LISP Version-Hashing

Anaheim IETF - LISP WG
Noel Chiappa, Luigi Ianone, Damien Saucez, Srini Subramanian, Dave Meyer, Vince Fuller, Darrel Lewis, John Zwiebel, Jesper Skriver, Isidor Kouvelas, Dino Farinacci
March 2010
Why Version Hashing?

• PITRs need to know when ETRs have changed their Database Mappings

• Define a Database Mapping change:
  - A locator address has been added or removed from a locator-set
  - A locator priority or weight has changed

• Problem unique to PITRs due to unidirectional data flow
  - But can be used in ITRs and LISP Mobile Nodes
What is a Version-Hash?

• 16-bit hash of EID record in a Map-Reply
  - Modulo the R-bit
• Change Reserved field to become version-hash
What is a Version-Hash?

• All ETRs at a site run independently
  - On the same database-mapping entries
  - On the same database-mapping policy entries
• So each computes the same version-hash
Who uses Version Hashes?

- PITRs (and ITRs as well) will send the version hash in encapsulated packets
- Header format now:

```
+-----------------+-----------------+-----------------+-----------------+
| L   | N|L|E| rflags |                 Nonce                     |
| I \ +-----------------+-----------------+-----------------+-----------------+
| S / |                       Locator Status Bits                     |
| P   +-----------------+-----------------+-----------------+-----------------+
```

- Changed header format:

```
+-----------------+-----------------+-----------------+-----------------+
| L   | N|L|E|V|I|flags|   Reserved    |          Version Hash         |
| I \ +-----------------+-----------------+-----------------+-----------------+
| S / |                       Locator Status Bits                     |
| P   +-----------------+-----------------+-----------------+-----------------+
```

- When V-bit is set, low-order 16-bits of nonce field contains the version-hash
Version-Hash Checking

- During ETR packet decapsulation:
  - When version-hash in LISP header does not match what ETR has cached, ETR sends a Solicit Map-Request (SMR)
- PITR (or ITR) responds to SMR with a Map-Request to any ETR at the site
- Map-Reply is returned with updated database mapping
Overloading Fields

• When V-bit is set
  - N-bit and E-bit must be 0 (using nonce field for version-hashing)

• When N-bit is set
  - V-bit must be 0 (using nonce field to hold 24-bit nonce)

• If V-bit and N-bit are both erroneously set
  - ETR interprets field as 24-bit nonce

• Echo-Noncing and Version-Hashing can run concurrently
  - Version-hashes need not be in every packet
  - Version hashing used when not in echo-nonce-request or echo-nonce state
Use Fletcher Checksum

• Sum up each byte while summing up each running sum

```
ip lisp database-mapping 240.22.0.0/16 1.11.22.22 priority 1 weight 100
ip lisp database-mapping 240.22.0.0/16 1.22.23.22 priority 2 weight 100
```

<table>
<thead>
<tr>
<th>Hex Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000</td>
<td>05A0</td>
</tr>
<tr>
<td>0x0210</td>
<td>1000</td>
</tr>
<tr>
<td>0x0000</td>
<td>0001</td>
</tr>
<tr>
<td>0xF016</td>
<td>0000</td>
</tr>
<tr>
<td>0x0164</td>
<td>FF00</td>
</tr>
<tr>
<td>0x0000</td>
<td>0001</td>
</tr>
<tr>
<td>0x010B</td>
<td>1616</td>
</tr>
<tr>
<td>0x0264</td>
<td>FF00</td>
</tr>
<tr>
<td>0x0000</td>
<td>0001</td>
</tr>
<tr>
<td>0x0116</td>
<td>1716</td>
</tr>
</tbody>
</table>

---

```
0x1A8E
0x1CB4
```

This is good!
Finds positional changes
Semantic of a Version Value

• Should the version be a hash or a monotonically increasing value?
• Do we need to know what’s different or which is relatively more current?
• The point is if the ETRs do not have the same mapping, neither version value type helps the problem
• We do not want the complexity of ETR synchronization
  - The site needs to resolve the conflict
  - Just need to spec what an ITR should do in this situation
  - Currency is in the eye of the ETR beholder
Out of Sync ETRs

(1) If ITR gets Map-Reply from D1, it can never know of a new version, it just uses active/active.

(2) If ITR gets Map-Reply from D2, it can never know if D1 and D2 are out of sync, it just uses active/backup.

(3) When ITR RLOC-probes, it gets inconsistent results, must pick one, a mono-version will tell you which is newer, a version hash can't.

Conclusion; mono-versions can make ITRs use the newer mapping but may require configuration and non-volatile storing of a version number which the user may have to deal with. Hashes are auto-generating.
Version-Hash vs Mono-Version

• Hash can tell you when the mappings are the same
• Mono-Version can only do this when ETRs are sync’ed
• Hashing has no sync requirement
• Better for the end-user, less to worry about
Proposed Plan

- If no objections, put into `draft-ietf-lisp-07`
- Make data-header changes in one revision
  - And reflect I-bit as well