DMAP: Global Name Resolution Services Through Direct Mapping

Tam Vu
WINLAB, Rutgers University
http://www.winlab.rutgers.edu/~tamvu/

(Joint work with Akash Baid, Yanyong Zhang, Thu D. Nguyen, Junichiro Fukuyama, Richard P. Martin, Dipankar Raychaudhuri)
Today’s Internet

- IP address is used as both:
  - Routing Locator - how a device is attached to the network
  - Identifier – “who” the device is

- Results in a lot of problems:
  - Mobility
  - Site/device/network multi-Homing
  - Scalability
  - Security
  - Addressing
  - ...

Source: Cisco VNI Mobile, 2012
Cisco Forecasts 10.8 Exabytes per Month of Mobile Data Traffic by 2016
Locator – Identifier Split

- Common idea is the separation of Identifier from Routing Locator
  - Locator is for routing
  - Identifier is for naming

- The approach advocated by industry and research communities (e.g. AIP, HIP, LISP, MILSA, MobilityFirst, etc..)
Example: MobilityFirst

Named devices, content and context
Human-readable name

Strong authentication, privacy
11001101011100100...0011
Public Key Based Global Identifier (GUID)

Distributed Global Naming Resolution Service

<table>
<thead>
<tr>
<th>GUID</th>
<th>NA</th>
</tr>
</thead>
<tbody>
<tr>
<td>LapA</td>
<td>NA10,NA12</td>
</tr>
<tr>
<td>PhoneX</td>
<td>NA20 =&gt; NA21</td>
</tr>
<tr>
<td>VideoB</td>
<td>NA20,NA99</td>
</tr>
</tbody>
</table>

GUID query returns NA(s)

Global Naming Resolution Service (GNRS) (E.g., L2 => {NA1:aNode32,NA2:aNode89,.....})

L2 addr?
NA2:aNode32
NA1:aNode89

NA99

WINLAB
Naming Service Design Goals

- Mobility is directly handled using dynamic identifier to locator mapping
- Low mapping look up latency (~100ms)
- Fast mobility support requires that the mappings be updated at a time-scale smaller than the inter-query time
- Low staleness
- Support heterogeneous networked objects including devices, sensors, context, content, etc..
- Flat Identifier support
- Flat identifiers would lead to substantially more number of identifier to locator entries
- Storage Scalability (~10 billion of mappings)
- As the heart of the whole network architecture, RS must be robust
- Decentralized, cooperating resolvers
# Existing Scalable Naming Systems

<table>
<thead>
<tr>
<th></th>
<th>Lookup Latency</th>
<th>Staleness</th>
<th>Support Flat ID</th>
<th>State overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>Low</td>
<td>High</td>
<td>✗</td>
<td>Low</td>
</tr>
<tr>
<td>LISP-TREE</td>
<td>High</td>
<td>Normal</td>
<td>✗</td>
<td>Low</td>
</tr>
<tr>
<td>LISP-DHT</td>
<td>Low</td>
<td>Normal</td>
<td>✗</td>
<td>High</td>
</tr>
<tr>
<td>DHT-MAP</td>
<td>Normal</td>
<td>Low</td>
<td>✓</td>
<td>High</td>
</tr>
<tr>
<td>SILMS</td>
<td>High</td>
<td>Low</td>
<td>✓</td>
<td>High</td>
</tr>
<tr>
<td>???</td>
<td>Low</td>
<td>Low</td>
<td>✓</td>
<td>Low</td>
</tr>
</tbody>
</table>
Outline

- Motivation
- Related work
- **DMap (Direct Mapping)**
  - Minimize latency through in-network single-hop hashing
  - Leveraging reachability information of underlying routing infrastructure
- Evaluation
- Conclusion
Direct Mapping (DMap)

- GUID: $00101100 \ldots 10011001$
- Consistent hash
- IP to AS# lookup
- Global Prefix Table (e.g., BGP)
<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS#</th>
<th>Next-hop address</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/8</td>
<td>1</td>
<td>8.8.8.8</td>
</tr>
<tr>
<td>67.10/16</td>
<td>55</td>
<td>67.10.1.1</td>
</tr>
<tr>
<td>44/8</td>
<td>101</td>
<td>44.32.1.1</td>
</tr>
</tbody>
</table>
- IPX = (44.32.1.153)
- Storage AS#
- K replicas
- Mapping Update
Direct Mapping (DMap)

Consistent hash

GUID (00101100……10011001)

IP \( x \) = (44.32.1.153)

IP to AS# lookup

Storage AS#

K replicas

Global Prefix Table (e.g., BGP)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS#</th>
<th>Next-hop address</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/8</td>
<td>1</td>
<td>8.8.8.8</td>
</tr>
<tr>
<td>67.10/16</td>
<td>55</td>
<td>67.10.1.1</td>
</tr>
<tr>
<td>44/8</td>
<td>101</td>
<td>44.32.1.1</td>
</tr>
</tbody>
</table>

Mapping Update

WINLAB
Direct Mapping (DMap)

- **GUID**: (00101100……10011001)
- **Consistent hash**
- **IP to AS# lookup**
- **Storage AS#**: K replicas

**Global Prefix Table**

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS#</th>
<th>Nexhop</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/8</td>
<td>1</td>
<td>8.8.8.8</td>
</tr>
<tr>
<td>67.10/16</td>
<td>55</td>
<td>67.10.1.1</td>
</tr>
<tr>
<td>44/8</td>
<td>101</td>
<td>44.32.1.1</td>
</tr>
</tbody>
</table>

**Mapping Update**

*User A (GUID = 10)*

*Update GUID=10*

*User B* wants to contact *A*
Direct Mapping (DMap)

GUID (00101100......10011001)

Consistent hash

IP_x = (44.32.1.153)

IP to AS# lookup

Retrieve Mapping from the closest AS

Global Prefix Table (e.g. BGP)

Prefix | AS# | Nexhop
--- | --- | ---
8/8 | 1 | 8.8.8.8
67.10/16 | 55 | 67.10.1.1
44/8 | 101 | 44.32.1.1

Mapping Lookup

WINLAB
Direct Mapping (DMap)

---

**GUID**
(00101100……10011001)

**Consistent hash**

**IP**

K

**IP to AS# lookup**

---

Retrieve Mapping from the closest AS

---

**Global Prefix Table**
(e.g. BGP)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS#</th>
<th>Nexhop</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/8</td>
<td>1</td>
<td>8.8.8.8</td>
</tr>
<tr>
<td>67.10/16</td>
<td>55</td>
<td>67.10.1.1</td>
</tr>
<tr>
<td>44/8</td>
<td>101</td>
<td>44.32.1.1</td>
</tr>
</tbody>
</table>

---

**Mapping Lookup**

---

User A (GUID = 10)

---

Update GUID=10

---

User B
Wants to contact A

---

40.3.22.4

---

LEGEND:
- Resolver/Router
- Update Flow
- Query Flow
Direct Mapping (DMap)

GUID (00101100……10011001)

Consistent hash

IPx = (44.32.1.153)

IP to AS# lookup

Retrieve the mapping from the closest AS

Global Prefix Table (e.g., BGP)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS#</th>
<th>Nexhop</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/8</td>
<td>1</td>
<td>8.8.8.8</td>
</tr>
<tr>
<td>67.10/16</td>
<td>55</td>
<td>67.10.1.1</td>
</tr>
<tr>
<td>44/8</td>
<td>101</td>
<td>44.32.1.1</td>
</tr>
</tbody>
</table>

Mapping Lookup

User A (GUID = 10)

Update GUID=10

User B

Wants to contact A

Resolver/Router

Update Flow

Query Flow

LEGEND:

Mapping Lookup WINLAB
Direct Mapping (DMap)

- Minimize latency through in-network single-hop hashing
- Leveraging reachability information of underlying routing infrastructure

<table>
<thead>
<tr>
<th></th>
<th>Lookup Latency</th>
<th>Staleness</th>
<th>Support Flat ID</th>
<th>State overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMAP</td>
<td>Low</td>
<td>Low</td>
<td>✓</td>
<td>~Zero</td>
</tr>
</tbody>
</table>
Challenges

- What if the hashed IP\textsubscript{x} doesn’t belong to any ASs?
  - IP hole problem

- Mappings could be stored in random ASs?
  - Limited locality

- Infrastructure dynamism (Routers and ASs)
  - Mapping entries inconsistency
Fixing IP Holes for IPv4

- Fixing IP Holes:
  - If hash of GUID falls in the IP hole, rehash that IP m times to get out of the hole
  - Lookup follows the same process to find GUID

Map of IP (/8) address space (white = unassigned addresses)

Value at m=10 is 0.0009
Fixing IP Holes for Larger Network Addressing Schemes

- In a general network addressing scheme, we can have more holes than used segments (e.g., IPv6)
- Used address segments are hashed into N buckets
  - a two-level index: (bucket ID: segment ID)
- Mapping GUID to NA
  - $H_1$(GUID) $\rightarrow$ bucket ID
  - $H_2$(GUID) $\rightarrow$ segment ID within a bucket
Capturing Locality

- Spatial locality:
  - GUIDs will be more often accessed by local nodes (within the same AS)

- Solution: Keep a local replica of the mapping
  - A lookup can involve simultaneous local lookup and global lookup
  - Updates are issued to both Local NRS (LNRS) and Global NRS (GNRS)
Inconsistent Mapping Entries

1. GUID Publishing

<table>
<thead>
<tr>
<th>GUID</th>
<th>NA</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>H</td>
</tr>
</tbody>
</table>

2. GNRS lookup

3. GNRS Reply: H
Inconsistent Mapping Entries

1. GUID Publishing
2. GNRS lookup
3. GNRS Reply: H
4. connect
5. miss
6. Keep checking GNRS until H’
Inconsistent Mapping Entries

1. GUID Publishing
2. GNRS lookup
3. GNRS Reply: H
4. connect
5. miss
6. Keep checking GNRS until H'
7. connect reply

GUID | NA
--- | ---
C   | H'

GNRS
Prototype and evaluation

- Internet-scale simulation
  - A custom built simulation at today’s Internet scale
    - With 26,000 Autonomous Systems
    - Real-world traffic and latency from DIMES repository
  - Lookup and update latency?
  - Storage Fairness?

- Emulation of GNRS on the Orbit Testbed
  - In memory Berkeley DB on each node
  - Topology according to the Jellyfish model
  - Each Orbit node representing multiple Ass

- Qualitative reasoning using Jellyfish model
  - Effects of number of replica on look up latency?
Simulation Results – Query Latencies

The graph shows the fraction of queries versus the round-trip query response time (ms) on a log scale. It compares different values of $K$: 1, 3, and 5. The graph indicates that for $K = 5$, the 95th percentile is reached at 86 ms, while for $K = 1$, it is reached at 173 ms.
Simulation Results – Load Distribution
Tomorrow’s Internet

- A Jellyfish model
  - Captures each AS’s distance to the core

- Tomorrow’s Internet
  - More and larger ASs
  - More direct paths between ASs and the core

![Graph showing roundtrip query response time vs. number of replicas (K)]
Conclusion

- We presented the concept, design and evaluation of a highly scalable, distributed cooperative mapping system, called **Dmap**.

- We shown that leveraging reachability information of underlying routing layer would help eliminating the need of maintaining states.

<table>
<thead>
<tr>
<th></th>
<th>Lookup Latency</th>
<th>Staleness</th>
<th>Support Flat ID</th>
<th>State overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>Low</td>
<td>High</td>
<td>×</td>
<td>Low</td>
</tr>
<tr>
<td>LISP-TREE</td>
<td>High</td>
<td>Normal</td>
<td>×</td>
<td>Low</td>
</tr>
<tr>
<td>LISP-DHT</td>
<td>Low</td>
<td>Normal</td>
<td>×</td>
<td>High</td>
</tr>
<tr>
<td>DHT-MAP</td>
<td>Normal</td>
<td>Low</td>
<td>✓</td>
<td>High</td>
</tr>
<tr>
<td>SILMS</td>
<td>High</td>
<td>Low</td>
<td>✓</td>
<td>High</td>
</tr>
<tr>
<td>DMap</td>
<td>Low</td>
<td>Low</td>
<td>✓</td>
<td>~Zero</td>
</tr>
</tbody>
</table>
THANK YOU !!!

http://www.winlab.rutgers.edu/~tamvu/

Image courtesy of Jonathan Zittrain
Backup slides
LISP-TREE(1)

Fig. 2. Global overview of LISP-TREE. 1. The requester asks the discovery part to have the locator of some authoritative ETRs for an EID $e$. 2. The discovery part provides this list. 3. The Map-Resolver sends a Map-Request to one of these ETRs. 4. The ETR sends a Map-Reply back with a mapping for $e$. 

Legend:
- LISP-TREE Query
- LISP-TREE Reply
- LISP-TREE Delegation
LISP-TREE(2)

(a) Recursive mode: queries in the discovery parts are progressively transmitted to a MS. The MS answer then back-walk the tree up to the root that sends the answer to the MR.

(b) Iterative mode: queries are moving back and forth from the MR and the LTSes, starting at a root LTS until a MR is reached. The MS then provides the answer to the MR.
LISP-DHT
DHT-MAP

Illustration for the registration process.

Illustration for the locator retrieval process.

CAN BASED approach
- Bloom filter for security and speed up
- Management and policies servers overlay
What to consider while building the system?

- Where to put the mappings?
- How to find the mappings?
- How to increase robustness?
- Decentralized, cooperating resolvers
  - Fairness?
  - Infrastructure churn?
- How to use cache?
Future research directions

- Balancing routing information with information being stored in GNRS
- Large scale implementation for verification
- Securing the mapping entries
- Secure controlling message