TOMORROW starts here.
Data Center Network Failure Detection

BRKDCT-2333

Arkadiy Shapiro
Manager, Technical Marketing
NX-OS and Nexus 7000/7700

arshapir@cisco.com
Session Goals

At the end of the session, the participants should understand:

- Where failure detection fits in achieving network fast convergence
- Unique design aspects of failure detection in a data center environment
- Which failure detection technologies are needed to achieve Data Center business needs and SLAs
- Advances in failure detection technologies for Data Center
Session Non-goals

This session does not include:

- Detailed discussion on other aspects of fast convergence beyond failure detection
- Discussion on user-driven failure detection methods (ping, traceroute etc) and using scripts / EEM to automate based on result or syslog / SNMP trap
- Detailed roadmap discussion for related Cisco products
Agenda

- Overview
- Layer 1 Failure Detection
- Layer 2 Failure Detection
- Layer 3 Failure Detection
- Summary
Agenda

- Overview
- Layer 1 Failure Detection
- Layer 2 Failure Detection
- Layer 3 Failure Detection
- Summary
Routing Convergence in Action
A quick reminder...

D: I don’t care, nothing changes for me

B: my link to C is down

C: my link to B is down

A: Ok, fine, will use path via D

B: Ooops.. Problem

Loss of Connectivity = t₄ − t₀
Routing Convergence Components

1. **Failure Detection**
2. Failure Propagation (flooding, etc.)
3. Topology/Routing Recalculation
4. Update of the routing and forwarding table (RIB & FIB)

IGP and BGP Reaction
Failure Detection Overview

- Detecting the failure is **most critical** and **most challenging** part of network convergence.

- Failure Detection can occur on different levels / layers:
  - Physical Layer (1)
  - Data link Layer (2)
  - Network Layer (3)
  - Service / Application (not covered)

- Do you really need to touch all the layers?
Failure Detection Tools
Layered Approach

- **Application / Service**: IP SLA
  - FabricPath / TRILL OAM

- **Layer 3**: Aggressive Timers for Various Protocols
  - BFD for BGP, OSPF, IS-IS, EIGRP, FHRPs, static and FabricPath / TRILL

- **MPLS**: BFD for MPLS LSPs / TE-FRR

- **Layer 2**: UDLD, LACP
  - 802.1ag CFM / Y.1731 FM

- **Layer 1**: 802.3ah Link OAM
  - Bit transmission
    - Signaling: Auto-negotiation / Remote Fault Indication
    - Other: Carrier Delay / Debounce
Interconnection Options

A. Layer 3 p2p

B. Layer 3 with a Layer 1 (DWDM) “bump” in wire

C. Layer 3 with a Layer 2 (Ethernet / Frame Relay / ATM switch) “bump” in wire

D. Layer 3 with a Layer 3 (Firewall / router) “bump” in wire
Data Center Requirements

Impact on network failure detection

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Meaning for Failure Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast (often sub-second) network convergence</td>
<td>Sub-second detection</td>
</tr>
<tr>
<td></td>
<td>Link / path isolation</td>
</tr>
<tr>
<td>Port density and multi-tenant scale</td>
<td>High number of protocol sessions</td>
</tr>
<tr>
<td></td>
<td>Protocol offload</td>
</tr>
<tr>
<td>High Availability</td>
<td>Support for SSO / ISSU</td>
</tr>
<tr>
<td>Link aggregation or wide ECMP</td>
<td>Ability to operate on all interface types</td>
</tr>
<tr>
<td>Simple to configure and maintain</td>
<td>More failure scenarios covered by one technology</td>
</tr>
</tbody>
</table>

- Will traditional enterprise networking approaches apply?
- Will routing failure detection technologies apply to switching environment?
Data Center Reference Topology – VPC

Core / Edge

Aggregation / DCI / Services

Access

L2 link
FEX link
L3 link

Dark Fiber

OTV
Data Center Reference Topology – Dynamic Fabric Automation

Spine / Services

Leafs

Compute and Storage

L3 Cloud
Focus on Specific Data Center Scenarios

- Layer 2 Classical Ethernet
  - Single p2p link
  - Bundle

- FabricPath / DFA
  - Single p2p link
  - Bundle

- Layer 3
  - Single p2p link
  - Bundle
  - Sub-interfaces
  - SVI on top of Classical Ethernet
  - SVI on top of FabricPath
Agenda

- Overview
- Layer 1 Failure Detection
- Layer 2 Failure Detection
- Layer 3 Failure Detection
- Summary

Layer 1

Bit transmission
Signaling: Auto-negotiation / Remote Fault Indication
Other: Carrier Delay / Debounce
Layer 1 Failure Detection – Ethernet

Link Fault Signaling

- Ethernet mechanisms like auto-negotiation (1 GigE) and link fault signalling (10 GigE 802.3ae/ 40 GigE 802.3ba) can signal local failures to the remote end.

- Challenge to get this signal across an optical cloud as relaying the fault information to the other end is not always possible.

"Bump" in Layer 1 link
Link Down Detection

How Fast?

- Link-down / interface-down event detection is hardware-dependent

- Older hardware (ex: Catalyst 6500 6704 linecard) used per-port polling mechanism
  - Result: up to 1 sec to detect link failures at driver level

- Nexus switches and latest Catalyst 6500 linecards – interrupt-driven notification
  - Result: very fast and **predictable** detection of link failure
Carrier Delay

- Running timer in software
- Filters link **up** and **down** events, notifies protocols
- This behaviour is not desirable for Fast Convergence

```
interface ...  
carrier-delay msec 0
```

- Standard **routing** platform feature
- NX-OS only supports on SVI and sets timer at 100 msec to suppress short flaps
- Not recommended to set carrier-delay to 0 on SVI
Debounce Timer

- Delay link **down** notification only
- Runs in firmware
- 100 msec default in NX-OS
- Most cases recommended to keep it at default
- Standard **switching** platform feature

```
switch(config)interface ...
switch(config-if)# link debounce time ?
<0-5000> Timer value (in milliseconds)
```
## Carrier Delay vs Debounce timer

### General Guidance

<table>
<thead>
<tr>
<th>Carrier Delay</th>
<th>Debounce timer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runs in software</td>
<td>Runs in firmware</td>
</tr>
<tr>
<td><strong>Applicable to:</strong></td>
<td><strong>Applicable to:</strong></td>
</tr>
<tr>
<td>• <strong>Routers</strong> except Ethernet LAN switching interfaces (i.e. Cisco 7600 with WS-X6708 card)</td>
<td>• <strong>Switches</strong> except WAN interfaces (i.e. ES+ or SIP/SPA on Catalyst 6500)</td>
</tr>
<tr>
<td>• WAN interfaces on switches (i.e. ES+ or SIP/SPA on Catalyst 6500)</td>
<td>• Ethernet LAN switching interfaces on routers (i.e. Cisco 7600 with WS-X6708 card)</td>
</tr>
<tr>
<td>• SVIs on switches</td>
<td></td>
</tr>
<tr>
<td>Filters link down and up events</td>
<td>Filters link down events only</td>
</tr>
</tbody>
</table>

Make sure to test before implementing!
Agenda

- Overview
- Layer 1 Failure Detection
- **Layer 2 Failure Detection**
- Layer 3 Failure Detection
- Summary
Unidirectional Link Detection (UDLD)

- Light-weight Layer 2 failure detection protocol
- Designed for detecting:
  - One-way connections due to physical or soft failure
  - Mis-wiring detection (loopback or triangle)
- Cisco proprietary, but listed in informational RFC 5171
- Runs on any single Ethernet link, even inside bundle
- Centralized implementation in DC switching platforms (hellos sent from supervisor, not from LC)
- **Message interval**: 7-90 sec (default: 15 seconds)
- **Detection**: $2.5 \times \text{interval} + \text{timeout value (4 sec)} \rightarrow \sim 21 \text{ sec}$
**UDLD Scenario 1**

Empty-Echo condition or age out

- **Echo Packet from A to B** has “My Switch-ID A, My Port-ID e x/y”

- When B sends the echo-reply back, it is expected to have “My Switch-ID B, My Port-ID e w/z” AND “Your Switch-ID A, Your Port-ID e x/y”.

- **Transmit path failure from A to B**

- When B sends the echo-reply back, the echo-reply packet has only “My Switch-ID B, My Port-ID e w/z. B timed out!”
UDLD Scenario 2
Miswiring Detection

- Caused by packet flowing only in one direction
- Key differentiating factor of UDLD
- With LC fiber connection, this error is less common
# UDLD Failure Reaction

## Normal vs. Aggressive mode

<table>
<thead>
<tr>
<th>Normal</th>
<th>Aggressive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set port to err-disable state in case of uni-direction condition: Empty Echo packet, Uni-direction, TX/RX loop, and Neighbor Mismatch</td>
<td>Set port to err-disable state in case of uni-direction condition: Empty Echo packet, Uni-direction, TX/RX loop, and Neighbor Mismatch</td>
</tr>
<tr>
<td>Does NOT err-disable the port in case of sudden cessation of udld packets</td>
<td>Set port to err-disable state in case of sudden cessation of UDLD packets: port is put in err-disable mode if no udld packets are received for 3 x hello-time + 5 sec (=50 secs, default)</td>
</tr>
</tbody>
</table>
Spanning Tree Bridge Assurance
Almost like a routing protocol...

- Turns STP into a bidirectional protocol
- Ensures spanning tree fails “closed” rather than “open”
- All ports with “network” port type send BPDUs regardless of state
- If network port stops receiving BPDUs, port is placed in **BA-Inconsistent** state (blocked)

```bash
%STP-2-BRIDGE ASSURANCE BLOCK: Bridge Assurance blocking port
Ethernet2/48 VLAN0700.
switch# sh spanning vl 700 | in -i bkn
Eth2/48          Desg BKN*4   128.304  Network P2p *BA_Inc
```

- NX-OS / Nexus caveats:
  - Not recommended on VPC ports
  - **ISSU on Nexus 5000 not supported with STP BA** (VPC peer-link is exception)
With Bridge Assurance

Stopped receiving BPDUS!

Root

BPDUs

Network

BPDUs

Network

BA Inconsistent

Malfunctioning switch

Stopped receiving BPDUS!

%STP-2-BRIDGE_ASSURANCE_BLOCK: Bridge Assurance blocking port Ethernet2/48 VLAN0700.

switch# show spanning vl 700 | in -i bkn

Eth2/48 Altn BKN*4 128.304 Network P2p *BA_Inc
UDLD “Original” Enterprise Deployment Scenarios
Still relevant to Data Center in 2014?

Figure 1: Spanning Tree Loop Prevention

Figure 2: Spanning Tree Fast Convergence

Figure 3: Ether-channel Convergence
UDLD Best Practices

- How much do you really need UDLD?
  - Physical uni-directional failures are communicated by Layer 1 mechanisms
  - STP Bridge Assurance to account for soft failures in either direction
  - LACP to account for failures on bundle members
  - Chance of miswiring is small
  - Are you on Layer 3 / FabricPath p2p link with bidirectional protocol already running?

- If UDLD is still needed:
  - Use normal mode
  - Use default timers
Ethernet Link OAM
IEEE 802.3ah (IEEE 802.3-2008, Clause 57)

- Provides mechanisms for “monitoring link operation”
- Runs on any single point-to-point Ethernet link
- Uses “Slow Protocol” frames called OAMPDUs
- OAMPDU interval: 100 msec – 1 sec (1-10 pps)
- Minimum Timeout: 200 msec (IOS XR), 2 sec (IOS)
- Extensible and flexible protocol
- Current support mainly on following platforms: Cisco 7600, ASR 9000, ASR 901, ASR 903, ME switches

(1) No more than 10 frames transmitted in any one-second period
Ethernet Link OAM
Key Functions for Future Data Center

- **OAM Discovery**
  - Discover OAM support, peer identity and capabilities per device
  - Timeout

- **Link Monitoring**
  - Basic error definitions for Ethernet so entities can detect degraded links and isolate them

- Remote Failure Indication
  - Mechanisms for one entity to signal another that it has detected an error

- Remote Loopback
  - Used to troubleshoot networks, allows one station to put the other station into a state whereby all inbound traffic is immediately reflected back onto the link

- Remote MIB Variable Retrieval
  - Ability to read one or more MIB variables from the remote DTE
Link OAM - Link Monitoring

- Monitor link quality every 1 sec (min)

- Conditions monitored:
  - Errored Symbol Period
  - Errored Frame
  - Errored Frame Period
  - Errored Frame Seconds
  - Receive CRC (Cisco defined – IOS only)
  - Transmit CRC (Cisco defined – IOS only)

- Configure error condition thresholds to:
  - Signal peer with “Event Notification” OAMPDU
  - Syslog / SNMP trap
  - Isolate the link
Link OAM – Link Monitoring
Example: CRC Detection and Link Isolation

- Problem
  - Ensure CRCs injected by devices don’t propagate through the network
  - Need to operate with or without neighbor discovery

- Solution
  - IEEE 802.3ah for link monitoring and error-disable

```
interface GigabitEthernet1/1
  ethernet oam
  ethernet oam link-monitor receive-crc window 1
  ethernet oam link-monitor receive-crc threshold high 10
  ethernet oam link-monitor high-threshold action error-disable-interface

......
Nov 10 09:56:08.643: EOAM LM(Gi1/1): sending an EventTLV!
Nov 10 09:56:09.643: %ETHERNET_OAM-5-LINK_MONITOR: 94 rx CRC errors detected over the last 1 seconds on interface Gi1/1.
Nov 10 09:56:09.643: EOAM LM(Gi1/1): sending an EventTLV!
Nov 10 09:56:09.647: %PM-SP-4-ERR_DISABLE: link-monitor-failure error detected on Gi1/1, putting Gi1/1 in err-disable state
```
Link OAM Failure Reaction

Path Isolation

- No standards that define this!
- Depending on implementation, available options for failure reaction / path isolation:
  - Syslog / SNMP trap
  - Signal peer using specific OAMPDU
  - Error-disable
  - Error-block / Ethernet Failure Detection (EFD)
- Error-disable – operate at Layer 1, useful when need to force manual intervention after error (like miswiring)
- Today, only IOS XR can isolate path based on peer timeout or received notification OAMPDU!
Link OAM Failure Reaction
Path Isolation with Ethernet Failure Detection (EFD)

- Allows OAM to bring interface “line protocol” down when a problem is detected
- Interface is “down” to higher protocols (STP, IGPs, BGP) – will trigger reconvergence
- OAM protocols continue to operate
- Automatic and deterministic recovery when fault is resolved
- Reduced interface up/down churn
- IOS XR only, cant be used with link monitoring

```
interface TenGigE 0/1/0/0
ethernet oam
  action link-fault efd
  action discovery-timeout efd
```
Link OAM
Ready for Data Center?

- Link OAM could be adopted in future Data Center

- Rely on UDLD / STP BA / LACP for now:
  - Link OAM miswiring detection only on IOS XR as proprietary extension
  - Link OAM path isolation based on timeout only on IOS XR

- Consider Link OAM today for Data Center edge if:
  - Existence of supporting platforms
  - Must adhere to standard protocols
  - Link Monitoring capabilities

More details: BRKNMS-2202 - Ethernet OAM: Technical Overview and Deployment Scenarios
Link Aggregation Control Protocol (LACP)
IEEE 802.1AX (formerly 802.3ad)

- Protocol used to:
  - Ensure configuration consistency across bundle members on both ends
  - Ensure wiring consistency (bundle members between 2 chassis)
  - Detect unidirectional links
  - Bundle member keepalive

- Peers exchange LACP BPDUs

- Loss of heartbeat typically triggers **port suspend**
LACP Slow and Fast Hellos

- Traditional LACP heartbeat intervals:
  - Long (default): 30 sec (90 sec failure detection)
  - Short (optional): 1 sec (3 sec failure detection)

- Heartbeats typically sent from supervisor, so no SSO / ISSU guarantee with aggressive timers

- **Best practice**: use long interval and configure on both peers
  - LACP does not agree on same values!
Agenda

- Overview
- Layer 1 Failure Detection
- Layer 2 Failure Detection
- Layer 3 Failure Detection
- Summary

Aggressive Timers for Various Protocols
BFD for BGP, OSPF, IS-IS, EIGRP, FHRPs, static and FabricPath / TRILL
In some cases, failure detection relies on checks at Layer 3

How quickly can I detect a failure (neighbor down event)?

L2 bridged network

DWDM/X without LoS propagation

Tunnels (GRE, IPsec, etc.)

Something just happened!
Is Layer 3 Failure Detection Tuning Necessary?

- **Needed when:**
  - Intermediate L2 hop over L3 link
  - Concerns over any protocol software failures
  - Concerns over unidirectional failures (link or ASIC) on point-to-point physical L3 links

- **May not be needed when:**
  - Point-to-point physical L3 links with no concerns over unidirectional failures
  - Enough software redundancy to account for protocol software failures
  - FHRPs are running in active-active mode (VPC / VPC+ / Anycast HSRP)
Layer 3 Failure Detection

Protocol Timers

- All Layer 3 protocols (FHRPs, BGP, IGPs etc) use HELLOs to:
  - Maintain adjacencies (pass protocol specific info)
  - **Check neighbour reachability and detect failure**

- Hello/Keepalive and Dead/Hold timers can be tuned down, however it is not recommended:
  - Each interface may have 2-3+ protocols establishing adjacency (e.g. HSRP, PIM, OSPF on SVI)
  - Increased supervisor CPU utilization → false-positives
  - Configuration **complexity** and waste of link bandwidth
  - Challenges supporting ISSU / SSO
  - Challenges achieving **sub-second detection**
  - **Having said this:** works reasonably well in small & controlled environments
Bidirectional Forwarding Detection (BFD)
RFC 5880 / 5881

- **Lightweight** hello protocol designed to run over multiple transport protocols:
  - IPv4, IPv6, MPLS, TRILL etc

- **Designed for sub-second Layer 3 failure detection**

- Any interested client (OSPF, BGP, HSRP etc.) registers with BFD and is notified as soon as BFD detects a neighbor loss

- All registered clients benefit from uniform failure detection

- Runs on physical, virtual and bundle interfaces

- UDP port 3784 / 3785 (for echo)
Layer 3 Failure Detection with BFD

- Bidirectional Forwarding Detection (BFD) – strongly recommended over aggressive protocol timers

**General advantages:**
- Reduced control plane load and link bandwidth usage
- Sub-second failure detection
- In-flight timer negotiation

**NX-OS and Nexus 7x00 advantages:**
- Stateful restart, SSO and ISSU support
- Protocol off-load / distributed implementation – I/O module transmits / receives BFD packets
- Sub-interface optimization for session scale
- Per-link implementations with bundles
BFD Operation Modes

- Session established using asynchronous control packets

- Asynchronous mode (no echo):
  - Control packets sent at negotiated rate
  - Independent session
  - Neighbor declared dead if no packet is received for \(<\text{interval} \times \text{multiplier}>\) period

- Additionally, if echo is negotiated:
  - Control packets sent at slow rate
  - Self-directed echo packets sent at negotiated rate (\(\min (Tx, echo Rx)\)), used for failure detection
BFD – OSPF Interaction

Example

X- Forwarding plane failure between R1 and R2
X- BFD detects failure between R1 and R2
X- OSPF adjacency reset between R1 and R2
BFD in NX-OS and Nexus switches

- Version 1
- IPv4 and IPv6 (Nexus 7x00 only), single-hop
- Packets sent with DSCP CS6 / CoS 6
- Nexus 3000/5500/6000/9000 with centralized implementation
- Nexus 7x00 with distributed implementation
  - All F2, F2e, F3 and M-series I/O modules
  - Nexus 2000 ports (L2 or L3) do not support BFD
  - F1 modules don’t support L3 ports, so they can’t host BFD sessions

Layer 3 Failure Detection

<table>
<thead>
<tr>
<th>Layer 3 Protocols</th>
<th>Default timers</th>
<th>Link state notifications</th>
<th>msec timer</th>
<th>Link state notifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSPF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BGP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HSRP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NX-OS Software Support

- Nexus 3000: 5.0(3)U2(2)
- Nexus 5500/6000: 6.0(2)N2(1)
- Nexus 7000: 5.0(2a)
- Nexus 7700: 6.2(2)
- Nexus 9000: 6.1(2)I1(1)
BFD Off-load
Addressing higher session scale

- **SUP-BFD** - BFD process running on Supervisor Engine
  - Interfaces with LC-BFD processes
  - Interfaces with BFD clients

- **LC-BFD** – BFD process running on CPU of each I/O module
  - Communicates with SUP-BFD process
  - Generates BFD hellos (echo and async)
  - Receives BFD hellos from peer (async)

- Support for stateful restart, SSO and ISSU

- Additional offload to FSA accelerators on F3*

* - Future in NX-OS 7.1

Check [Cisco Nexus 7000 Verified Scale Guide](#) for latest scalability information
BFD Sub-interface Optimization

Scaling up number of sessions per module

- Single session selected among all v4/v6 sessions to run at fastest interval
- Rest of sessions run at slow interval (default: 2 sec)
- Failure of a “fast” session signaled to all “slow” sessions
- **Use-case**: high scale multi-tenancy with more routing protocol sessions than BFD sessions supported on I/O module
Layer 3 Fast Failure Detection and Link Bundles

Challenges

- Scenarios:
  1. Layer 2 bundle between 2 SVIs
  2. Layer 3 bundle

- Each node uses a hash algorithm to distribute the load across bundle members

- Chances are high that control plane packets are only carried on a single link:
  - Can’t reliably test all links
  - Single bundle member malfunction can cause black holes which remain undetected
  - Rely on Layer 1 or Layer 2 (LACP/PaGP/UDLD/OAM) detection

- Can use parallel Layer 3 links instead, load-sharing properties are often similar

- Two approaches for BFD:
  1. Single session
  2. Per-link sessions
BFD Logical Mode
Nexus 3000 / 5500 / 6000 / 7x00

- **Single**: BFD session per L3 destination address
- Internal algorithm to determine which I/O module hosts BFD session
- BFD packet distribution:
  - Prior to NX-OS 5.2(1) – Tx packets are polarized on one bundle link per session
  - From NX-OS 5.2(1) – Tx packets are round-robin load-balanced on all bundle links
- Rx packets are always polarized on one bundle link per session
- Async + echo
- Minimum interval is 250 msec x 3
BFD Per-link Mode
Nexus 6000 / 7x00

- BFD session per port-channel member
- Master session on SUP consolidates member states and communicates with clients
- LACP is required for port-channels
- Async only, no echo
- Layer 3 port-channel / sub-interface only
- Minimum interval: 50 msec x 3
BFD Interoperability with Bundles

- **Use-case: Nexus – non-Nexus at DC edge**
- Current standards do not address this!
- **Single session**
  - Easiest to achieve with current standards and implementations
  - **Verified interoperability between IOS XR BLB mode, IOS and NX-OS single session mode**
- **Per-link sessions**
  - Recommended, but platform proprietary
  - IETF [draft-ietf-bfd-on-lags-04](http://www.ietf.org/rfc/rfc5880.txt) will address interoperability!
BFD and Nexus Virtual Port-Channel (VPC / VPC+)

- BFD over VPC (MCEC) – due to no L3 over VPC support
- BFD over VPC peer-link support:
  - Nexus 7000 M1, M2 or F3 peer-link
  - Nexus 7000 F2/F2e peer-link in F2/F2e VDC
  - Nexus 7000 F2e peer-link in M-F2e VDC*
  - Nexus 3000 (no echo)
  - Nexus 5500/6000

* - Software support in NX-OS 6.2
**BFD for FabricPath**

FabricPath IS-IS as BFD client

- **Use-case:** peer switch path failure detection
- Upcoming support for FabricPath and DFA
- IETF draft-ietf-trill-rbridge-bfd-07
  - Does not include bundle per-link, so proprietary implementation
  - BFD TLV (RFC 6213) - IS-IS notifies BFD of Rbridge IDs and waits for BFD session to come up before reaching full adjacency
- Ethernet Link OAM could be adopted in future
- FabricPath OAM in the works for service / end-to-end failure detection and monitoring
BFD for FabricPath / TRILL
Point-to-Point vs. Shared Ethernet segment

- TRILL specifies support shared Ethernet segment with several peers
- FabricPath can only peer on point-to-point links
- BFD may be more needed for TRILL than FabricPath except…
FabricPath and Failure Detection
Data Center Interconnect Example

- DCI may require BFD for FabricPath
BFD over FabricPath
FabricPath as a Transport for BFD

- Routing protocol / FHRP peering over FabricPath network
- BFD between SVIs or sub-interfaces
BFD over FabricPath

Example

- **Use-case:** HSRP on CE connected devices

- Forwarding path testing?
  - HSRP – multicast following FP multi-destination trees
  - BFD – unicast
  - Potentially different paths

- Recommendation to increase BFD timers
  - Account for any failures in FabricPath network and re-convergence
BFD for Static Routes

Single Hop Example

- **Use-case:** gateway / next-hop liveliness detection
- **Fail-close solution** (remove static route and not reinstate until BFD is up)
- Must be configured on both ends

```
ip route 0.0.0.0/0 Vlan10 20.0.0.1
ip route static bfd Vlan10 20.0.0.1

ip route 30.0.0.0/24 Vlan 20 10.0.0.1
ip route static bfd Vlan20 10.0.0.1
```

```
switch# sh ip route
0.0.0.0/0, ubest/mbest: 1/0
*via 20.0.0.1, Vlan 10, [1/0], static

switch# sh ip route
30.0.0.0/0, ubest/mbest: 1/0
*via 10.0.0.1, Vlan 20, [1/0], static
```
BFD Multihop
RFC 5883

- BFD single hop sends packets with TTL=1
- If intermediate device decrements TTL, need multi-hop
- Use-case 1: eBGP multihop
- Use-case 2: firewall / NAT load-balancing with static route
IP SLA tracking for Static Route or PBR

Alternative to BFD Multi-hop with more capabilities

- Establish IP SLA probe as an object
- PBR path or static route availability is dependent on IP SLA state or **reachability**
- PBR is done in hardware on Nexus switches
- Use-case: firewall / NAT load-sharing with **PBR** or static route
BFD and Security

- Disable platform hardware security mechanisms for BFD echo to function:
  - uRPF (per interface)
    ```
    no [ip|ipv6] verify unicast source reachable-via [any|rx]
    ```
  - IDS checks (global)
    ```
    no hardware ip verify address identical
    ```
  - IP redirects (per interface)
    ```
    no ip redirects
    no ipv6 redirects
    ```

- Open rules to allow echo packets though firewall or enable loopback as source IP:
  ```
  bfd echo-interface <a_loop_back_interface>
  ```
1. If Layer 3 fast failure detection is needed, use BFD for all protocols
2. If cant use BFD, verify aggressive timer support
3. Always plan your BFD scale and check with platform capabilities (centralized vs distributed architecture, interface and client support locally and on peer)
4. Use BFD echo (default) whenever possible, check security
5. On Layer 3 port-channels, use per-link mode and prefer that over echo
6. BFD single-hop for BGP – make sure neighbor update source is a directly connected interface
7. Make sure BFD packets are prioritized appropriately (Marked with IP precedence 6 / DSCP CS6 / CoS 6, can also be classified by udp 3784+3785)
Agenda

- Overview
- Layer 1 Failure Detection
- Layer 2 Failure Detection
- Layer 3 Failure Detection
- Summary
Summary of Recommendations

- Layer 2 Classical Ethernet
  - Single p2p link
  - Bundle

- FabricPath / DFA
  - Single p2p link
  - Bundle

- Layer 3
  - Single p2p link
  - Bundle
  - Sub-interfaces
  - SVI on top of Classical Ethernet
  - SVI on top of FabricPath
Summary

- Fast Failure Detection is **Key** to Fast Convergence
- Match Data Center requirements to the right set of technologies
  - One protocol may be enough – keep it simple!
  - Evolving field with industry and Cisco innovations
- Design your network to take advantage of best practices
### Related Cisco Live Milan 2014 Events

<table>
<thead>
<tr>
<th>Session-ID</th>
<th>Session Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TECRST-3314</td>
<td><strong>Fast Convergence</strong> and Scalability in Virtualized Network Designs</td>
</tr>
<tr>
<td>BRKNMS-2202</td>
<td><strong>Ethernet OAM</strong>: Technical Overview and Deployment Scenarios</td>
</tr>
<tr>
<td>BRKDCT-2081</td>
<td>Cisco <strong>FabricPath</strong> Technology and Design</td>
</tr>
</tbody>
</table>
Call to Action…

Visit the World of Solutions:-

- Cisco Campus
- Walk-in Labs
- Technical Solutions Clinics

- Meet the Engineer

- **Lunch Time Table Topics**, held in the main Catering Hall

- **Recommended Reading**: For reading material and further resources for this session, please visit [www.pearson-books.com/CLMilan2014](http://www.pearson-books.com/CLMilan2014)
Complete Your Online Session Evaluation

- Complete your online session evaluation
- Complete four session evaluations and the overall conference evaluation to receive your Cisco Live T-shirt
BFD Clients and Interface Types

Nexus 7x00

- Supported BFD IPv4 clients:
  - OSPFv2
  - EIGRP
  - IS-IS
  - PIM
  - BGP
  - HSRP
  - VRRPv2
  - Static routes
  - MPLS TE FRR
  - OTV IS-IS (from NX-OS 6.2(2))

- Supported BFD IPv6 clients from NX-OS 6.2:
  - OSPFv3
  - EIGRP
  - IS-IS
  - PIMv6
  - BGP
  - Static (from NX-OS 6.2.2a)

- Supported Layer 3 interface types:
  - Switched Virtual Interface (SVI)
  - Physical Port and sub-interface
  - Port Channel (per-link and logical modes) and sub-interface

Note: check configuration guides and release notes for latest supported configuration
BFD Clients and Interface Types
Nexus 3000 / 3100

- Supported BFD IPv4 clients:
  - OSPFv2
  - BGP
  - Static

- Supported Layer 3 interface types:
  - Switched Virtual Interface (SVI)
  - Physical Port and sub-interface
  - Port Channel (logical mode only) and sub-interface

Note: check configuration guides and release notes for latest supported configuration
BFD Clients and Interface Types
Nexus 5500/6000

- Supported BFD IPv4 clients:
  - OSPF v2
  - BGP
  - EIGRP
  - PIM
  - HSRP
  - VRRP
  - Static route

- Supported Layer 3 interface types:
  - Switched Virtual Interface (SVI)
  - Physical Port and sub-interface
  - Port Channel (logical mode, per-link only for Nexus 6000) and sub-interface

Note: check configuration guides and release notes for latest supported configuration
# Protocol Comparison

<table>
<thead>
<tr>
<th>OSI Layer</th>
<th>BFD</th>
<th>UDLD</th>
<th>Link OAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>L3</td>
<td>L2</td>
<td>L2</td>
</tr>
<tr>
<td>Standard</td>
<td>IETF RFC 5880 / 5881 (with some Cisco enhancements)</td>
<td>Cisco proprietary</td>
<td>IEEE 802.3-2008 (with some Cisco enhancements)</td>
</tr>
<tr>
<td>Failures Detected</td>
<td>Uni-directional soft failures</td>
<td>Uni-directional soft failures</td>
<td>Uni-directional soft failures</td>
</tr>
<tr>
<td></td>
<td>Bidirectional soft failures</td>
<td>Bidirectional soft failures</td>
<td>Bidirectional soft failures</td>
</tr>
<tr>
<td></td>
<td>Uni-directional soft failures</td>
<td>Mis-wiring Detection</td>
<td>Mis-wiring Detection (IOS XR)</td>
</tr>
<tr>
<td></td>
<td>Bidirectional soft failures</td>
<td></td>
<td>Link Degradation</td>
</tr>
<tr>
<td>Failure Reaction</td>
<td>Notify peer and clients</td>
<td>Error-disable (depending on mode)</td>
<td>Notify peer</td>
</tr>
<tr>
<td></td>
<td>Remove link from bundle (IOS XR, IETF standard in future)</td>
<td></td>
<td>Error-disable (depending on error type and platform)</td>
</tr>
<tr>
<td></td>
<td>BFD dampening (IOS XR)</td>
<td></td>
<td>Error-block</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ethernet Failure Detection (IOS XR)</td>
</tr>
<tr>
<td>Bundles and Virtual Interfaces</td>
<td>Bundle logical, bundle per-link, SVI, sub-interface</td>
<td>Single L2 links</td>
<td>Single L2 links</td>
</tr>
<tr>
<td>Message Interval and Timeout</td>
<td>Configurable, exchanged and negotiated</td>
<td>Configurable and exchanged</td>
<td>Configurable, not exchanged</td>
</tr>
<tr>
<td></td>
<td>Timeout generally in msec</td>
<td>Timeout generally in 20+ seconds</td>
<td>Timeout generally in 2+ seconds</td>
</tr>
<tr>
<td>ISSU</td>
<td>Timer inflation</td>
<td>Flush message sent (IOS XR)</td>
<td>No (can be extended in future)</td>
</tr>
</tbody>
</table>