Introduction to Seamless BFD

Nagendra Kumar Nainar
Cisco Systems
naikumar@cisco.com
Acknowledgement

• Carlos Pignataro
• Reshad Rahman
Why BFD?

Rapid Failure Detection

Fast Network Convergence

Guaranteed SLA

Protocol Dependency

- IGP timers tweaking
- Failure Propagation
- Path recalculation

FRR

- LFA, rLFA, TI-LFA
- IP, MPLS

OAM Tool Set

- Continuity Check
- Connectivity Verification
- ...
What is BFD?

- BFD stands for **Bidirectional Forwarding Detection**.
  - “BFD can provide failure detection on any kind of path between systems, including direct physical links, virtual circuits, tunnels, MPLS Label Switched Paths (LSPs), multihop routed paths, and unidirectional links (so long as there is some return path, of course).”

- Failure Detection protocol enabled within any IP Path and Clients (like routing protocol or interface instance) can subscribe to its update.

- Fast and lightweight generic failure detection mechanism.
  - Timer negotiations.
  - **Sub-second failure detection**.
  - Fixed sized control header.
  - Diagnostic relay.
  - Media independent (Ethernet, POS, Serial, etc).
  - Runs over UDP, data protocol independent (IPv4, IPv6, LSP).
  - Application independent (IGP/Tunnel liveliness, FRR trigger, etc).
# BFD Overview – Header

<table>
<thead>
<tr>
<th>Vers</th>
<th>Diag</th>
<th>Sta</th>
<th>P</th>
<th>F</th>
<th>C</th>
<th>A</th>
<th>D</th>
<th>M</th>
<th>Detect Mult</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>My Discriminator</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Your Discriminator</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Desired Min Tx Interval</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Required Min Rx Interval</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Required Min Echo Rx Interval</td>
<td></td>
</tr>
</tbody>
</table>

- **Identifies the session**
- **Tx/Rx Interval**
- **Last state change code**
- **Current Session State**

- **Auth Type**
- **Auth Len**
- **Authentication Data..**
BFD Overview – Basic Operation

- **Session Trigger**
  - No Discovery mechanism
  - Client Application triggered (IGP/MPLS-TE)

- **Session Establishment**
  - Per session “ Discriminator” assigned
  - 3 way handshake for parameter negotiation
  - Receive/Transmit Timer Negotiation

- **Failure Notification**
  - BFD notifies the client on failure
Challenges

• **Stateful**
  - State entries on both ends

• **Scalability**
  - Inefficient Resource usage

• **Slow Handshake**
  - 3-way Discriminator handshake
  - Lacks rapidness in initial path verification

• **Technology Adherence**
  - SDN/nFV etc.
  - “Continuity Check” (Path) to “Liveliness check” (Entity)
S-BFD Overview

- RFC 7880
  - Protocol definition.
  - Defines 2 “main” components

- S-BFD Discriminator
  - Unique value pre-assigned to each monitoring entity
  - Propagated to all nodes

- S-BFD Reflector Session
  - Responds if Your Discriminator (YD) matches local S-BFD Discriminator
S-BFD Discriminator

- A Unique value pre-assigned for each entity to be monitored.
  - Entity can be a node, resource, virtual host etc.

- Pre-assigned S-BFD Discriminator value is advertised to all nodes within the domain.
  - OSPF (RFC7884), ISIS (RFC7883), L2TP (RFC7885) protocol extensions available.
Each node hosting the entity to be monitored creates a reflector session.
  - One session for each Discriminator.

Responds if “Your Discriminator” matches local value.
  - No per session state entries.
S-BFD Initiator behavior

- Initiator instructed to monitor any resource.
  - Manual instruction, Protocol instruction etc
- Queries Discriminator value from local IGP database.
  - Eliminates the need for 3-way hand shake
- Generate S-BFD Control packet as below:
  - “My Discriminator” as any random value
  - “Your Discriminator” as “02020202” - the value assigned and advertised by Responder for Entity E1.
  - “State” to a value describing the local state.
  - Set “Demand” bit in control packet.
S-BFD Responder behavior

- Receiver replies if “Your Discriminator” matches local value.
- Replies with the status of the entity (UP/Down)
- No “per-session” state entries created.
S-BFD Overview – Basic Operation

Session Trigger
For Entity E1

Discrim Query

Discrim Reply

Session Establishment

Client

Failure Notification

Reflector
Session
for E1

IGP

S-BFD

R1

S-BFD

R2
Uni-directional Path Monitoring

- NVE Clients uses NVE-Hub1 as Primary GW as long as it is up.
- NVE-Hub1 pre-assigns and advertises S-BFD Discriminator as 0x02020202
- Each NVE client triggers S-BFD Control packet with YD=0x02020202
- NVE-Hub1 reflects the respond if YD matches local value.
- Irrespective of the number of NVE Clients, the session on NVE-Hub will be just one.
In Virtual Datacenter environment, Clients monitor resource within physical host.

A unique Discriminator assigned for each such resource.

Clients uses the respective Discriminator and send the control packet to physical host.
Benefits of S-BFD – State Efficiency & Scalability

- Drastically reduces the number of state entries

Number of session state is directly proportional to neighbors

<table>
<thead>
<tr>
<th>BFD Reflector Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>YD1</td>
</tr>
<tr>
<td>YD2</td>
</tr>
<tr>
<td>YDn</td>
</tr>
</tbody>
</table>

One reflector session maintained irrespective of the number of neighbors

S-BFD Reflector Session

<table>
<thead>
<tr>
<th>YD1</th>
<th>Any</th>
<th>reflect</th>
</tr>
</thead>
</table>
Benefits of S-BFD - Rapidness

- Eliminates 3 way handshake
- No Discriminator negotiation
Benefits of S-BFD – Backward Compatible

- Backward Compatible with BFD.
- Re-uses same Diag Codes.
- Assigns Discriminator outside BFD range
Seamless BFD Evaluation
Evaluation – Test Environment

Test Environment:
Number of Sessions: ~1000
Metrics studied: Memory consumption, CPU Utilization for BFD process, Time taken for session establishment.

- Tested with variable number of sessions between test units.
- Testing done with session incremented every few hundreds.
- Metric measured includes Memory consumption, CPU utilization, time taken to establish the session.
- Above metrics are measured for each iteration with different number of sessions.

*The statistics are collected from testing environment and is not absolute value but relative values.
## Evaluation Results

### Traditional BFD

<table>
<thead>
<tr>
<th>No. of Sessions</th>
<th>Memory Consumed (in KB)</th>
<th>Time taken for session establishment (in msec)</th>
<th>CPU Util</th>
<th>Rate of control packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>2138</td>
<td>4012</td>
<td>0.18%</td>
<td>Variable*</td>
</tr>
<tr>
<td>400</td>
<td>4437</td>
<td>9120</td>
<td>0.85%</td>
<td>Variable*</td>
</tr>
<tr>
<td>600</td>
<td>6679</td>
<td>12238</td>
<td>1.02%</td>
<td>Variable*</td>
</tr>
<tr>
<td>960</td>
<td>10672</td>
<td>17331</td>
<td>2.03%</td>
<td>Variable*</td>
</tr>
</tbody>
</table>

*The statistics are collected from testing environment and is not absolute value but relative values*

### Seamless BFD

<table>
<thead>
<tr>
<th>No. of Sessions</th>
<th>Memory Consumed (in KB)</th>
<th>Time taken for session establishment (in msec)</th>
<th>CPU Util</th>
<th>Rate of control packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>122</td>
<td>200</td>
<td>0.12%</td>
<td>Variable*</td>
</tr>
<tr>
<td>400</td>
<td>245</td>
<td>2510</td>
<td>0.68%</td>
<td>Variable*</td>
</tr>
<tr>
<td>600</td>
<td>369</td>
<td>3011</td>
<td>0.94%</td>
<td>Variable*</td>
</tr>
<tr>
<td>960</td>
<td>590</td>
<td>4000</td>
<td>1.92%</td>
<td>Variable*</td>
</tr>
</tbody>
</table>

### Memory Utilization Graph

- BFD(5.3.4)
- BFD(6.1.2)
- S-BFD(5.3.4)
- S-BFD(6.1.2)

### Time taken Graph

- BFD(5.3.4)
- BFD(6.1.2)
- S-BFD(5.3.4)
- S-BFD(6.1.2)
Q&A