IP-Layer Soft Handoff Implementation in ILNP

Ditchaphong (Dean) Phoomikiattisak, Saleem Bhatti
School of Computer Science, University of St Andrews
{dp32|saleem}@st-andrews.ac.uk

www.st-andrews.ac.uk
Outline

■ Goals
■ Problems
■ Overview of ILNP
■ ILNPv6 Implementation in Linux
■ Performance Evaluation
■ Conclusions & Future Works
Goals

■ **Enable IP Mobility using totally end-to-end model:**
  – Eliminate additional network entities
  – Minimise complexity and overhead

■ **Minimise packet loss during handoff:**
  – “IP-Layer Soft Handoff”
# Problems of Mobility using IP Addresses

<table>
<thead>
<tr>
<th>Protocol Layer</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>FQDN, IP address</td>
</tr>
<tr>
<td>Transport</td>
<td>IP address (+ port number)</td>
</tr>
<tr>
<td>Network</td>
<td>IP address</td>
</tr>
<tr>
<td>(Interface)</td>
<td>IP address</td>
</tr>
</tbody>
</table>

MH: Mobile Host

Diagrams:
- Site Network 1
- Site Network 2
Problems of IP Addresses

“IP addresses considered harmful”, (editorial) Brian Carpenter, ACM SIGCOMM CCR, Apr 2014
Many proposed solutions

- MIPv6 (RFC 6275) and extensions:
  - FMIPv6 (RFC 5568)
  - HMIPv6 (RFC 5380)
- PMIPv6 (RFC 6543)
- HIP (RFC 5201, RFC 5206)
- LISP (RFC 6830)
- SHIM6 (RFC 5533)
- Multipath TCP (RFC 6824)
- ILNP (RFC 6740)
Overview of ILNP

- “Identifier Locator Network Protocol”
- RFCs 6740-6748 (IRTF RRG, Experimental)
- Replace IP address with Node Identifier (NID) and Locator (L64)
- End-to-end architecture: no new entity required
## Overview of ILNP

<table>
<thead>
<tr>
<th>Protocol Layer</th>
<th>IP</th>
<th>ILNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>FQDN or IP address</td>
<td>FQDN (RFC1958)</td>
</tr>
<tr>
<td>Transport</td>
<td>IP address (+ port number)</td>
<td>Node Identifier (+ port number)</td>
</tr>
<tr>
<td>Network</td>
<td>IP address</td>
<td>Locator</td>
</tr>
<tr>
<td>Interface</td>
<td>IP address</td>
<td>(dynamic mapping)</td>
</tr>
</tbody>
</table>
Overview of ILNP

■ **NID:**
  - Is not topologically significant
  - Name a node, not an interface
  - Upper layer protocol bind only to NID

■ **L64:**
  - Is topologically significant
  - Names a (sub)network (as today's network prefix)
  - Used only for routing and forwarding in the core
ILNPv6 Implementation in Linux

- **ILNPv6:**
  - ILNP implemented as a superset of IPv6

- **Dual-stack approach:**
  - Re-use IPv6 codes

- **Kernel version 3.8.0**
ILNPv6 Implementation in Linux – Encoding NID and L64

IPv6 address (as in RFC3587 + RFC4291):

```
<table>
<thead>
<tr>
<th>64 bits</th>
<th>64 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicast Routing Prefix</td>
<td>Interface Identifier</td>
</tr>
</tbody>
</table>
```

same syntax, different semantics

IPv6 routing (address) prefix

ILNPv6 I-L vector (as in RFC6741):

```
<table>
<thead>
<tr>
<th>64 bits</th>
<th>64 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locator</td>
<td>Node Identifier (NID)</td>
</tr>
</tbody>
</table>
```

same syntax and semantics as IPv6 routing (address) prefix
so IPv6 core routers work as today

these bits only examined and acted upon by end systems
ILNPv6 Implementation in Linux – Name Resolution

- Add new syntax in /etc/hosts:

  L64 | preference, NID     hostname

  e.g.

  2001:1111:0000:0000|10,225:90ff:fe10:4959 ilnp1

- `getaddrinfo()` is modified to interpret this new syntax

**For experimental purposes only**

*DNS also possible (RFC6742, BIND 9.9.3-P1)*
ILNPv6 Implementation in Linux – ILNP Communication Cache (ILCC)

- Store information of current active ILNP sessions
  - Local/Remote NID
  - Local/Remote L64
  - Local/Remote Nonce (bidirectional, for now)

- Add when a communication session start

- Update when a node handoff

- Expire after idle time
ILNPv6 Implementation in Linux – Session Initiation

/etc/hosts (DNS)

Who is Y?

ILCC (X)

<table>
<thead>
<tr>
<th>NID_local</th>
<th>NID_X</th>
</tr>
</thead>
<tbody>
<tr>
<td>L64_local</td>
<td>L_{1X} (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID_Y</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L_Y (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N_{XY}</td>
</tr>
</tbody>
</table>

First Packet (with N_{XY})
ILNPv6 Implementation in Linux – Handoff: Hard Handoff

- Change of network:
  - simple
  - change L64 value from L1 to L2
  - gratuitous packet loss (similar to Mobile IP)
ILNPv6 Implementation in Linux – Handoff: Soft Handoff

- Unique to ILNP:
  - use L1 and L2 simultaneously (analogous to radio soft handoff)
  - minimises gratuitous packet loss
**ILNPv6 Implementation in Linux – Handoff: Locator Update (LU)**

**ILCC (X)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NID_local</td>
<td>NID&lt;sub&gt;X&lt;/sub&gt;</td>
</tr>
<tr>
<td>L64_local</td>
<td>L&lt;sub&gt;1X&lt;/sub&gt; (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID&lt;sub&gt;Y&lt;/sub&gt;</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L&lt;sub&gt;Y&lt;/sub&gt; (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N&lt;sub&gt;XY&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

**ILCC (Y)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NID_local</td>
<td>NID&lt;sub&gt;Y&lt;/sub&gt;</td>
</tr>
<tr>
<td>L64_local</td>
<td>L&lt;sub&gt;Y&lt;/sub&gt; (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID&lt;sub&gt;X&lt;/sub&gt;</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L&lt;sub&gt;1X&lt;/sub&gt; (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N&lt;sub&gt;XY&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

New prefix received
ILNPv6 Implementation in Linux – Handoff: Locator Update (LU)

ILCC (X)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NID_local</td>
<td>NID_\text{X}</td>
</tr>
<tr>
<td>L64_local</td>
<td>L_{1\text{X}} (\text{Expired* or Valid*})</td>
</tr>
<tr>
<td></td>
<td>L_{2\text{X}} (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID_\text{Y}</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L_{\text{Y}} (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N_{\text{XY}}</td>
</tr>
</tbody>
</table>

* Hard Handoff
* Soft Handoff

LU (L_{2\text{X}}) with N_{\text{XY}}

ILCC (Y)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NID_local</td>
<td>NID_\text{Y}</td>
</tr>
<tr>
<td>L64_local</td>
<td>L_{\text{Y}} (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID_\text{X}</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L_{1\text{X}} (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N_{\text{XY}}</td>
</tr>
</tbody>
</table>

new prefix received
ILNPv6 Implementation in Linux – Handoff: Locator Update (LU)

**ILCC (X)**

<table>
<thead>
<tr>
<th>NID_local</th>
<th>NID_x</th>
</tr>
</thead>
<tbody>
<tr>
<td>L64_local</td>
<td>L_{1x} (Expired* or Valid*) L_{2x} (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID_y</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L_y (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N_{xy}</td>
</tr>
</tbody>
</table>

**ILCC (Y)**

<table>
<thead>
<tr>
<th>NID_local</th>
<th>NID_y</th>
</tr>
</thead>
<tbody>
<tr>
<td>L64_local</td>
<td>L_y (Active)</td>
</tr>
<tr>
<td>NID_remote</td>
<td>NID_x</td>
</tr>
<tr>
<td>L64_remote</td>
<td>L_{1x} (Expired) L_{2x} (Active)</td>
</tr>
<tr>
<td>Nonce</td>
<td>N_{xy}</td>
</tr>
</tbody>
</table>

* Hard Handoff
* Soft Handoff

new prefix received

LU (L_{2x}) with N_{xy}

LU-ACK (L_{2x}) with N_{xy}
Performance Evaluation

- **Aim:**
  - Performance of Layer 3 handoff provided by ILNP
  - Performance of overall application flow

- **Ignore for now:**
  - Layer 2 handoff
  - QoE at the application
Performance Evaluation

- Real systems, wired connection
- Streamed UDP (emulated VoIP and ViIP) from H1 to H2
- H2 handoff every 9 sec, with 5 sec in overlap area
Results: Packet Loss

The mean gratuitous packet loss

Hard Handoff
Soft Handoff

Hard handoff: gratuitous loss observed
Soft handoff: nearly zero gratuitous loss
Results: Delay

The mean packet delay of WiFi/3G-A network

![Graph showing packet delay comparison between different scenarios and network types.]

The mean hand-off delay

![Graph showing hand-off delay comparison between different scenarios and network types.]

Hard handoff and soft handoff provide similar behaviour
Conclusion

- Implementation of ILNPv6 is not a huge work
  - Dual-stack implementation is possible:
    - Does extend current IPv6 code
- ILNPv6 could enable seamless mobility
  - Soft Handoff:
    - Minimise Packet loss
    - Enable smooth vertical handoff
  - Low handoff delay:
    - 1 RTT
Future Works

- Evaluation with wireless networks
- Direct performance comparison with Mobile IPv6
- More scenarios e.g. simultaneous movement

http://ilnp.cs.st-andrews.ac.uk/