Nth-Best: A Path-Hunting Solution

Brian Dickson
briand@ca.afilias.info
Summary of this draft

• Provide a solution to path-hunting
• Secondary benefit – fix persistent oscillations
• Solution should work well for both IBGP and EBGP
• Backwards compatible via capabilities negotiation
• Locally configurable depth (value of N)
Cost Summary

• Requires additional memory usage for in-rib
• requires N passes of best-path
• Update processing cost will often be reduced
• Reduction in path-hunting further reduces update frequency, and thus total cost
• No impact to FIB size
• FIB updates when best changes – no different
Basic Idea: pick the local best N and send to peers
Basic Idea: delete and send withdrawal immediately

NB: *This* withdrawal is **ALWAYS** a transient state, and will be followed later with new set of N best. We do this to propagate the withdrawal as far and as fast as possible.
Basic Idea: promote second-best
Detail per prefix: received per peer

Peer Sends | IN-RIB
---|---
Peer A: best → best, nth-best → nth-best
Peer B: best → best, nth-best → nth-best
Peer C: best → best, nth-best → nth-best

RIB:
- best
- nth-best
Selection: Pass 1
Only compare the Best from each IN-RIB

Peer Sends

Peer A
- best
- second-best

Peer B
- best
- second-best

Peer C
- best
- second-best

IN-RIB

best
second-best

best
second-best

RIB
best
second-best
Selection: Pass 2
Temporarily promote next-best of IN-RIB selected in previous pass

Peer Sends

<table>
<thead>
<tr>
<th>Peer</th>
<th>best</th>
<th>second-best</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer A</td>
<td></td>
<td>best</td>
</tr>
<tr>
<td>Peer B</td>
<td></td>
<td>second-best</td>
</tr>
<tr>
<td>Peer C</td>
<td></td>
<td>second-best</td>
</tr>
</tbody>
</table>

IN-RIB

<table>
<thead>
<tr>
<th>best</th>
<th>second-best</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>best</td>
</tr>
</tbody>
</table>

RIB

<table>
<thead>
<tr>
<th>best</th>
<th>second-best</th>
</tr>
</thead>
</table>
Update criteria

- We always want a feasible second-best through nth-best
- Next-best choice is temporarily used
- Nth-best, if promoted, must always be some peer's current best
- Track linkages from IN-RIB to RIB for optimization, so we can...
- Avoid re-calculating
Comparison with draft-walton-*

- draft-walton-add-paths and -bgp-route-oscillation address only the oscillation issue
- doesn't affect path-hunting behaviour
- does solve oscillation issue
- this draft also solves oscillation issue
- this add-paths version is more inclusive and flexible
- suggest merger of add-paths drafts
Comparison with sigcomm2000 paper example

- Excellent paper by Labowitz, Ahuja, Bose, Jahanian, sigcomm2000 on Delayed Internet Routing Convergence
  - Examples of unconstrained redistribution and path-hunting steps observed, v.s. MRAI timers
  - With proposed modification, withdrawal propagation occurs without MRAI timer exp.
  - Minimal announcements, mostly withdrawals
  - Nearly no path hunting – withdrawals catch up
Summary

• Incremental to current BGP standard
• Extra memory needed
  - Sensible implementations likely to minimize the impact of additional paths/attributes
• No additional FIB usage
• Reduced CPU usage, bandwidth, churn
• Dramatically faster local and global convergence
• Next steps?
Thank you

• Presentation on IETF 72 site
• Draft updates forthcoming (including some presented here and sent to IDR list)
• Current status: working on implementation via quagga, nearly complete
• Author: Brian Dickson, briand@ca.afilias.info