RIFT: a Novel DC Fabric Routing Protocol

DRAFT-PRZYGIENDA-RIFT

IETF ’98
• DC Fabric Routing is a Specialized Problem
• RIFT: a Novel Routing Algorithm for DC Fabric Underlay
DC Fabric Routing: A Specialized Problem

- Clos/Fat-Tree Topology Variations
- Current State of Dynamic DC Routing
- Dynamic DC Routing Requirements Matrix
CLOS Variation Topologies

• CLOS Offers Well-Understood Blocking Probabilities
• Work Done at AT&T (Bell Systems) in 1950s for Crossbar Scaling
• Fully Connected Clos is Dense and Expensive
• Data Centers Today Tend to Be Variations of “Folded Fat-Tree”, I.E. Input Stages are Same as Output Stages and Clos is “Partial”
CURRENT STATE OF AFFAIRS

• Several of Large DC Fabrics Use E-BGP with Band-Aids as IGP (RFC7938)
  – “Looping Paths” (allow-as)
  – “Relaxed Multi-Path ECMP”
  – AS Numbering Schemes to Control “Path Hunting” via Policies
  – Add Paths to Support Multi-Homing, ECMP on EBGP
  – Efforts to Get Around 65K ASes and Limited Private AS Space
  – Proprietary Provisioning and Configuration Solutions, LLDP Extensions
  – “Violations” of FSM Like Restart Timers and Minimum-Route-Advertisement Timers
• Others Run IGP (ISIS)
• Yet Others Run BGP over IGP (Traditional Routing Architecture)
• Less Than More Successful Attempts @ Prefix Summarization, Micro- and Black-Holing
  – Works Better for Single-Tenant Fabrics Without LAN Stretch or VM Mobility
### Dynamic DC Routing Requirements Breakdown (RFC7938+)

<table>
<thead>
<tr>
<th>Problem / Attempted Solution</th>
<th>BGP modified for DC (all kind of “mods”)</th>
<th>ISIS modified for DC (RFC7356 + “mods”)</th>
<th>RIFT Native DC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link Discovery/Automatic Forming of Trees/Preventing Cabling Violations</td>
<td>⚠</td>
<td>⚠</td>
<td>✓</td>
</tr>
<tr>
<td>Minimal Amount of Routes/Information on ToRs</td>
<td>⚠</td>
<td>⚠</td>
<td>✓</td>
</tr>
<tr>
<td>High Degree of ECMP (BGP needs lots knobs, memory, own-AS-path violations) and ideally NEC and LFA</td>
<td>⚠</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Traffic Engineering by Next-Hops, Prefix Modifications</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>See All Links in Topology to Support PCE/SR</td>
<td>⚠</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Carry Opaque Configuration Data (Key-Value) Efficiently</td>
<td>×</td>
<td>⚠</td>
<td>✓</td>
</tr>
<tr>
<td>Take a Node out of Production Quickly and Without Disruption</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Automatic Disaggregation on Failures to Prevent Black-Holing and Back-Hauling</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Minimal Blast Radius on Failures (On Failure Smallest Possible Part of the Network “Shakes”)</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Fastest Possible Convergence on Failures</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Simplest Initial Implementation</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>
RIFT: Novel Dynamic Routing Algorithm for Clos Underlay

- General Concept
- Automatic Cabling Constraints
- Automatic Disaggregation on Failures
- Automatic Flooding Reduction
- More Goodies
IN ONE PICTURE: DIRECTION, LINK-STATE UP, DISTANCE VECTOR DOWN & A BOUNCE
Automatic Topology Constraints

- **Level 0 = Leaf**
- **Pod 0 = Any Pod**
- **Automatic Rejection of Adjacencies Based on Minimum Configuration**
- **A1 to B1 Forbidden Due to POD Mismatch**
- **A0 to B1 Forbidden Due to POD Mismatch (A0 Already Formed A0-A1 Even If POD Not Configured on A0)**
- **B0 to C0 Forbidden Based on Level Mismatch**
- **Could Form Other Topology Variations as Well**
**Automatic De-Aggregation**

- **South Representation of the Red Spine is Reflected by the Green Layer**
- **Lower Red Spine Switch Sees that Upper Node Has No Adjacency to the Only Available Next-Hop to P1**
- **Lower Red Node Disaggregates P1**
Automatic Flooding Reduction

- Each “B” Node Computes From Reflected South Representation of Other “B” Nodes
  - Set of South Neighbors
  - Set of North Neighbors
- Nodes Having Both Sets Matching Consider Themselves “Flood Reduction Group” and Load-Balance Flooding
- Fully Distributed, Unsynchronized Election
- In this Example Case B1 & B2
- Each Node Chooses Based on Hash Computation which Other Nodes’ Information it Forwards on First Flood Attempt
- Similar to DF Election in EVPN but Much Faster
POLICY GUIDED PREFIXES (PGP)

• **South and North Variant since the “Propagation Direction” is Fixed**
  – **Avoids the “Colliding Diffused Computation Fronts” Problems**

• **Propagate like Distance Vector but Based on Flooding**
  – **No Necessity to Build Specialized Updates “per Peer”**

• **Ingress Policies Can be Applied on PGPs**
  – **No Need for “Refreshes” on Policy Changes**

• **Uses**
  – **Traffic Engineering like SR**
Moreover

• Traffic Engineering, SR is Included via PGP
• Packet Formats are Completely Model Based
• Channel Agnostic Delivery, Could Be QUICK, TCP, UDP, UDT
• Prefixes are Mapped to Flooding Element Based on Local Hash Functions
  – One Extreme Point is a Prefix Per Flooded Element = BGP Update
• Purging (Given Complexity) is Omitted
• Key-Value Store is Supported (e.g. Service Configuration During Flooding) Including Policies and “Best Copy Tie-Breaking”
SUMMARY OF RIFT ADVANTAGES

- ADVANTAGES OF LINK-STATE AND DISTANCE VECTOR
  - FASTEST POSSIBLE CONVERGENCE
  - AUTOMATIC DETECTION OF TOPOLOGY
  - MINIMAL ROUTES ON TORs
  - EASY TO ACHIEVE HIGH DEGREE OF ECMP/N-ECMP
  - MINIMAL BLAST RADIUS ON FAILURES
  - FAST DE-COMMISSIONING OF NODES

- NO DISADVANTAGES OF LINK-STATE OR DISTANCE VECTOR
  - REDUCED FLOODING
  - AUTOMATIC NEIGHBOR DETECTION

AND SOME NEITHER CAN DO

- AUTOMATIC DISAGGREGATION ON FAILURES
- SCOPE CONTROLLED KEY-VALUE STORE
SAMPLE COMPARISON TO IGP

- 21 NODES
- 60 LINKS
- 600 PREFIXES
- ALL RUN ON A SINGLE 4 CORES LOW END I7
- COMPARISON RIFT TO EQUIVALENT IGP
  - AVG. NODE CPU USE: 3x BETTER
  - CONVERGENCE (RIB): 4x FASTER
  - FLOODING: 4x LESS TRANSMISSIONS
THANK YOU ...
BLITZ OVERVIEW OF TODAY’S ROUTING

- **Link State & SPF**
- **Distance/Path Vector**


**Link State and SPF = Distributed Computation**

- **Topology Elements**
  - Nodes
  - Links
  - Prefixes
- **Each Node Originates Packets with Its Elements**
- **Packets are “Flooded”**
- **“Newest” Version Wins**
- **Each Node “Sees” Whole Topology**
- **Each Node “Computes” Reachability To Everywhere**
- **Conversion is Very Fast**
- **Every Link Failure Shakes Whole Network**
- **Flooding Generates Excessive Load for Large Average Connectivity**
- **Periodic Refreshes**
Distance/Path Vector = Diffused Computation

- Prefixes “Gather” Metric When Passed Along Links
- Each Sink Computes “Best” Result And Passes It on (Add-Path Changed That)
- A Sink Keeps All Copies, Otherwise It Would Have to Trigger “Re-Diffusion”
- Loop Prevention is Easy on Strictly Uniformly Increasing Metric
- Ideal for “Policy” Rather Than “Reachability”
- Scales When Properly Implemented to Much Higher # of Routes Than Link-State