Stateless/Partial-state 1:N Network Address and Protocol Translation between IPv4 and IPv6 Nodes

X. Li, C. Bao, C. Metz
2010-03-25
Outline

• Introduction
• Stateless/Partial-state 1:N translation
  – Concept
  – Scenarios supported
  – Comparisons
  – Why call it partial-state
  – Implementation
• Deployment Considerations
• Example
Introduction

- **Stateless** [draft-ietf-behave-v6v4-xlate], etc
  - Support both IPv6 initiated and IPv4 initiated communications
  - Can not use IPv4 addresses effectively

- **Stateful** [draft-ietf-behave-v6v4-xlate-stateful], etc
  - Use IPv4 addresses effectively
  - Only support IPv6 initiated communication
  - Need to maintain states

- **Partial-state 1:N translation**
  - Support both IPv6 initiated and IPv4 initiated communications
  - Use IPv4 addresses effectively
  - Less state and complexity than full-blown stateful
  - Require less work to log translation bindings
Scenarios supported

Scenario 1  “an IPv6 network to the IPv4 Internet”
Scenario 2  “the IPv4 Internet to an IPv6 network”

Scenario 3  “an IPv4 network to the IPv6 Internet”
Scenario 4  “the IPv6 Internet to an IPv4 network”

Scenario 5  “an IPv6 network to an IPv4 network”
Scenario 6  “an IPv4 network to an IPv6 network”

Scenario 7  “the IPv6 Internet to the IPv4 Internet”
Scenario 8  “the IPv4 Internet to the IPv6 Internet”
Concept of 1:N translation (1)

- Port range
  - $P_0 \cap P_K = \varnothing$
  - $P_0 \cup P_1 \cup \ldots \cup P_K = I$

- If $R=256$, an IPv4 /24 can support 65,000+ IPv6 hosts
Concept of 1:N translation (2)

- **Algorithm**
  - For host K, the allowed port number (P) are \( P=j \times N + K \) (\( j=0, 1, ..., N-1 \))
  - For the destination port number (P), the packets will be sent to host \( K=(P \mod N) \) (% is the Modulus Operator)

- **Server port considerations**
  - For example, relaying HTTP from IPv4 to IPv6 can be used [draft-wing-behave-http-46-relay-02]
Comparisons (1)

• **NAT64**
  - Stateful
  - Only supports IPv6 initiated communication
  - Public IPv4 addresses can be shared dynamically by several IPv6 hosts
  - Use any IPv6 address

• **1:N Xlate**
  - Stateless/partial-state
  - Supports both IPv4 and IPv6 initiated communications
  - Public IPv4 addresses can be shared by several predefined IPv6 hosts (less efficiency)
  - Use extended IPv4-translatable address
Comparisons (2)

Address and Port: Algorithm based


Address: algorithm based; Port: state database if …

58.200.192.10#1024 ↔ 2001:da9:ff3a:c8c0:a00:0000:100:0#2000
58.200.192.10#1025 ↔ 2001:da9:ff3a:c8c0:a00:0000:100:1#2000
58.200.192.10#1026 ↔ 2001:da9:ff3a:c8c0:a00:0000:100:2#2000

Address and Port: State database

202.38.102.1#2000 ↔ 2001:da8::100#3000
202.38.102.1#2001 ↔ 2001:da8::101#3000
202.38.102.1#2002 ↔ 2001:da8::200#3000
Why call it partial state

- Stateless translation
  - 1:1 address translation
  - 1:N address translation with modified IPv6 end systems

- Partial-state translation
  - The address mapping is fully algorithm based. The states are used for port number mapping only.
  - No session table created if the source port number from IPv6 to IPv4 is in the range defined by the extended IPv4-translatable address. If application has port restrictions (e.g. P, P+1), the state of port mapping remains.
  - For the destination port number of the packet from the IPv4 to IPv6, no session table is created

- Full-state translation (stateful)
  - End system can use any IPv6 addresses
### Implementation (1)

**Address format**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>/32</td>
<td>prefix</td>
<td>IPv4(32)</td>
<td>u/g</td>
<td>suffix</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>/40</td>
<td>prefix</td>
<td>IPv4(24)</td>
<td>u/g</td>
<td>IPv4(8)</td>
<td>suffix</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>/48</td>
<td>prefix</td>
<td>IPv4(16)</td>
<td>u/g</td>
<td>IPv4(16)</td>
<td>suffix</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>/56</td>
<td>prefix</td>
<td>IPv4(8)</td>
<td>u/g</td>
<td>IPv4(24)</td>
<td>suffix</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>/64</td>
<td>prefix</td>
<td>IPv4(8)</td>
<td>u/g</td>
<td>IPv4(32)</td>
<td>suffix</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>/96</td>
<td>prefix</td>
<td>IPv4(32)</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>

### IPv4-converted address

### IPv4-translatable address

- [I-D.ietf-behave-v6v4-address-format] extensions
Implementation (2)

Port coding

<table>
<thead>
<tr>
<th>(4 bits)</th>
<th>Index Range (12 bits)</th>
<th>Multx ratio</th>
<th># of Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000-000</td>
<td>1</td>
<td>65,536</td>
</tr>
<tr>
<td>1</td>
<td>000-001</td>
<td>2</td>
<td>32,768</td>
</tr>
<tr>
<td>2</td>
<td>000-003</td>
<td>4</td>
<td>16,384</td>
</tr>
<tr>
<td>3</td>
<td>000-007</td>
<td>8</td>
<td>8,192</td>
</tr>
<tr>
<td>4</td>
<td>000-00f</td>
<td>16</td>
<td>4,096</td>
</tr>
<tr>
<td>5</td>
<td>000-01f</td>
<td>32</td>
<td>2,048</td>
</tr>
<tr>
<td>6</td>
<td>000-03f</td>
<td>64</td>
<td>1,024</td>
</tr>
<tr>
<td>7</td>
<td>000-07f</td>
<td>128</td>
<td>512</td>
</tr>
<tr>
<td>8</td>
<td>000-0ff</td>
<td>256</td>
<td>256</td>
</tr>
<tr>
<td>9</td>
<td>000-1ff</td>
<td>512</td>
<td>128</td>
</tr>
<tr>
<td>A</td>
<td>000-3ff</td>
<td>1,024</td>
<td>64</td>
</tr>
<tr>
<td>B</td>
<td>000-7ff</td>
<td>2,048</td>
<td>32</td>
</tr>
<tr>
<td>C</td>
<td>000-fff</td>
<td>4,096</td>
<td>16</td>
</tr>
</tbody>
</table>

- Use 16 bits to encode the port number range
  - 4 bits: multiplexing ratio
  - 12 bits: the host index
## Implementation (3)

### Algorithm

<table>
<thead>
<tr>
<th>Source port</th>
<th>Destination port</th>
</tr>
</thead>
<tbody>
<tr>
<td>In defined range</td>
<td>Entry exists</td>
</tr>
<tr>
<td>No in defined range</td>
<td>Copy the port number.</td>
</tr>
</tbody>
</table>

**From IPv6 to IPv4**
- Copy the port number.
- If the entry exists, use it to map the port number.
- If the entry does not exist, create the entry (time out for UDP or state machine for TCP) and use it to map the port number.
- Copy the port number.

**From IPv4 to IPv6**
- Copy the port number.
- Copy the port number.
- Use it to map the port number.
- Copy the port number.
Deployment Considerations

- **Using Modified IPv6 Hosts in an IPv6 Network**
  - Stateless 1:N XLATE
- **Using Unmodified IPv6 Hosts in an IPv6 Network**
  - Partial-state 1:N XLATE
- **Using mixed unmodified and modified IPv6 Hosts**
  - Partial-state 1:N XLATE
Examples

- Use old address format
  - 2001:da8:ff00::/40 as prefix
  - No U-octet
  - Port coding is in last 32 bits ratio:index

http://www.ivi2.org/demo2.html
Example (server)

- Host C1 (125.34.46.137) in the IPv4 Internet initiates communication with IPv6 end system Host0.
  - On the IPv4 Internet
    - Src#p= 125.34.46.137:1856 (random port)
    - Dst#p= 58.200.192.10:4096 (server port)
  - On translator
    - Src#p= [2001:DA9:FF7d:222e:8900::]:1856 (random port)
    - Dst#p= [2001:DA9:FF3A:C8C0:A00:0:100:0]:4096 (server port)
  - On an IPv6 network
    - Src#p= [2001:DA9:FF7d:222e:8900::]:1856 (random port)
    - Dst#p= [2001:DA9:FF3A:C8C0:A00:0:100:0]:4096 (server port)
Example (client)

• An IPv6 end system Host0 initiates communication with Host S2 (http://202.38.105.1:80) in the IPv4 Internet

  – On an IPv6 network
  – Src#p= [2001:DA9:FF3A:C8C0:A00:0:100:0]:10327 (random port)
  – Src#p= [2001:252:ffca:2669:100::]:80 (server port)

  – On translator
  – Src#p= [2001:DA9:FF3A:C8C0:A00:0:100:0]:8192 (mapped port)
  – Src#p= [2001:252:ffca:2669:100::]:80 (server port)

  – On the IPv4 Internet
  – Src#p= 58.200.192.10:8192 (mapped port)
  – Dst#p= 202.38.105.1:80 (server port)
Remarks

• Stateless/partial-state 1:N Xlate is a natural extension of stateless 1:1 Xlate

• It is an alternative approach of stateful Xlate

• When partial-state Xlate is used, the modification of IPv6 end system is NOT required. However, if this kind of modification is allowed, the translation may reduce to stateless.

• Stateless 1:N translation is one of key technologies for Double IVI [draft-xli-behave-divi]