisco IPv6 Networks"

5. Gary C. Kessler (Auerbach Press, August 1997) "Handbook on Local Area Networks"

6. R. Hinden (Nokia) and S. Deering (Cisco Systems) RFC 2373 - "IP Version 6 Addressing Architecture"


10. DHCP Implementation and Security RFCs: 2939, 3004, 3011, 3046, 3118, 3203, 3256, 3361, 3396, 3397, 3442, 3456, 3495, 3527, 3594, 3634, 3679, 3825, 3925, 3942, 3993, 4014, 4030, 4039, 4174, 4280, 4361, 4388, 4390, 4578, 1541, 2489, 3115, 3315, 3319, 3646, 3633, 3898, 4075, 4242, 4280, 4776, 2855, 1542, 1534, 2131, 4361, 2132, 3942, 2485, 2563, 2610, 2855, 2937, 4649, 4580, 4704, 4833, 3315, 4361, 3319, 3633, 3646, 3736, 3898, 4075, 4076, 4280, and 4339.
Author:

Eugene Terrell
Principle Director
Research & Development

Engineering Theoretical Technologies
Research & Development Publications
(ETT-R&D Publications)

3312 64th Avenue Place
Oakland, CA. 94605
Voice: 510-636-9885
E-Mail: eterrell00@netzero.net

"This work is Dedicated to my first and only child, 'Princess Yahnay', because she is the gift of Dreams, the true treasure of my reality, and the 'Princess of the Universe'. (E.T. 2006)"