A Click implementation of the Locator/ID Separation Protocol
The Internet is Broken!
The Internet is broken!

Growth of BGP routing table

http://www.potaroo.net/
Reasons of the BGP growth?

• More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

• More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links
Reasons of the BGP growth?

• More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- More and more networks are internally fragmented. Internet is cheaper than internal links.
Reasons of the BGP growth?

- Multihoming
Reasons of the BGP growth?

- Traffic Engineering

Client: AS4567
130.104/16

Provider AS123

Provider AS456

Internet
Reasons of the BGP growth?

- Traffic Engineering
Reasons of the BGP growth?

- Traffic Engineering
Reasons of the BGP growth?

- Traffic Engineering
Reasons of the BGP growth?

- Traffic Engineering
Reasons of the BGP growth?

- Traffic Engineering
The Internet is broken!

- The IP addresses currently used by end-hosts play two complementary roles
  - **Identifier**: the IP address identifies (with port) the endpoint of transport flows
  - **Locator**: the IP address indicates the paths used to reach the end-host
The Cure?

The Locator-Identifier Separation Protocol (LISP)

draft-ietf-lisp-05.txt
LISP Main Design Goals

• Minimize required changes to Internet
• No end-systems (hosts) changes
• Be incrementally deployable
• No router hardware changes
• Minimize router software changes
The Locator Identifier Separation Protocol (1/2)

- Define a router-based solution where current IP addresses are separated in two different spaces:
  - **EndPoint Identifiers (EID)**
    - identify end-hosts
    - non-globally routable
    - hosts in a given site are expected to use EIDs in the same prefix
  - **Routing Locators (RLOC)**
    - attached to routers (router interfaces)
    - globally routable
The Locator Identifier Separation Protocol (2/2)

• Follows the Map-and-Encap principle

• A **mapping system** maps EID prefixes onto site routers RLOCs

• Routers **encapsulate** the packets received from hosts before sending them towards the destination RLOC

• Routers **decapsulate** the packets received from the Internet before sending them towards the destination hosts
Example

Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%

ISP3
3/8

ISP1
1/8

ISP2
2/8

Site 1

Site 2
Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%
Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%
Example

Mapping System

\texttt{2001:DB8B::/56}
- \texttt{3.2.2.1 1 100\%}
- \texttt{2.2.2.1 2 100\%}

\texttt{2001:DB8A::/56}
- \texttt{1.1.1.1 1 75\%}
- \texttt{2.1.1.1 1 25\%}

Map-Request: \texttt{2001:DB8B::5678?}

Site 1
- 1.1.1.1
- 2.1.1.1

ISP3
- 3/8

ISP1
- 1/8

Site 2
- 2.2.2.1
- 3.2.2.1

ISP2
- 2/8
Example

Mapping System

2001:DB8B::/56
• 3.2.2.1 1 100%
• 2.2.2.1 2 100%

2001:DB8A::/56
• 1.1.1.1 1 75%
• 2.1.1.1 1 25%

Map-Reply:
2001:DB8B::/56 1.1.1.1
• 3.2.2.1 1 100%
• 2.2.2.1 2 100%

2001:DB8A::1234

ISP3
3/8

ISP1
1/8

ISP2
2/8

Site 1

Site 2

2001:DB8B::5678
Example

Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%
Example

Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%
Example

Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%
Example

Mapping System

2001:DB8B::/56
- 3.2.2.1 1 100%
- 2.2.2.1 2 100%

2001:DB8A::/56
- 1.1.1.1 1 75%
- 2.1.1.1 1 25%
LISP Data Packet
(IP(UDP(LISP(IP)))))
LISP Header

- include/clicknet/lisp.h

```c
struct lisphdr {
#if CLICK_BYTE_ORDER == CLICK_LITTLE_ENDIAN
  unsigned rflags:5;
  unsigned e_bit:1;
  unsigned l_bit:1;
  unsigned n_bit:1;
#else
  unsigned n_bit:1;
  unsigned l_bit:1;
  unsigned e_bit:1;
  unsigned rflags:5;
#endif
  unsigned lisp_data_nonce:24;
  uint32_t lisp_loc_status_bits;
} __attribute__((__packed__));
```
Do you speak Click?
LISP-Click

• LISP Data Plane implementation
• user space
• interoperable with (x-test @ IETF76)
  • Cisco NX-OS implementation
  • OpenLISP
• very experimental code!
LISP-Click Graph

From Device + ARP + IP

IPClassifier(...)

RLOC UDP/4242

LISPDecap(cache)

cache::LISPCache()

RLOC UDP/4241

LISPDecap(cache)

LISPEncap(cache)

To Device + ARP + IP

LISPMissManager()
IPClassifier Element

From Device + ARP + IP

**IPClassifier(...)**

- RLOC UDP/4242
- RLOC UDP/4241

**LISPMissManager()**

**cache::LISPDecap(cache)**

**LISPCache()**

**LISPEncap(cache)**

**To Device + ARP + IP**
IPClassifier Element

• Determine the action to take
  1. Control plane packet
     dst RLOC dst udp port 4342
  2. Decap
     dst RLOC dst udp port 4341
  3. Encap
LISP Cache Element

FromDevice + ARP + IP → IPClassifier(...) → RLOC UDP/4242

- cache::LISPCache()
  - LISPDecap(cache)
  - LISPEncap(cache)

LISPMissManager() → ToDevice + ARP + IP
LISPCache Element

- Stores Mapping Database and Cache
  - 2 radix trees (Quagga implementation :-( )
    - 2 priorities: primary OR backup
    - ECMP
  - Stores RLOCs information
    - 1 Hashtable
  - Control Socket (write cache.insert/.update/.rloc)
    - insert/update mappings
    - control RLOCs information
- Deal with control-plane packets (not implemented)
LISPDDecap Element

FromDevice + ARP + IP → IPClассifier(...) → RLOC UDP/4242

→ cache::LISPCache() → ToDevice + ARP + IP

→ LISPDecap(cache) → LISPEncap(cache)

→ LISPMissManager()
LISPDecap Element

- Decap in Click
  - `Packet::pull(len)`
    - Remove a header from the front of the packet
LISPDDecap Element

elements/lisp/lispdecap.cc

```c
Packet * LISPDDecap::simple_action(Packet *p_in) {
    click_ip * ip = (click_ip *)(p_in->data());
    click_udp * udp = (click_udp *)(ip + 1);
    struct lisphdr * lisp = (struct lisphdr *)(udp + 1);
    click_ip * payload = (click_ip *)(lisp + 1);
    IPAddress deid = IPAddress(payload->ip_dst);

    ... // random stuff ;-) 

    int hsize = sizeof(click_ip) + sizeof(click_udp) 
    + sizeof(struct lisphdr);
    p_in->pull(hsize);
    p_in->set_dst_ip_anno(deid);
    return p_in;
}
```
LISPEncap Element

FromDevice + ARP + IP → IPClассifier(...) → RLOC UDP/4242

To Device + ARP + IP

LISPMissManager() → cache::LISPCache() → LISPDecap(cache) → LISPEncap(cache)
LISPEncap Element

- Encap in Click
  - `Packet::push(len)`
  - *Add space for a header before the packet*
void LISPEncap::push(int, Packet *p_in)
{
    IPAddress drloc = ... // chose a destination RLOC
    ... // random stuff ;-)

    int hsize = sizeof(click_ip) + sizeof(click_udp)
                + sizeof(struct lisphdr);
    WritablePacket *p_ = p_in->push(hsize);

    ... // random stuff ;-)  

    p_->set_dst_ip_anno(drloc);
    output(0).push(p_);
}

• elements/lisp/lispencap.cc
LISPEndcap Element

- On cache miss, annotate the packet with SET_MISS_ANNO
LISPMissManager
Element

IPClassifier(…)

RLOC UDP/4242

RLOC UDP/4241

cache::LISPCache()

LISPDécap(cache)

LISPEncap(cache)

FromDevice + ARP + IP

LISPMissManager()

ToDevice + ARP + IP
LISPMissManager Element

• if packet is annotated
  • Send Map-Request for the EID (not implemented)
  • Drop the packet
• otherwise, send the packet
Further work

• IPv6 and cross AFI
• sanity checks
• smart source RLOC selection
• Full Control Plane
  • outside Click, via the Control Socket?
Want to see more?

• Meet @ 1:45pm for a live demo!
Need your help

• How to avoid ICMP port unreachable error?
  • for the moment: run a dummy server listening on UDP port 4341

FromDevice(eth0, SNIFFER 0)
Thank you

http://inl.info.ucl.ac.be
Backup
Reasons of the BGP growth?

Prefix length distribution
Reasons of the BGP growth?

- Before CIDR: Classfull network
  - ask IANA for \( n \) addresses => receive a full class
    - Class C: up to 256 addresses
    - Classe B: up to 65,536 addresses
    - Classe A: up to 16,777,216
  - First come, first served

<table>
<thead>
<tr>
<th>Network</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.0.0.0/8</td>
<td>MIT</td>
</tr>
<tr>
<td>19.0.0.0/8</td>
<td>Ford Motor Company</td>
</tr>
<tr>
<td>20.0.0.0/8</td>
<td>Computer Sciences Corporation</td>
</tr>
</tbody>
</table>
Reasons of the BGP growth?

• With CIDR: classless network
  • need $n$ addresses => receive a $/\log(n)$ prefix
  • Provider Independent (PI)
    • Owned by sites and globally announced
  • Provider Aggregatable (PA)
    • Given by ISPs from their own address block to customers. Only announce the ISP prefix