LISP Update
(Spec updates, new protocols, interworking, and prototype/deployment status)

IRTF Vancouver - December 2007

Dave Meyer, Vince Fuller, Darrel Lewis, Eliot Lear, Scott Brim, Dave Oran, Noel Chiappa, John Curran & Dino Farinacci
Agenda

• Main LISP spec   Dino   :15
• LISP-CONS       Dave   :05
• LISP-NERD       Eliot  :05
• LISP-ALT        Vince  :15
• LISP-EMACS      Scott  :05
• Prototype       Dino   :15
• Interworking    Darrel :15
• Q & A           All    :15
LISP Main

Dino Farinacci
<dino@cisco.com>
LISP Main Agenda

- Diff description of:
  - draft-farinacci-lisp-02.txt
  - draft-farinacci-lisp-05.txt

- Why so many Mapping Database designs
  - CONS, NERD, ALT, EMACS

- What else we have been working on
  - Study on Mobility
  - Study on Interworking

- Open policy for LISP
• Added Mobility section with intro text:

Mobility without address changing. Existing mobility mechanisms will be able to work in a locator/ID separation scenario. It will be possible for a host (or a collection of hosts) to move to a different point in the network topology either retaining its home-based address or acquiring a new address based on the new network location. A new network location could be a physically different point in the network topology or the same physical point of the topology with a different provider.
Diff of -02 to -05

• Editorial changes, included definitions:
  – xTR, AFI, Negative Mapping Entry
• Statements on MTU
  – Survey indicates link MTUs are either 4470 or 9180 bytes
  – Not an issue in practice
• New data-plane UDP port number 4341
  – 4342 previously allocated for control-plane
Diff of -02 to -05

- Change data-plane packet format
  - No need for LISP type field anymore
  - Loc-reach-bits from 12 bits to 32 bits
  - Nonce from 48 bits to 32 bits
  - LISP header length did not change
  - Will latch on this packet format (no more changes)
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Version</td>
<td>IHL</td>
<td>Type of Service</td>
<td>Total Length</td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Fragment Offset</td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Source Routing Locator</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Destination Routing Locator</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Source Port</td>
<td>Dest Port (4341)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>UDP length</td>
<td>UDP Checksum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Locator Reach Bits</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>LISP Nonce</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Version</td>
<td>IHL</td>
<td>Type of Service</td>
<td>Total Length</td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Fragment Offset</td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Time to Live</td>
<td>Protocol</td>
<td>Header Checksum</td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Source EID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
<tr>
<td>Destination EID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
<td>++++++++++++++</td>
</tr>
</tbody>
</table>
Diff of -02 to -05

- Reassigned LISP Type values
  - No more 'Data Type'
- Clarification on the use of UDP checksums for data packets
  - Allowing checksum value 0 transmitted so routers don’t have to walk entire packet
  - Okay for IPv4, apparently violates RFC 2434 for IPv6
- R-bit added to Map-Reply
  - So loc-reach-bits conveyed in multi-record Map-Reply
Why so many Mapping Database Designs?

- Tough Questions that need real answers:
  - Where to put the mappings?
  - How to find the mappings?
  - Is it a push model?
  - Is it a pull model?
  - Do you use secondary storage?
  - Do you use a cache?
  - What about securing the mapping entries?
  - How to secure control messages?
  - What about protecting infrastructure from DOS-attacks?
  - What about controlling packet loss and latency?
Why so many Mapping Database Designs?

• Chicago IETF we presented
  – Pull model CONS
  – Push model NERD
  – Mentioned LISP 1.5, now is LISP-ALT
• Added LISP-EMACS
• Main LISP spec documents data-triggered Map-Replies (i.e. LISP 1.0)
Why so many Mapping Database Designs?

• All have same assumptions
  - Subscription time mapping changes
  - Locator reachability kept out of the mapping databases
  - Don’t depend on any other security design or infrastructure
Why so many Mapping Database Designs?

- How do they differ
  - Control-plane
    - CONS and NERD
  - Data-plane
    - LISP 1.0, ALT, and EMACS

- Each are trading off something different

- Desire
  - Look at alternatives and home on to 1 or 2 to go deeper (i.e. prototype)
What Else are We Working On?

- Prototyped a mobility design
  - Got it to work
  - Scale stress on the CN’s ETR
  - Solved ITR spoofing but not mapping authorization
  - Putting on shelf for now
  - Will continue to test
What Else are We Working On?

- Thinking a lot about Interworking
  - How LISP and non-LISP sites talk to each other
    - With different connection initiation models
    - With different addressing models
  - LISP site to LISP site interworking we believe is done
Open Policy for LISP

• It’s been 1 year since the IAB RAWS
  – Some of us committed to working in the IETF and IRTF in an open environment
• This is not a Cisco only effort
  – We have approached and recruited others
  – There are no patents (cisco has no IPR on this)
  – All documents are Internet Drafts
• We need designers
• We need implementers
• We need testers
• We need research analysis
• We want this to be an open effort!
LISP-CONS

Dave Meyer
<dmm@cisco.com>
CONS Agenda

• What is CONS?
• How CONS Works
• What We’ve Learned
What is CONS?

- **CONS** is a LISP Control Plane proposal
  - i.e., an Internet-scale distributed ID-to-locator mapping service
- So scalability is paramount to any solution in this space
  - Problem: controlling \((\text{state} \times \text{rate})\)
  - Preview: Consider also resolution latency
- If both factors large, we have a problem
  - \text{state} will be \(O(10^{10})\) hosts
    - Idea: aggregate EIDs into EID-prefixes to control state
  - \text{So rate must be small}
    - Idea: make mappings have \text{subscription time frequency}
    - And no reachability information in the mapping database
LISP-CONS

• LISP-CONS is a hybrid push/pull approach
• Push EID-prefixes (but not mappings) at upper levels of hierarchy
• Pull from lower levels of hierarchy
• Mappings stay at lower-levels
  - Requests get to where the mappings are
  - Replies are returned
  - This is a crucial point as we’ll see in a bit
• Getting to the lower-levels via pushing of EID-prefixes
• LISP-CONS is a mapping system for LISP 3.0
LISP-CONS

• We can get good EID-prefix aggregation
  – If hierarchy based on EID-prefix allocation and not topology
  – Then build a logical topology based on the EID-prefix allocation

• Map-Requests routed through logical hierarchy
  – Key is the EID

• Map-Reply returned to originator
  – With mapping record {EID-prefix, Locator-set}
Here's how it works

Legend:

{} : mapping entry
[] : EID aggregate
: mapping table

1. Map-Request 1.1.1.1
   - No EID-Prefix within mesh, forward to parent peer
2. Map-Request 1.1.1.1
   - No mapping cached, forward to parent peer
3. Map-Request 1.1.1.1
   - CAR has mapping, returns Map-Reply to orig CAR EID address

Legend:

ITR
qCAR
CDR
CDR Mesh
Level-0

CDR
CDR
CDR Mesh
Level-1

CDR
CDR
CDR Mesh
Level-n

ETR
qCAR
rCAR
rCAR
qCAR

1.0.0.0/8
1.1.0.0/16
1.1.0.0/24
1.1.1.0/24
1.1.2.0/24

LISP Update
IRTF Vancouver Dec 2007
Slide 22
What We’ve Learned

• We wanted to optimize aggregatability of EID prefixes
  - That led to the design in which only EID prefixes were pushed around at the higher levels (not the mappings themselves)
  - We were concerned about the rate*state product

• However, another dimension came up
  - Latency
  - So you have to trade off rate, state, and latency
  - If you push, you wind up with the whole database in network elements (state)
  - If you pull, you incur latency
  - If you try to do mobility, you get lots of updates (rate)
What We’ve Learned

• Current thinking is that a different hybrid approach might be most feasible
  – Perhaps push the whole mapping table around in the “CDR” level
• ITRs pull mappings from the “CAR” level
• This has a few nice properties:
  – You can get the whole mapping table
    • If you happen to want it
  – Latency is reduced because you don’t have to traverse the whole hierarchy to get the mappings
LISP-NERD

Eliot Lear
<elear@cisco.com>
NERD Agenda

- Goals
- What Is NERD
  - Revised from Chicago IETF
- Analysis
- What we’ve learned
Goals

- Provide a mapping mechanism for LISP that...
  - Scales to size XXXL
  - Adds no latency to connectivity
  - Uses as many transmission mechanisms as is practicable
  - Is Secure
  - Uses as much of existing code as is practicable
So What IS NERD?

- A signed compact database of EID to RLOC mappings
- A CDN is used to distribute signed databases and updates
- Successive incremental updates are used to keep databases up to date without having to retrieve entire copies.
NERD Format

```
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----------------------------------------------+
| Schema Vers=1 | DB Code      | Database Name Size |
+-----------------------------------------------+
| Database Version                                  |
+-----------------------------------------------+
| Old Database Version or 0                         |
+-----------------------------------------------+
| Database Name                                    |
+-----------------------------------------------+
| PKCS#7 Block Size      | Reserved      |
+-----------------------------------------------+
| PKCS#7 Block containing Certificate and Signature |
+-----------------------------------------------+
```
The Data

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Num. RLOCs</td>
<td>EID Mask Len</td>
<td>EID AFI</td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>End point identifier</td>
<td></td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Priority 1</td>
<td>Weight 1</td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Routing Locator 1</td>
<td></td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Priority 2</td>
<td>Weight 2</td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Routing Locator 2</td>
<td></td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Priority 3</td>
<td>Weight 3</td>
</tr>
<tr>
<td>+----------------+-----------------+--------------------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Implementation

• On the ITR:
  - Equivalent of wget with a configured URI or set of URIs
  - Discovery can be through SP (DHCP option or some P2P mechanism)
  - Pre-configured list of signing CAs

• On the Database authority
  - OOB Registration + OA&M
Growth Assumptions

Best case: Linear, 25k per year
Or 20% Compounded Growth

Things may get hairy around 2058

$10^9$
Or consider population

**World Population: 1950-2050**

- 3 Billion
- 4 Billion
- 5 Billion
- 6 Billion
- 7 Billion
- 8 Billion
- 9 Billion

Source: U.S. Census Bureau, International Data Base, July 2007 version.
How much multihoming?

• $0.1\%$ of $10,000,000,000,000$ people = $10,000,000$ routes $O(10^7)$
• $1\% = O(10^8)$
• $10\% = O(10^9)$

These two seem most interesting
## What does that get you?

<table>
<thead>
<tr>
<th># EIDs</th>
<th>2 RLOCs</th>
<th>4 RLOCs</th>
<th>8 RLOCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^5$</td>
<td>3.6 MB</td>
<td>6 MB</td>
<td>10.8 MB</td>
</tr>
<tr>
<td>$10^6$</td>
<td>36 MB</td>
<td>60 MB</td>
<td>108 MB</td>
</tr>
<tr>
<td>$10^7$</td>
<td>360 MB</td>
<td>600 MB</td>
<td>1.08 GB</td>
</tr>
<tr>
<td>$10^8$</td>
<td>3.6 GB</td>
<td>6 GB</td>
<td>10 GB</td>
</tr>
<tr>
<td>$10^9$</td>
<td>36 GB</td>
<td>60 GB</td>
<td>600 GB</td>
</tr>
</tbody>
</table>

Assume top 64 bits of all IPv6 addresses
What about memory?
# Extreme Server Load: Cold Start Scenario

<table>
<thead>
<tr>
<th># Simultaneous Requests</th>
<th>10 Servers</th>
<th>100 Servers</th>
<th>1,000 Servers</th>
<th>10,000 Servers</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>480 sec</td>
<td>48 sec</td>
<td>48 sec</td>
<td>48 sec</td>
</tr>
<tr>
<td>1,000</td>
<td>80 min</td>
<td>480 sec</td>
<td>48 sec</td>
<td>48 sec</td>
</tr>
<tr>
<td>10,000</td>
<td>13.3 hr</td>
<td>80 min</td>
<td>480 sec</td>
<td>48 sec</td>
</tr>
<tr>
<td>100,000</td>
<td>*</td>
<td>13.3 hr</td>
<td>80 min</td>
<td>480 sec</td>
</tr>
<tr>
<td>1,000,000</td>
<td>*</td>
<td>*</td>
<td>13.3 hr</td>
<td>80 min</td>
</tr>
<tr>
<td>10,000,000</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>13.3 hr</td>
</tr>
</tbody>
</table>

Assumptions: $10^8$ EIDs with 4 RLOCs per EID, 1g/s access, 100% efficient use (tcp overhead not accounted)
Updates occur every hour (longer than an hour is BAD)
## Nominal Server Load

<table>
<thead>
<tr>
<th>% Daily Change</th>
<th>100 Servers</th>
<th>1,000 Servers</th>
<th>10,000 Servers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>200 sec</td>
<td>20 sec</td>
<td>2 sec</td>
</tr>
<tr>
<td>0.5</td>
<td>1,000 sec</td>
<td>100 sec</td>
<td>10 sec</td>
</tr>
<tr>
<td>1.0</td>
<td>2,000 sec</td>
<td>200 sec</td>
<td>20 sec</td>
</tr>
<tr>
<td>5.0</td>
<td>10,000 sec</td>
<td>1,000 sec</td>
<td>100 sec</td>
</tr>
<tr>
<td>10</td>
<td>20,000 sec</td>
<td>2,000 sec</td>
<td>200 sec</td>
</tr>
</tbody>
</table>
What Could POSSIBLY Go Wrong??!

- Database changes when ETRs are renumbered or parameters change
  - What happens when a VERY large ISP decides to renumber VERY quickly?
- Existing Hardware may not like lots of writes
- Large boxes will need faster memory (not commodity)
- What happens when a database authority goes bust?
What We’ve Learned (or still need to)

• NERD requires data storage in lots of places
• We can build small NERD routers to database sizes today of up to $10^8$ EIDs using commodity hardware.
• How do we organize CDN a/o P2P networks to scale the cold start scenario higher?
More information

- draft-lear-lisp-nerd-02.txt
- Comments to rrg@psg.com ;-}
LISP-ALT

Vince Fuller
<vaf@cisco.com>
LISP-ALT Agenda

• Introduction
  - Design Goals
  - What it is and how it works
  - Infrastructure components
  - How it differs from CONS and why

• Example topology
LISP-ALT Design Goals

• Use as much technology as reasonable
  - Use what works and no more
• Minimal memory impact on ITRs
• Provide data path to reduce latency
• Allow infrastructure players to achieve new revenue source
LISP-ALT: What and How

• **Conceptually similar to CONS:**
  - Hybrid push/pull
  - Hierarchical EID prefix assignment
  - Aggregation of EID prefixes

• **But operationally very different:**
  - BGP and GRE instead of new protocol; may have deployment/ops advantages
  - Option for data-triggered Map-Replies
LISP-ALT Routers and the LAT

- LISP-ALT routers form “Logical Alternative Topology” (LAT)
  - Interconnected by GRE tunnels
  - BGP used for EID prefix propagation
  - Logical hierarchy (like CONS CAR/CDR)
- ITRs and ETRs connect at “edge”
- Who runs LISP-ALT routers?
  - ISPs, IXCs, RIRs, Neutral parties?
GRE and BGP Operation

- EID prefixes originated into BGP by ETRs or LISP-ALT routers
- ITRs learn EID prefixes via BGP from LISP-ALT routers or use “default”
  - Map-Requests are forwarded into the LAT via first-hop LISP-ALT router(s)
  - LAT routes Map-Request to ETR that “owns” EID prefix
- LISP-ALT routers aggregate prefixes “upward” in the alternative topology
Data-Triggered Mappings

- ITRs have the option of forwarding data for “un-mapped” EIDs into LAT
- Data forwarded across LAT to ETR that originates the EID prefix
- LISP Map-Reply “triggered” from ETR to ITR, installed in ITR cache
- Following traffic uses cached RLOCs
- Scaling/performance issues
The LISP Alternate Topology

Legend:
- EIDs → Green
- Locators → Red
- GRE Tunnel → Magenta
- TCP connection → Green
- Physical link → Black
- Data Packet → Yellow
- Map-Reply → Brown

EID-prefix 240.0.0.0/24

EID-prefix 240.1.0.0/16

11.0.0.1 → 240.1.1.1
240.0.0.1 → 240.1.1.1
11.0.0.1 → 240.1.1.1
240.0.0.1 → 240.1.1.1
240.0.0.1 → 240.1.1.1
EID-prefix 240.1.0.0/16

240.0.0.1 → 240.1.1.1

1.1.1.1 → 1.1.1.1
2.2.2.2
3.3.3.3

12.0.0.1

1.1.1.1 → 11.0.0.1
240.0.0.1 → 240.1.1.1
11.0.0.1 → 240.1.1.1
240.0.0.1 → 240.1.1.1
11.0.0.1 → 240.1.1.1

EID-prefix 240.1.1.0/24

EID-prefix 240.1.2.0/24

EID-prefix 240.2.1.0/24
LISP-EMACS

Scott Brim
<sbrim@cisco.com>
EMACS Agenda

• Brief Intro
• Why is it different than LISP-ALT?
• What is the fate of EMACS?
EMACS Intro

• Uses alternate topology like LISP-ALT
  – BGP over GRE
• Find ETR by multicast Data Probe
• PIM Bidirectional shared tree used
  – Over GRE topology only
• ETRs hash their EID-prefixes to a joined multicast group
• Wrong ETRs ignore
• Right ETR responds with Map-Reply over alternate or direct topology
EMACS Intro

Legend:
- EIDs -> Green
- Locators -> Red
- Tunnel
- Data Packet
- Map-Reply
- $G2 = 238.0.2.0$

EID-prefix
$2.0.100.0/24$

LISP Update

IRTF Vancouver Dec 2007

Slide 54
How is it different than LISP-ALT?

- LISP-ALT advertises EID-prefixes
  - In BGP over a GRE topology
  - Table size reduced by aggregation
  - Table size increases by deaggregation
- LISP-EMACS advertises multicast tree roots
  - In BGP over a GRE topology
  - Table size linear with number of tree roots
  - Number of routes can be less than in LISP-ALT
  - Packets will flow to more than the intended site
What is the fate of EMACS?

- EMACS is an interesting idea
- Is table reduction compelling enough for the cost of deploying PIM over GRE
- Concern about overloading sites joined to same group
- Wanted to document what could be done
- Probably won’t get prototyped
  - However, easy to implement once you do LISP-ALT
LISP Prototype

Dino Farinacci
<dino@cisco.com>
Prototype Agenda

- What’s been Added (since Chicago IETF)
- What Testing has been Accomplished
- What’s Next for Prototype and Testing
- Status on Pilot Deployment
What’s been Added

- Added 240/4 support
  - To use as EIDs
- Added 'glean-mapping' support
  - And route-returnability check for verifying when an EID has moved to a new ITR
- Brought implementation up to date with draft-farinacci-lisp-05.txt
What’s been Added

- Added LISP 1.5/LISP-ALT support
  - BGP advertises EID-prefixes over GRE tunnels
  - Data Probes sent over GRE topology
  - Map-Replies returned over GRE topology
What Testing has been Accomplished

• Detailed Test Plan written and being executed against
• Multiple EID-prefix testing completed
• Multiple locator testing completed
• Started LISP-ALT testing
LISP-ALT Topology

Legend:
- EIDs -> Green
- Locators -> Red
- eBGP-over-GRE

Dave's Lab at UofO
- AS 32768.1

Dino's Lab at cisco (non-LISP)

ITR/ETR

Darrel's Lab behind Comcast
- AS 32768.3
- PI EID-prefix 3.0.0.0/8
- 240.3.0.0/16
- ITR/ETR

ITR/ETR

ITR/ETR

Vince's Lab at cisco
- AS 32768.2
- L1-L4

Shep's Lab at Shepfarm
- AS 32768.4
- PI EID-prefix 4.0.0.0/8
- 240.4.0.0/16

Legend:
- EIDs -> Green
- Locators -> Red
- eBGP-over-GRE

PI EID-prefix 1.0.0.0/8
240.1.0.0/16

PI EID-prefix 2.0.0.0/8

L1-L2

LISP Update

IRTF Vancouver Dec 2007
What’s Next for Prototype and Testing

• Deeper dive into LISP-ALT
  – Send Map-Requests over GRE topology
  – Experiment with re-encapsulating and recursive ITRs

• More testing on map entry changing

• Think more about security mechanisms

• Think more and experiment with hybrid models
  – LISP-ALT with NERD
  – LISP-ALT with CONS
What’s Next for Prototype and Testing

- Think more and experiment with movement
- Think more about aggregation and anti-entropy models
- Implement Address-Family crossover support
  - IPv6 EIDs over IPv4 Locators
- Implement Interworking Draft
Status on Pilot Deployment

• Taking names for external pilot
  - Must be able to dedicate minimum of 1 day a week

• Shooting for Spring '08 start date

• Goals:
  - Test multiple implementations
  - Experience with operational practices
  - Learn about revenue making opportunities
LISP Interworking

Darrel Lewis
<darlewis@cisco.com>
Interworking Agenda

- Introduction
- LISP Interworking Models
- LISP Proxy Tunnel Routers (PTRs)
- LISP-NAT
Introduction

• Published Draft
  - draft-lewis-lisp-interworking-00.txt
  - Didn’t make Vancouver cut-off
• Not called ‘transition’ for a reason
  - Analogous to IPv4 to IPv6
LISP Interworking Models

• Non-LISP site to Non-LISP site
  - Today’s Internet
• LISP site to LISP site
  - LISP has this covered!
• LISP site to Non-LISP site
• Non-LISP site to LISP site
  - These last two are related
Reference Interworking Topology

Non LISP Site

LISP Site: LISP-R
(addressed from routable space)

LISP Site: LISP-NR
(addressed from non-routable space)
Routable EIDs

• EIDs published in both the existing BGP DFZ and the LISP mapping database
  – Essentially there are no sites that are ‘LISP-NR’
• EIDs can only be withdrawn from a table after transition is ‘completed’
• This mechanism will provide a low cost way for initial LISP sites to transition...
• But this isn’t really a viable option long term
Proxy Tunnel Routers (PTRs)

- Originate EID Prefixes
  - Advertise highly aggregated EID-prefix space
- Encapsulating non-LISP Traffic
Properties of PTRs

- Traffic is Asymmetrical
- Placing near the source of traffic allows for traffic to be routed on RLOCs as soon as possible
**PTR Topology**

Non LISP Site

LISP Site: LISP-R
(addressed from routable space)

LISP Site: LISP-NR
(addressed from non-routable space)

Proxy TR (PTR)
Not used in the case of LISP-NR site talking to another LISP-NR site.
LISP-NAT

- There are two main cases that involve LISP-NAT:
  - Hosts at LISP sites that use non-routable global EIDs speaking to non-LISP sites using global addresses
  - Hosts at LISP sites that use RFC 1918 private EIDs speaking to other sites, who may be either LISP or non-LISP
LISP-NAT Topology

Non LISP Site

Host

CE

LISP Site: LISP-R
(addressed from routable space)

CE (xTR)

PE

PE

PE

Host

NAT Pool
PA EID Space
128.200.1.0/24

Site Pool
LISP-NR
240.1.1.0/24

LISP Site: LISP-NR
(addressed from non-routable space)

ASNx

LISP Update

IRTF Vancouver Dec 2007

Slide 76
Q & A

<lisp-interest@lists.civil-tongue.net>
Internet Drafts

draft-farinacci-lisp-05.txt
draft-meyer-lisp-cons-03.txt
draft-lear-lisp-nerd-02.txt
draft-fuller-lisp-alt-02.txt
draft-curran-lisp-emacs-00.txt
draft-lewis-lisp-interworking-00.txt