



Big Cloud Fabric – Training & Lab Guide

PRESENTED BY BIG SWITCH NETWORKS

INFO@BIGSWITCH.COM



Training Outline

TRAINING OUTLINE

Day 1

Module 0:	Introduction and Logistics	20 mins
Module 1:	Introduction to Big Cloud Fabric (BCF)	45 mins
Module 2:	BCF Access and Configuration <ul style="list-style-type: none">▪ Access Modes (GUI, CLI, REST)▪ Configuration Management & System Configuration▪ Lab – Hands-on BCF Environment	45 mins
Module 3:	BCF Bring-up <ul style="list-style-type: none">▪ Components,▪ Management / Control Networks▪ Controller / Switch / Fabric Bring up▪ Lab – Working with Controllers & Switches	90 mins
Module 4:	BCF Logical Topology <ul style="list-style-type: none">▪ Tenants, Segments▪ Logical Routers, Policies	60 mins

TRAINING OUTLINE

Day 2

Module 4: BCF Logical Topology (Cont'd)

- **Lab – Configuring BCF Topologies** 120 mins

Module 5: BCF Additional Configurations

- QoS, Multicast
- BGP (Connect to L3 network)
- Inter-POD Connectivity (L3, L2 – VLAN/VxLAN)
- Extending Segments
- BPDU Protection (Connect to xSTP enabled network)
- Storm Control
- sFlow, SPAN

Module 6: BCF Troubleshooting

- Tools & Capabilities
- Flow Chart
- Starting Points
- **Lab – Troubleshooting BCF** 120 mins

TRAINING OUTLINE

Day 3

Module 7: BCF Operations	40 mins
▪ Upgrade BCF, High Availability ▪ RMA Procedures, Change Management	
Module 8: BCF Analytics	20 mins
▪ Analytics dashboards, Navigation, ▪ Custom & Advanced Queries ▪ Lab – Using Analytics	60 mins
Module 9: BCF vCenter Integration	60 mins
▪ Automation, Visibility, Troubleshooting	
Module 10: BCF In-Depth	60 mins
▪ Fabric Components, Fabric LAGs ▪ Discovering Endpoints, Packet Walks	
Module 11: Contacting Support	10 mins

Module 0 – Introduction and Logistics

INTRODUCTION AND LOGISTICS

Module Outline

- Company Introduction
- Personal Introductions
- Logistics & BSN Labs

BIG SWITCH NETWORKS – OVERVIEW

Big Switch Networks

The Next-Generation Data Center Networking Company

Company Mission

Next-Generation DC switching

Next-Generation DC security and monitoring

Products/
Solutions



Big
Cloud
Fabric



Big
Monitoring
Fabric

VMware SDDC
(vSphere, NSX, vSAN)

OpenStack
(NFV/Private Cloud)

Containers
(Docker, Kub, Mesos, RH)

Pervasive
Visibility

DMZ
Security

Cloud
Monitoring



SDN Controller Software

Open Networking Hardware

Scale-out Fabric Architecture

Networking
Architecture
(Hyperscale
Inspired)

INTRODUCTION AND LOGISTICS

Personal Introductions

- Name
- Current area of responsibility
- Network experiences
- Course expectations

INTRODUCTION AND LOGISTICS

Administrivia & BSN Labs

Admin

- Class hours
- Lunch/Breaks
- Wifi – password?
- Emergencies

BSN Labs / Training Labs

- Sign in using provided Email & Password
- Recommended Browsers
 - Windows 10 Users: Firefox
 - All other users: Chrome
- CLI Access Methods
 - Preferred: SecureCRT, PuTTY, Mac terminal, . . .
 - Alternative: Web based CLI provided

INTRODUCTION AND LOGISTICS

Connecting to BSN Labs

BSN Labs

- Labs URL
 - <http://labs.bigswitch.com/training>



- Login credentials
 - Email: user<xx>@bigswitch.com
 - Password: As provided by instructor

LOGIN

ACCESS BSN LABS

By registering and logging in you agree to our [terms of service](#) and [privacy policy](#).

SIGN IN

INTRODUCTION AND LOGISTICS

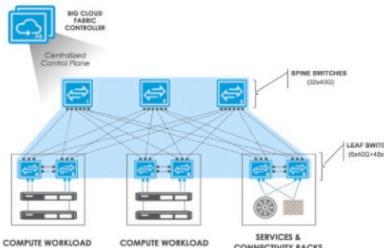
Launch Module

Launch Module

- Select tab Big Cloud Fabric (if not selected)
- Launch Module 1
- Description may not be the same as in the following screenshot

 Big Cloud Fabric

 Big Monitoring Fabric



MODULE 1
Big Cloud Fabric 3.5 P-Edition Training
Introduction to the fundamental concepts and the architecture of Big Cloud Fabric.

LAUNCH

INTRODUCTION AND LOGISTICS

Launch Progress

Launch Progress

- Labs hosted on AWS
- Takes about 10 – 15 minutes to provision lab

OK, HANG-ON WHILE WE LAUNCH:

MODULE 1

Big Cloud Fabric Training

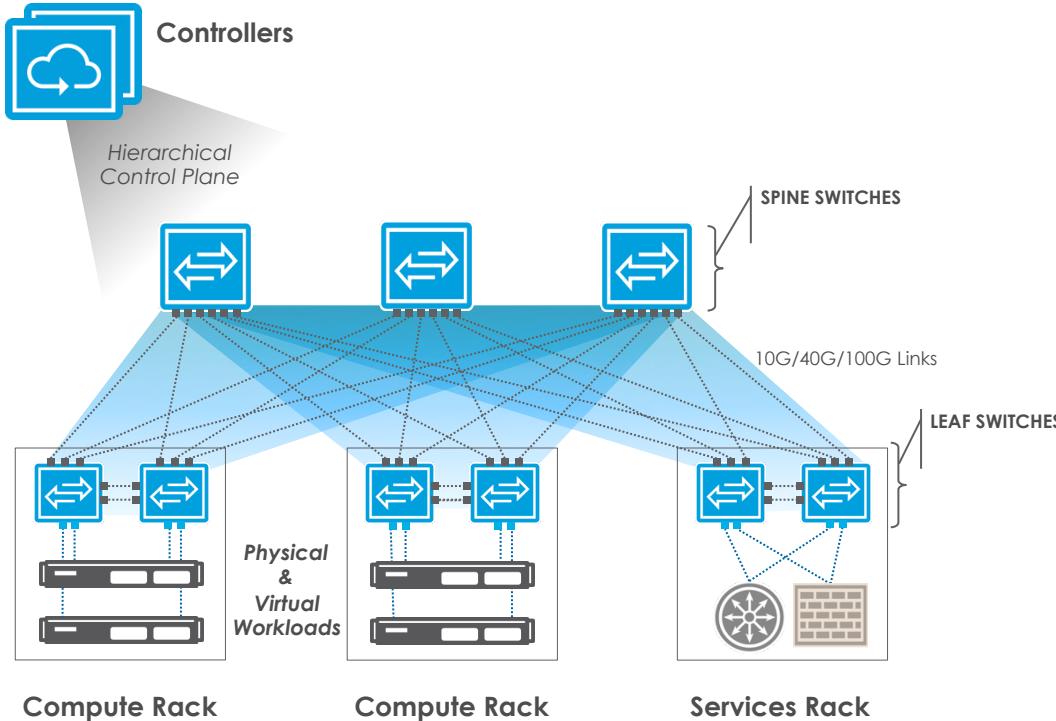


Questions?

Module 1 – Introduction to Big Cloud Fabric

INTRODUCTION TO BIG CLOUD FABRIC

SDN Data Center Fabric



Hyper Scale Inspired

- Spine / Leaf Ethernet fabric
- Open Ethernet switches
- Centralized control

Simplicity

- Protocol free
- Zero touch
- Single pane of glass

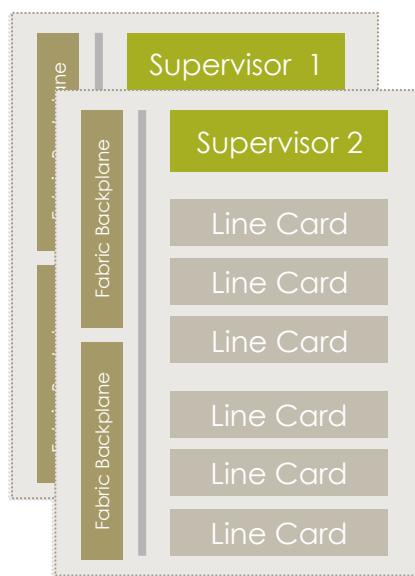
Capabilities

- Tenant centric
- Service Insertion
- Programmable
- Orchestration integration

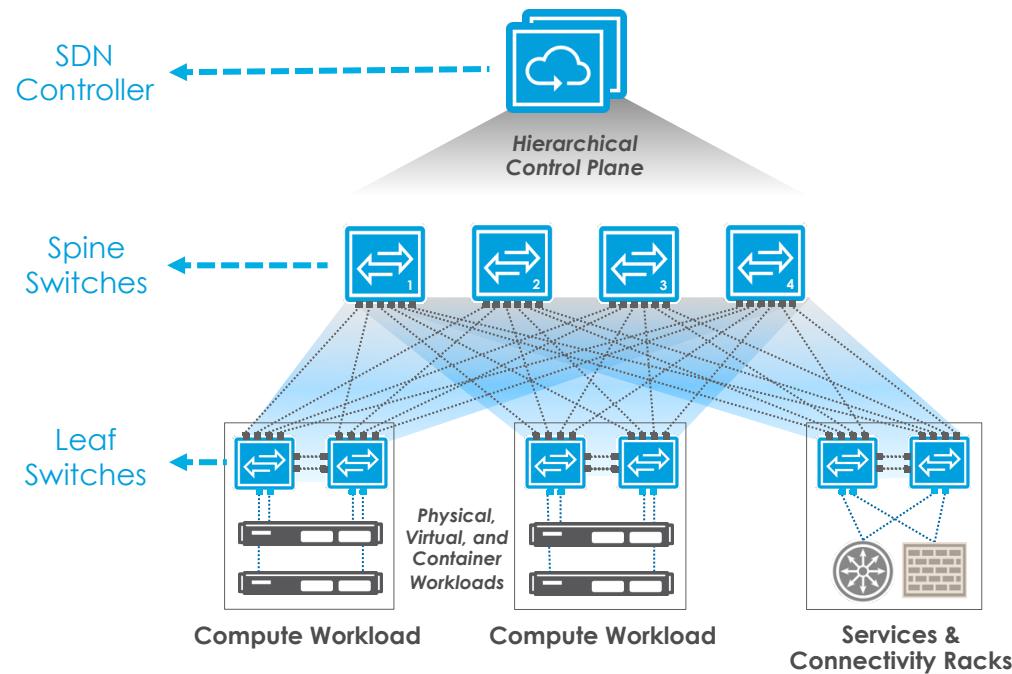
INTRODUCTION TO BIG CLOUD FABRIC

Logical Chassis Architecture – “One Big Switch”

Traditional Chassis Pair

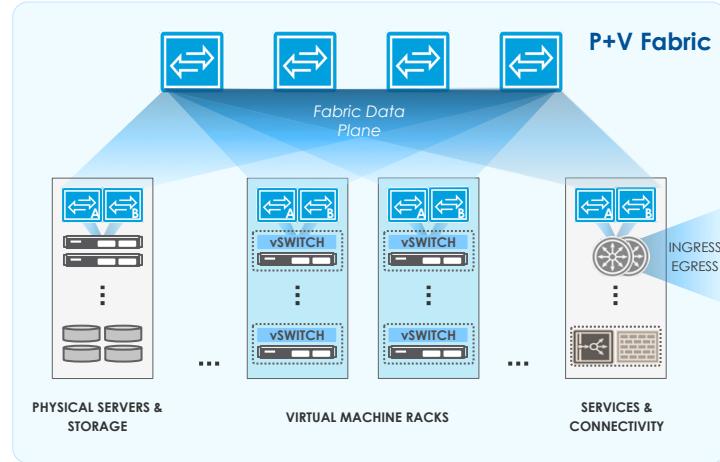
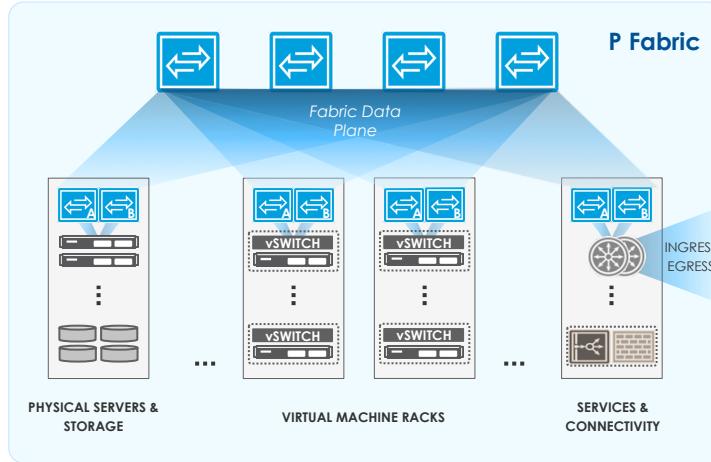


Logical Chassis Pair



INTRODUCTION TO BIG CLOUD FABRIC

Two Editions



Fabric Domain

- Controllers, Spine and Leaf switches

Use Cases

- L2/L3 Data Center switch, . . .

Fabric Domain

- Controllers, Spine, Physical Leaf, Virtual Leaf Switches

Use Cases

- OpenStack & Container Integration

INTRODUCTION TO BIG CLOUD FABRIC

Use Cases

1 Data Center Fabric



P Fabric

2



P Fabric

3



P+V Fabric

4 Containers



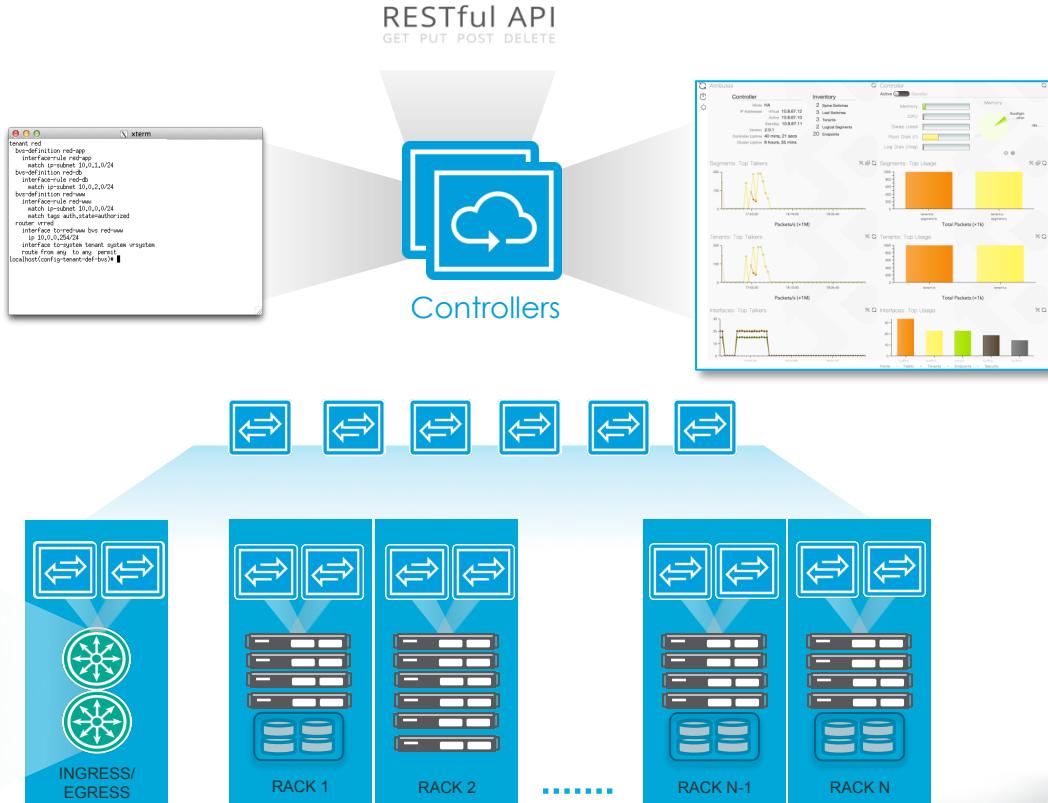
P+V Fabric

Use Cases: IaaS Clouds, Big Data/HPC, VDI, NFV, SDS, ...

INTRODUCTION TO BIG CLOUD FABRIC

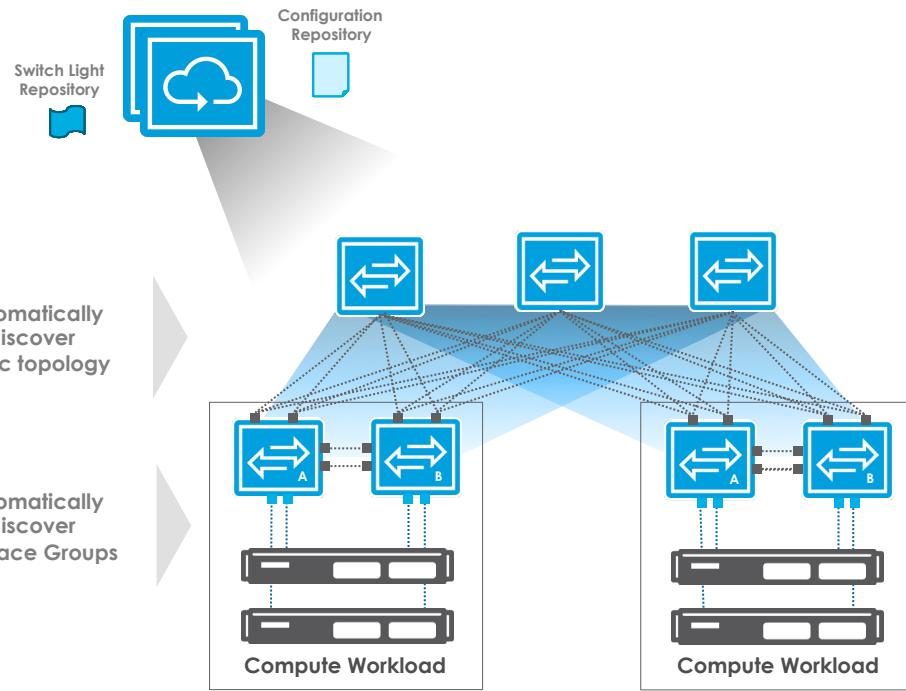
Single Pane of Glass

- CLI/GUI/REST API
 - CLI/GUI are REST clients
- Over 130 devices
- Configuration
- Monitoring
- Troubleshooting



INTRODUCTION TO BIG CLOUD FABRIC

Zero Touch Fabric



BCF Bring-up

- Install and cable switches & controllers
- Perform first-boot configuration on controller
- Configure switch MAC addresses and roles
- Power up switches

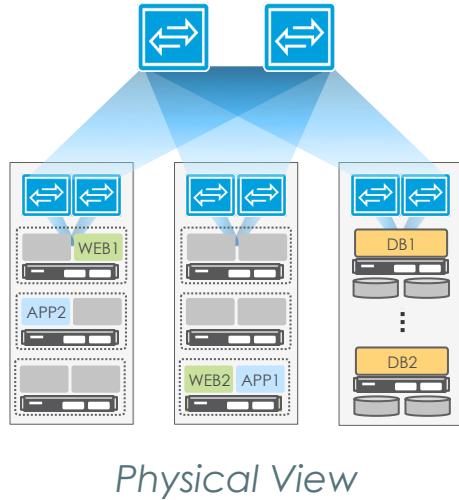
- Auto discover switches
- Auto load Switch Light OS
- Auto discover fabric links
- Auto discover hosts (vCenter & OpenStack)
- Auto discover Interface Groups (vCenter & OS)



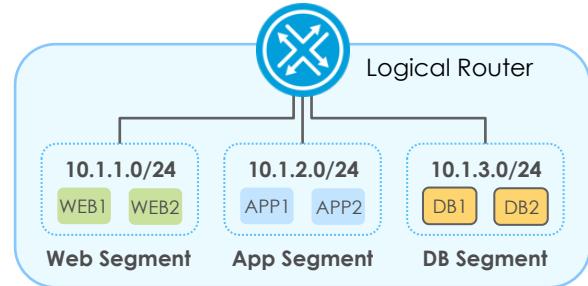
No box-by-box bring-up or configurations

INTRODUCTION TO BIG CLOUD FABRIC

Logical View



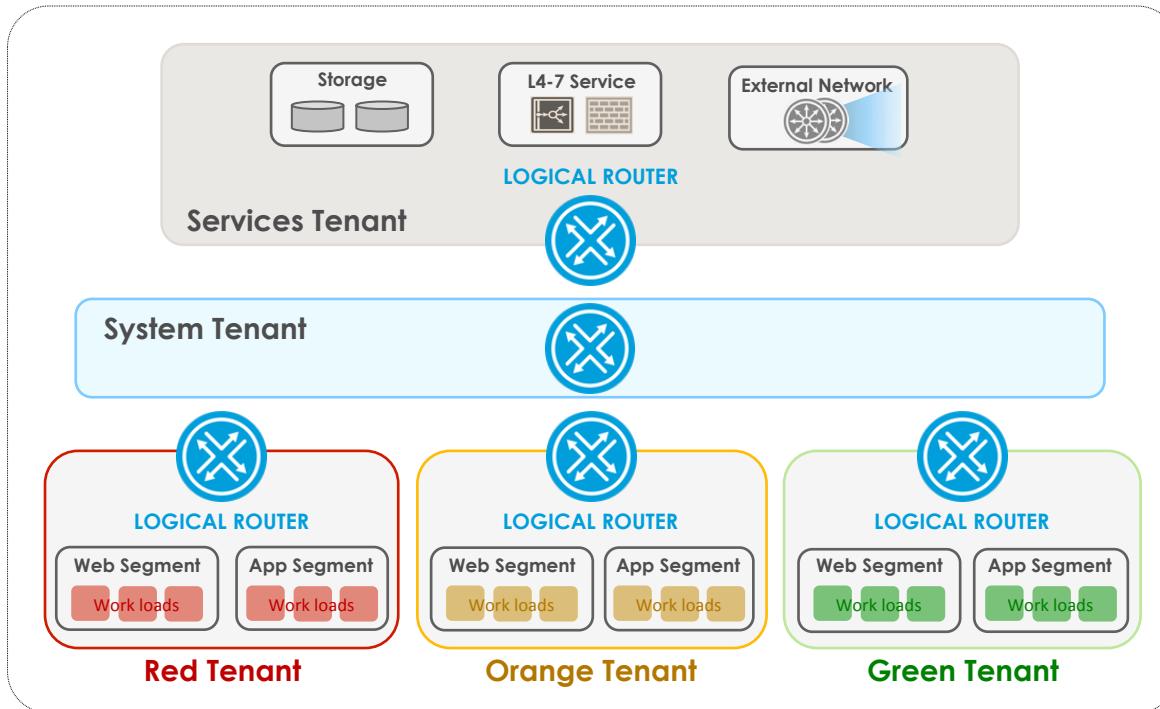
Tenants
Segments
Logical Routers



Logical View

INTRODUCTION TO BIG CLOUD FABRIC

Network Topology



Network Topology

- Multi-tenant configurations
- System tenant for inter-tenant connectivity
- BGP for external connectivity

Policies

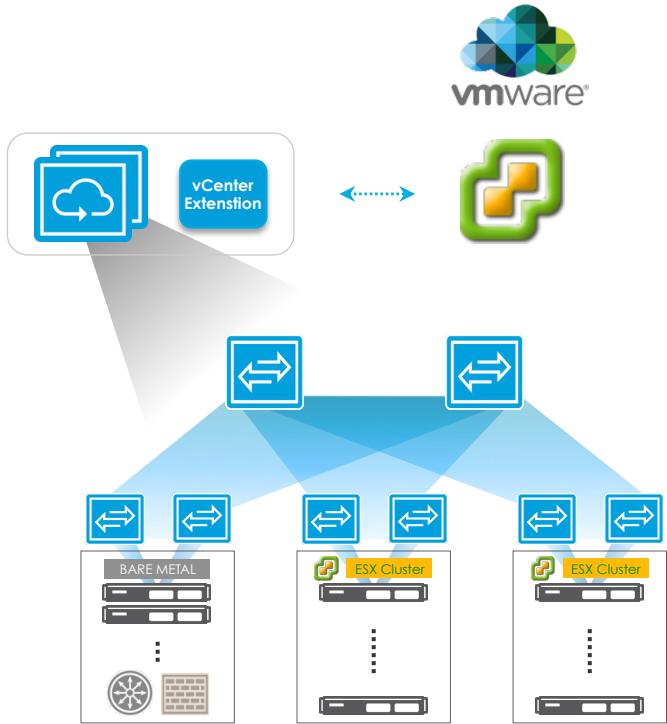
- Granular access control
- Declarative mode (what to do) vs. Imperative mode (how to do it)
- Configurable on all routers

Service Insertion

- Implement security policies
- Redirect traffic using policies

INTRODUCTION TO BIG CLOUD FABRIC

vCenter Integration



Single source for virtual/physical network configurations

Automation

- Auto learn virtual network configurations (vSwitches & vCenter port groups)
- Auto detect ESXi hosts
- Auto configure host uplinks, tenant, L2 segments & VMs
- Auto policy migrations for vMotion/DRS

Visibility

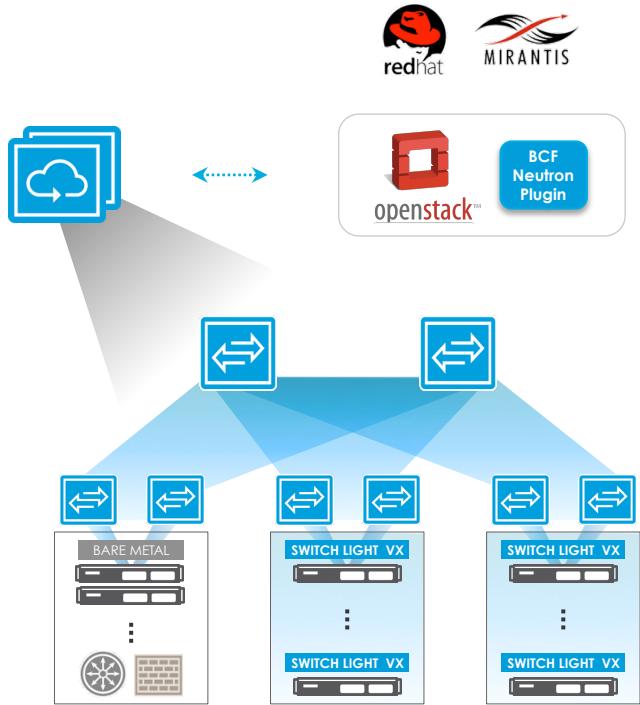
- Virtual network for BCF admin
- Physical network for vCenter admin

Troubleshooting

- VM to VM level traceability for both admins
- Test path (fabric trace) utility available to both admins
- VCenter events captured in analytics

INTRODUCTION TO BIG CLOUD FABRIC

OpenStack Integration



Integration

- One place to configure end-to-end networking
- Brings virtual switches into the BCF domain

Automation

- Auto learn Neutron virtual network configurations (networks, subnets, routers & floating IPs)
- Auto detect compute nodes (CN)
- Auto configure CN uplinks, tenant, L2 segments & VMs

Visibility

- Virtual network for BCF admin
- Physical network for OpenStack admin

Troubleshooting

- VM to VM level traceability for both admins
- Test path (fabric trace) utility available to both admins
- OpenStack events captured in analytics

INTRODUCTION TO BIG CLOUD FABRIC

Analytics

Pre-configured dashboards

- Physical (Controller / Switch)
- Logical (Tenant / Segment / Endpoints / ...)

Search logs

- Events, errors, state changes, ...
- Configuration changes (REST, CLI or GUI)
- Command history

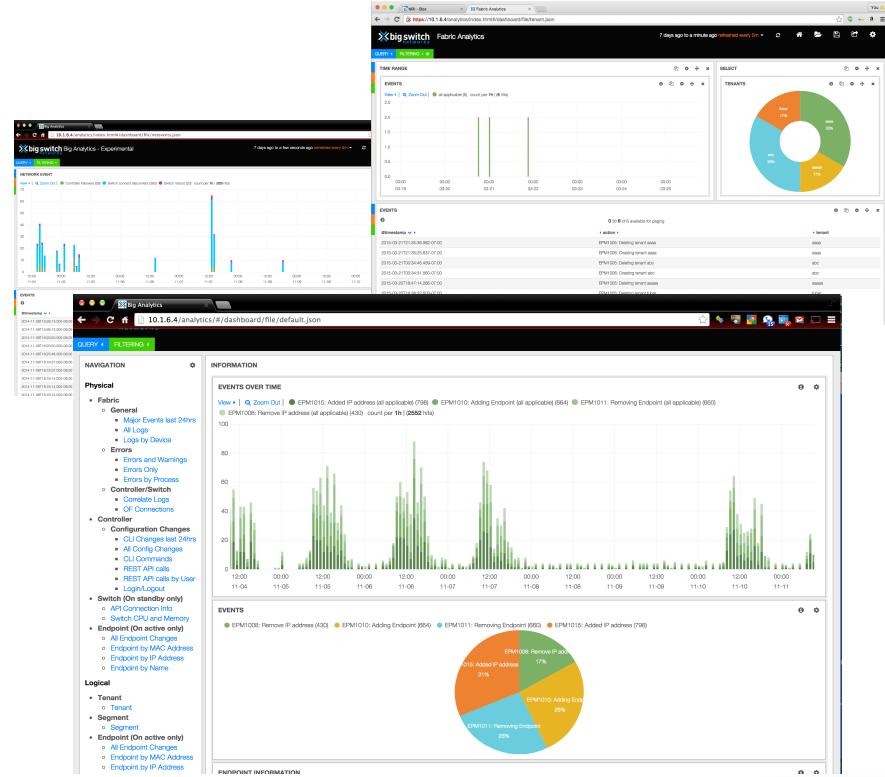
Visualize Data

- Switch / Interface / Tenant utilization
- Perform trend analysis

Powerful query language

- Drill down data sets using simple mouse clicks
- Write your own query strings

Build new and custom dashboards



INTRODUCTION TO BIG CLOUD FABRIC

High Availability

Switch Redundancy

- Redundant spine
- Leaf Groups for redundancy at TOR

Link Redundancy

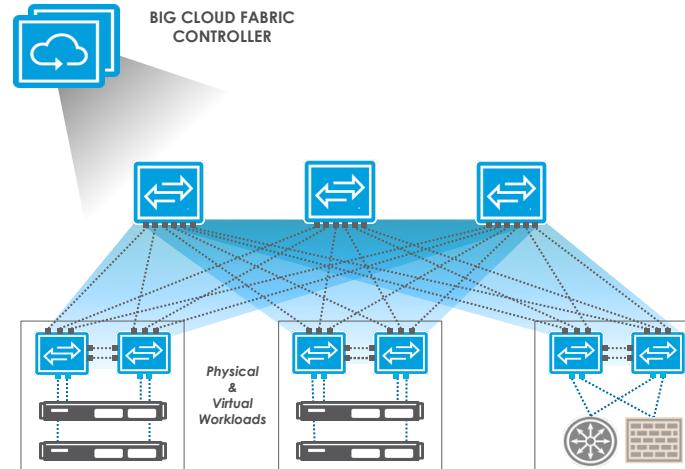
- Every leaf to every spine
- Dual inter-switch links
- Dual server and external router links

Controller Redundancy

- Active / Standby
- Headless Mode
 - Controller failures
 - Control plane failure
 - Control plane interface failure

Distributed Routing

- Logical routers on each switch



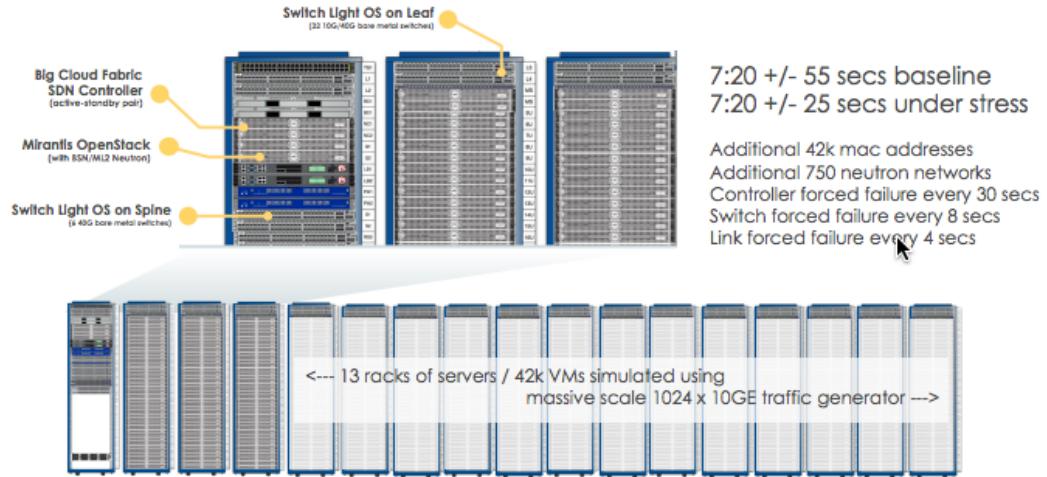
Comprehensive HA
No single point of failure

INTRODUCTION TO BIG CLOUD FABRIC

Resiliency

Chaos Monkey Testing

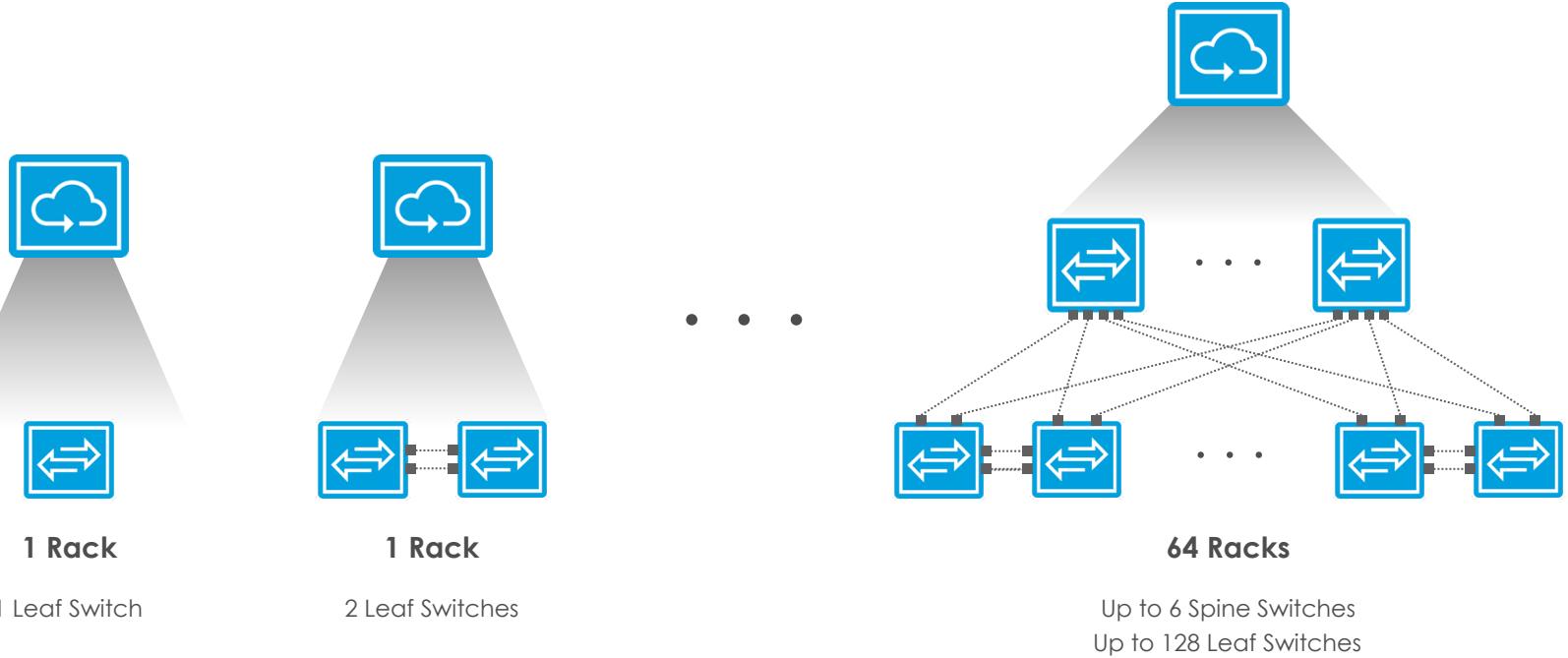
- 16 rack POD (32 leaf, 6 spine)
- 42k simulated End-points/VMs
- 640+ forced component failures
 - Controller fail-over every 30 seconds
 - Switch fail-over every 8 seconds
 - Link fail-over every 4 seconds



Conclusion: 640 component failures in 30 minutes with no impact on application performance

INTRODUCTION TO BIG CLOUD FABRIC

Fabric Topologies / Deployments



INTRODUCTION TO BIG CLOUD FABRIC

Controllers

Controller	Platform	Configuration	Deployment
VM	ESX 6.0 Ubuntu 14.04	12 vCPU (min 1 GHz) 56 GB RAM, 300 GB HDD 4 vNIC	Lab
VM	RHEL 7.2	12 vCPU (min 1 GHz) 56 GB RAM, 300 GB HDD 4 vNIC	Production (3 Racks, 120 Servers)
HW – Standard	Dell R430	12 Cores 64 GB RAM, 2TB HDD 4x1G, 2x10G NIC	Production (P Fabric)
HW – Large	Dell R430	24 Cores 64 GB RAM, 2TB HDD 4x1G, 2x10G NIC	Production (P+V Fabric)

INTRODUCTION TO BIG CLOUD FABRIC

Ethernet Switches

Processor	Configuration	Accton	Models		Role
			Dell	HPE	
Trident II	48x10G, 6x40G	AS5712-54X	S4048-ON		Spine / Leaf
	32x40G	AS6712-32X	S6000-ON		Spine / Leaf
Trident II+	48x10G, 6x40G	AS5812-54X		AL6921F	Spine / Leaf
	48x10GT, 6x40G	AS5812-54T	S4048T-ON	AL6921T	Spine / Leaf
	32x40G	AS6812-32X	S6010-ON	AL6941F	Spine / Leaf
Tomahawk	64x40G		S6100-ON		Spine
	32x100G **	AS7712-32X	Z9100-ON	AL6960F	Spine / Leaf

** 100G switches support 10G, 25G, 40G, and 100G speeds

INTRODUCTION TO BIG CLOUD FABRIC

Innovations & Summary

SDN Controller

- Protocol free fabric
- Centralized management & control

Zero Touch Fabric

- Auto switch discovery
- Auto fabric formation

Tenant Aware Fabric

- Tenant centric configurations
- Fine-grained inter-tenant access control

Distributed Logical Routing

- Tenant router in every leaf switch
- System tenant router in every spine switch

Service Aware Fabric

- Service insertion at Inter-segment and inter-tenant level
- Insertion of physical or virtual services for physical or virtual workloads

Headless Mode

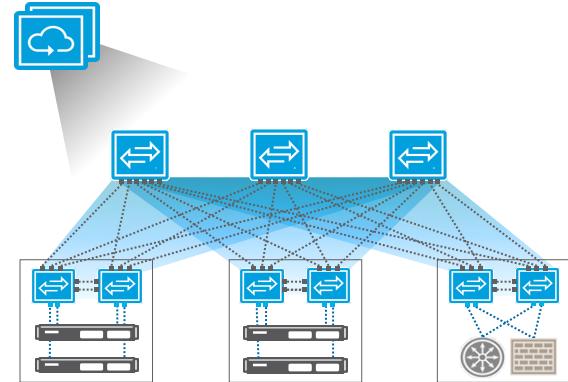
- Resiliency during double failure
- No state timeouts or expunged programming

Fabric LAG

- ECMP like L2 LAG
- Load balances across spine & leaf switches

Fabric View

- Advanced Multi-node Troubleshooting, Analytics & Telemetry
- VM to VM visibility



Questions?



Module 2 – BCF Access and Configuration

BCF ACCESS AND CONFIGURATION

Module Outline

- Access Modes
 - GUI
 - CLI
 - REST
- Configuration Management
- System Configuration
- **Lab – Hands-on with BCF Environment**

BCF ACCESS AND CONFIGURATION

GUI – Dashboard

- Direct browser to controller's IP address
- Dashboard view
 - Fabric inventory & status
 - Version & uptime
 - Controller utilizations
 - Switch utilizations
 - Top Talkers
 - Tenants, Segments, Interfaces
- To return, click on the Big Switch logo



BCF ACCESS AND CONFIGURATION

GUI – Menu

[Fabric](#)[Logical](#)[Endpoints](#)[Visibility](#)[Settings](#)[Integration](#)[Security](#)[Default admin](#)

- Centralized view:
 - **Fabric** – Manage switches, interfaces, interface groups
 - **Logical** – Manage tenant, segments, routes, policy, next-hop groups
 - **Endpoints** – Manage fabric endpoints (logical representations of servers and hosts)
 - **Visibility** – Monitor and troubleshoot fabric configuration and activity
 - **Settings** – Configure the logging of system and fabric activity to a remote server (SNMP, TACACS, AAA, Logging, etc.)
 - **Integration** – Orchestration integration Configuration (vCenter)
 - **Security** – Manage users and groups
 - **Current user name** – Current user (click to sign out)

BCF ACCESS AND CONFIGURATION

GUI – Topology

Fabric Logical Endpoints Visibility Settings Integration Security Default admin



Actions

Reboot All Connected

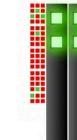
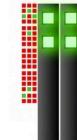
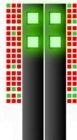
Controller Nodes



Spine Switches



Leaf Switches



All Virtual Switches

BCF ACCESS AND CONFIGURATION

GUI – Topology

Fabric Logical Endpoints Visibility Settings Integration Security Default admin

SHOW LINKS MODE

Spine Switch

Name s1
MAC 70:72:
Connected ✓
Connected Since 2015-12
Fabric Status ✓
Management IP Address fe80::
Allocated IP Address —
Port 45290
Interfaces 32

Actions

- Visualize
- Configure
- Clear Configuration
- Reset Connection
- Reboot
- Shutdown
- Beacon

Spine Switches

Leaf Switches

The diagram illustrates a network topology with two active Spine switches (one labeled 'Active' and one labeled 'Standby') and multiple Leaf switches. The Spine switches are interconnected by green links, forming a central fabric. Each Spine switch is connected to several Leaf switches, which are represented by vertical bars with red and green segments. The 'Active' Spine switch is connected to four Leaf switches, while the 'Standby' Spine switch is connected to three. The Leaf switches are arranged in two rows, with some having two segments and others having three or four segments.

BCF ACCESS AND CONFIGURATION

GUI – Topology

Fabric Logical Endpoints Visibility Settings Integration Security Default admin

SHOW LINKS MODE

Port 60311
Interfaces 54
Leaf Group R2

Actions

- Visualize
- Configure
- Clear Configuration
- Split From Leaf Group
- Reset Connection
- Reboot
- Shutdown
- Beacon

Leaf Group Peers

Name	MAC
r2i2	70:72:cf:c7:00:63

Interfaces

Controller Nodes

Spine Switches

Leaf Switches

The screenshot shows the Big Switch Networks BCF Access and Configuration GUI. The top navigation bar includes Fabric, Logical, Endpoints, Visibility, Settings, Integration, Security, and Default admin. A 'SHOW LINKS MODE' button is highlighted with a red circle. The main interface displays a network topology with two controller nodes labeled 'Active' and 'Standby'. These nodes are connected to a central spine switch, which in turn connects to several leaf switches. The leaf switches are shown as vertical bars with red and green segments. On the left side, there are sections for Actions (Visualize, Configure, etc.), Leaf Group Peers (listing r2i2 with MAC 70:72:cf:c7:00:63), and Interfaces. A sidebar on the left shows Port 60311, Interfaces 54, and Leaf Group R2.

BCF ACCESS AND CONFIGURATION

GUI – Logical Overview

[Fabric](#)[Logical](#)[Endpoints](#)[Visibility](#)[Settings](#)[Integration](#)[Security](#)[Default admin](#)

Logical Overview

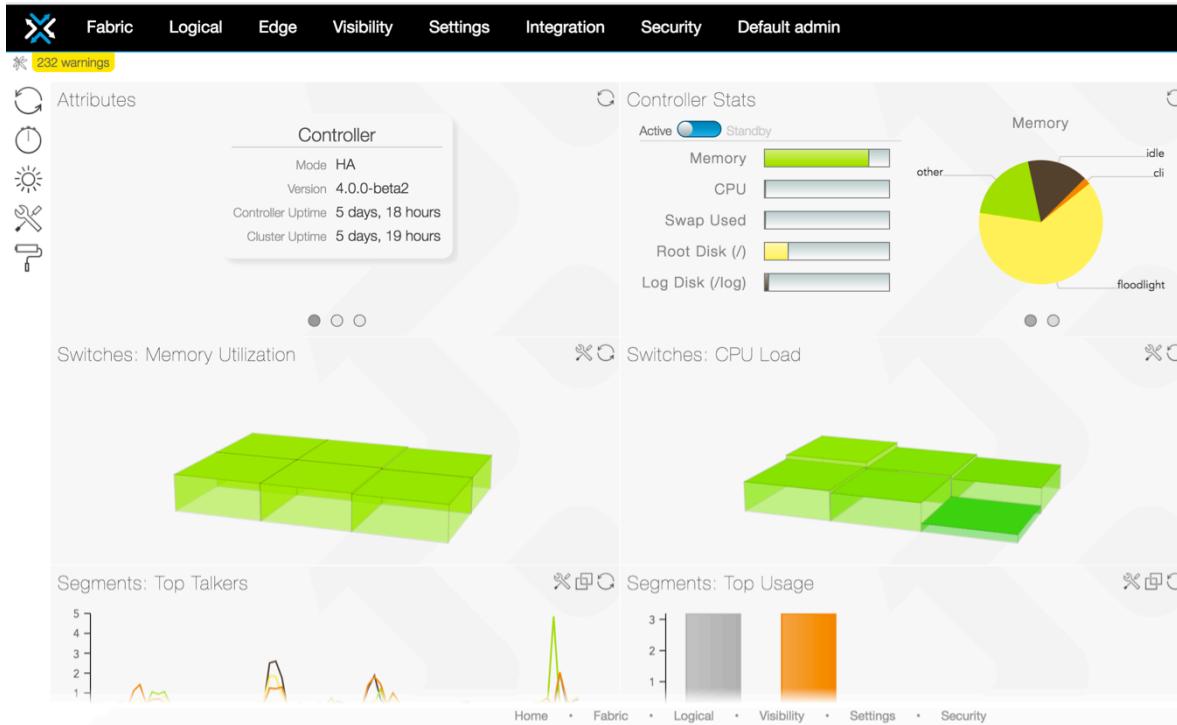
Background: [Color Swatch]

Tenants	Segments
mgmt	S external Tenant: mgmt
vcenter-u2	S mgmt1 Tenant: mgmt
	S mgmt2 Tenant: mgmt
	S new Tenant: mgmt
	S os-103 Tenant: mgmt
	S os-104 Tenant: mgmt
	S vcenter-u2-100 Tenant: vcenter-u2 vSphere portgroups (1), ...
	S vcenter-u2-200 Tenant: vcenter-u2 vSphere portgroups (1), ...

The diagram illustrates the logical connections between the system node (top) and the mgmt and vcenter-u2 nodes (bottom). A solid line connects the system node to the mgmt node, while dashed lines connect the system node to both the mgmt and vcenter-u2 nodes.

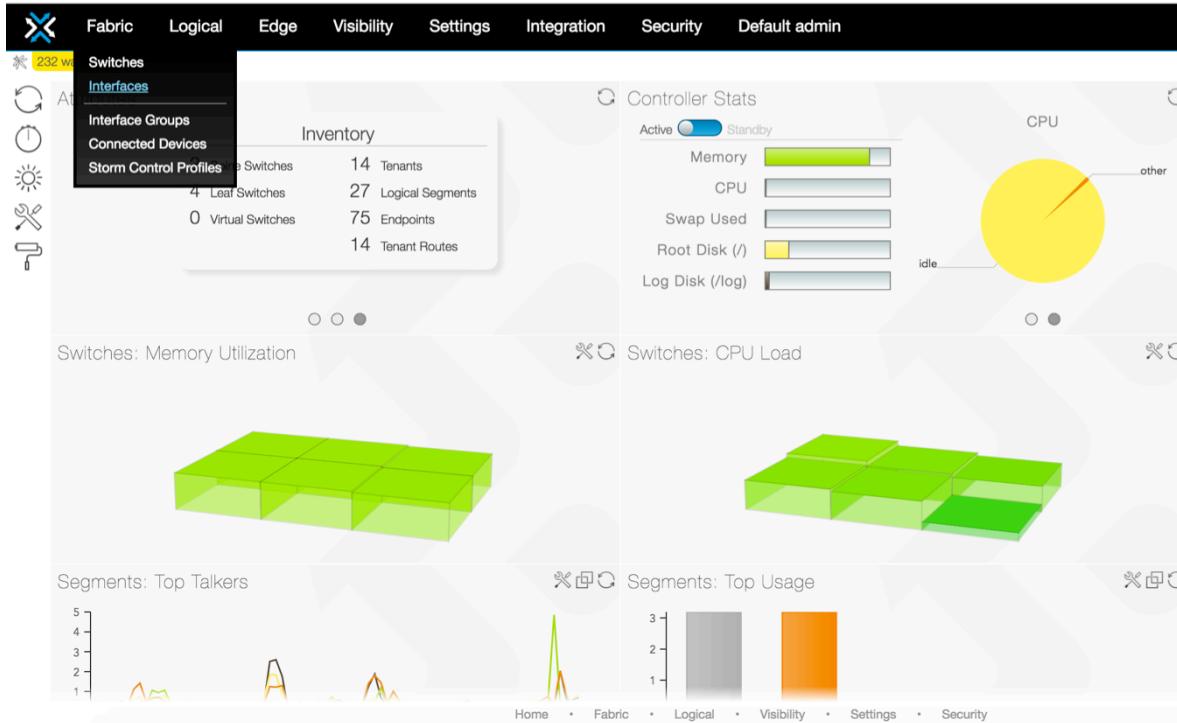
BCF ACCESS AND CONFIGURATION

GUI – Walk Through



BCF ACCESS AND CONFIGURATION

GUI – Walk Through



BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. At the top, there is a navigation bar with tabs: Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation bar, the URL is displayed as Big Cloud Fabric™ • Fabric • Interfaces.

The main content area is titled "Interfaces". It features a table with the following columns: Switch, Switch MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. The table has 29 rows, each representing an interface on the "dt-spine2" switch. The "Status" column uses green checkmarks for "Up" and red X's for "Link Down". The "Spine Switch" and "Leaf Switch" columns are mostly blank or show a dash. The "Virtual Switch" and "Interface Group" columns are also mostly blank or show a dash.

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1	—	✓ Up	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet11	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet12	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet13	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet14	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet15	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet16	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet17	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet18	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet19	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet2	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet20	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet21	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet22	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet23	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet24	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet25	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet26	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet27	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet28	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet29	—	✗ Link Down	✓	—	—	—

At the bottom of the table, there are navigation links: Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface for managing interfaces. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ • Fabric • Interfaces.

The main area is titled "Interfaces" and displays a table of interface configurations. The columns are: Switch, Switch MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. There are 29 rows listed, each corresponding to an interface named "ethernet1" through "ethernet29". The "Status" column indicates the link status: "Up" for ethernet1 and "Link Down" for all other interfaces. The "Spine Switch" column contains checkmarks for most interfaces, while "ethernet11" and "ethernet29" have red X marks. The "Leaf Switch" and "Virtual Switch" columns are mostly blank or contain dashes. The "Interface Group" column is also mostly blank or contains dashes.

A context menu is open over the first row (ethernet1), listing the following options:

- Shutdown
- Disable BPDU Guard
- Set Breakout Property
- Edit Description
- Set Storm Control Profile

At the bottom of the table, there are navigation links for Home, Fabric, Logical, Visibility, Settings, and Security.

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1	—	✓ Up	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet11	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet12	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet13	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet14	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet15	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet16	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet17	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet18	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet19	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet2	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet20	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet21	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet22	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet23	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet24	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet25	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet26	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet27	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet28	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet29	—	✗ Link Down	✓	—	—	—

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ > Fabric > Interfaces.

The main area is titled "Interfaces". It features a table with the following columns: MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. The table contains 29 rows, each representing an interface. A context menu is open over the first few rows, showing options: Show/Hide Columns, Shutdown Selected Interfaces, and Startup Selected Interfaces.

MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
00:0c:29:00:00:5e	ethernet1	—	✓ Up	✓	—	—	—
00:0c:29:00:00:5e	ethernet11	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet12	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet13	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet14	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet15	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet16	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet17	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet18	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet19	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet2	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet20	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet21	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet22	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet23	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet24	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet25	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet26	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet27	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet28	—	✗ Link Down	✓	—	—	—
00:0c:29:00:00:5e	ethernet29	—	✗ Link Down	✓	—	—	—

At the bottom of the table are navigation links: Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface with the 'Fabric' tab selected. In the center, there's a table titled 'Interfaces' listing various network interfaces. A modal window titled 'Configure Table Columns' is open, allowing users to select which columns to display. The left side of the modal lists columns grouped by category, and the right side shows individual checkboxes for each column.

Switch	Switch MAC	Interface Name
dt-spine2	70:72:cf:ae:a0:5e	ethernet1
dt-spine2	70:72:cf:ae:a0:5e	ethernet11
dt-spine2	70:72:cf:ae:a0:5e	ethernet12
dt-spine2	70:72:cf:ae:a0:5e	ethernet13
dt-spine2	70:72:cf:ae:a0:5e	ethernet14
dt-spine2	70:72:cf:ae:a0:5e	ethernet15
dt-spine2	70:72:cf:ae:a0:5e	ethernet16
dt-spine2	70:72:cf:ae:a0:5e	ethernet17
dt-spine2	70:72:cf:ae:a0:5e	ethernet18
dt-spine2	70:72:cf:ae:a0:5e	ethernet19
dt-spine2	70:72:cf:ae:a0:5e	ethernet2
dt-spine2	70:72:cf:ae:a0:5e	ethernet20
dt-spine2	70:72:cf:ae:a0:5e	ethernet21
dt-spine2	70:72:cf:ae:a0:5e	ethernet22
dt-spine2	70:72:cf:ae:a0:5e	ethernet23
dt-spine2	70:72:cf:ae:a0:5e	ethernet24
dt-spine2	70:72:cf:ae:a0:5e	ethernet25
dt-spine2	70:72:cf:ae:a0:5e	ethernet26
dt-spine2	70:72:cf:ae:a0:5e	ethernet27
dt-spine2	70:72:cf:ae:a0:5e	ethernet28
dt-spine2	70:72:cf:ae:a0:5e	ethernet29

Check Columns to Show

Category	Column	Action
Connected Device	All Columns	select / unselect
	BPDU Guard	select / unselect
	Connected Device	select / unselect
	Connected Device DCBX TLV Info	select / unselect
	Features	select / unselect
	QoS PFC Info	select / unselect
	Storm Control	select / unselect
	Switch	<input checked="" type="checkbox"/>
	Switch MAC	<input checked="" type="checkbox"/>
	Interface Name	<input checked="" type="checkbox"/>
Description	<input checked="" type="checkbox"/>	
Status	<input checked="" type="checkbox"/>	
Consolidated Admin/Link Status	<input type="checkbox"/>	
Admin Status	<input type="checkbox"/>	
Link Status	<input type="checkbox"/>	
Hardware Address	<input type="checkbox"/>	
Number	<input type="checkbox"/>	
Spine Switch	<input checked="" type="checkbox"/>	
Leaf Switch	<input checked="" type="checkbox"/>	
Virtual Switch	<input checked="" type="checkbox"/>	
Interface Group	<input checked="" type="checkbox"/>	
Storm Control Profile	<input type="checkbox"/>	
Storm Control State	<input type="checkbox"/>	
BPDU Guard Enabled	<input type="checkbox"/>	
BPDU Guard State	<input type="checkbox"/>	
Breakout	<input type="checkbox"/>	
Auto-Negotiation	<input type="checkbox"/>	
PFC Match	<input type="checkbox"/>	
PFC Configured	<input type="checkbox"/>	
PFC Received	<input type="checkbox"/>	
iSCSI Match	<input type="checkbox"/>	
iSCSI Configured	<input type="checkbox"/>	
iSCSI Received	<input type="checkbox"/>	
Connected Device	<input type="checkbox"/>	
Connected Interface Group	<input type="checkbox"/>	
Connected Device Description	<input type="checkbox"/>	
Connected Device Chassis ID	<input type="checkbox"/>	
Connected Device Port ID	<input type="checkbox"/>	
Connected Device Port Description	<input type="checkbox"/>	
Connected Device Protocol	<input type="checkbox"/>	
Connected Device DCBX PFC TLV Sent to Device	<input type="checkbox"/>	
Connected Device Application TLVs Sent to Device	<input type="checkbox"/>	
Connected Device DCBX PFC TLV Rec'd from Device	<input type="checkbox"/>	
Connected Device Application TLVs Rec'd from Device	<input type="checkbox"/>	

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ • Fabric • Interfaces. The main content area is titled "Interfaces". It features a table with 29 rows, each representing an interface on a switch named "dt-spine2". The columns are: Switch, Switch MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. The "Status" column contains icons indicating the link status: a green checkmark for "Up" and a red X for "Link Down". Most interfaces show "Link Down". The "Spine Switch" and "Leaf Switch" columns are mostly blank or contain a single dash. The "Virtual Switch" and "Interface Group" columns also contain dashes. A "Filter table rows" input field is located at the top left of the table. At the bottom of the table, there are navigation links for Home, Fabric, Logical, Visibility, Settings, and Security.

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1	—	✓ Up	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet11	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet12	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet13	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet14	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet15	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet16	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet17	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet18	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet19	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet2	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet20	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet21	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet22	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet23	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet24	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet25	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet26	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet27	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet28	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet29	—	✗ Link Down	✓	—	—	—

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface for managing interfaces. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ > Fabric > Interfaces.

The main area is titled "Interfaces". It features a table with the following columns: Switch, Switch MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. A tooltip "Add filter for this exact value" is visible over the "Description" column header. The table contains 29 rows, each representing an interface on a switch named "dt-spine2". All interfaces listed have a status of "Link Down".

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet11		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet12		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet13		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet14		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet15		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet16		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet17		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet18		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet19		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet2		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet20		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet21		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet22		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet23		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet24		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet25		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet26		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet27		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet28		✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet29		✗ Link Down	✓	—	—	—

Below the table are navigation links: Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ > Fabric > Interfaces. The main content area is titled "Interfaces". A search bar contains the filter "name==ethernet1". The table displays the following data:

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1	—	✓ Up	✓	—	—	—
dt-spine1	70:72:cf:ae:a5:f4	ethernet1	—	✓ Up	✓	—	—	—
dt-leaf2b	70:72:cf:b5:ff:ec	ethernet1	—	✓ Up	—	✓	—	arista-2010
dt-leaf2a	70:72:cf:b7:6d:12	ethernet1	—	✓ Up	—	✓	—	arista-2010
dt-leaf1b	70:72:cf:bc:c1:2c	ethernet1	—	✓ Up	—	✓	—	arista-2009
dt-leaf1a	70:72:cf:b7:6d:f0	ethernet1	—	✓ Up	—	✓	—	arista-2009

At the bottom left, the timestamp is Nov 4, 2016, 1:52:46pm PDT. At the bottom right, there are pagination controls: Show: 10, 25, 100, (1 - 6 / 6).

Navigation links at the bottom include Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ > Fabric > Interfaces. The main content area is titled "Interfaces". A search bar contains the filter "name:==ethernet1". The table displays the following data:

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1	—	✓ Up	✓	—	—	—
dt-spine1	70:72:cf:ae:a5:f4	ethernet1	—	✓ Up	✓	—	—	—
dt-leaf2b	70:72:cf:b5:ff:ec	ethernet1	—	✓ Up	—	✓	—	arista-2010
dt-leaf2a	70:72:cf:b7:6d:12	ethernet1	—	✓ Up	—	✓	—	arista-2010
dt-leaf1b	70:72:cf:bc:c1:2c	ethernet1	—	✓ Up	—	✓	—	arista-2009
dt-leaf1a	70:72:cf:b7:6d:f0	ethernet1	—	✓ Up	—	✓	—	arista-2009

Below the table, the date and time are shown as Nov 4, 2016, 1:52:46pm PDT. To the right, there are buttons for "Show: 10 25 100" and "(1 - 6 / 6)". The bottom navigation bar includes links for Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

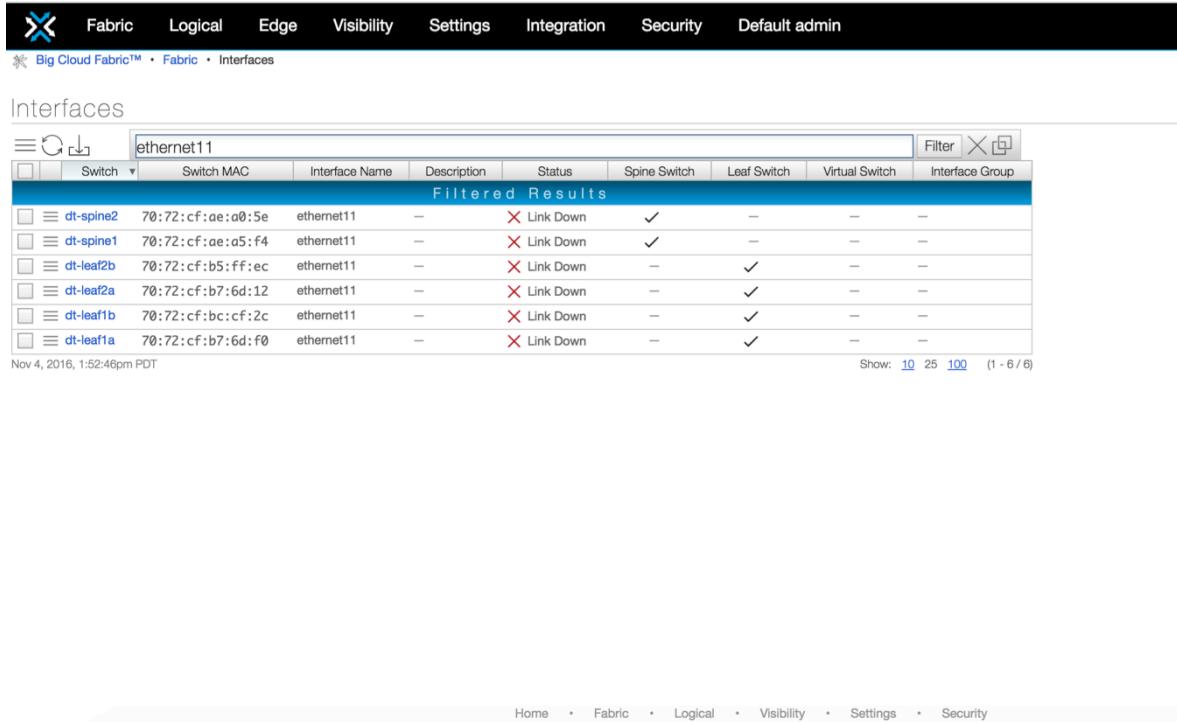
GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ • Fabric • Interfaces. The main content area is titled "Interfaces". It features a table with 29 rows, each representing an interface on a switch named "dt-spine2". The columns are: Switch, Switch MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. The "Status" column uses green checkmarks for "Up" and red X's for "Link Down". The "Spine Switch" and "Leaf Switch" columns show mostly blank or "—" entries. The "Virtual Switch" and "Interface Group" columns also show mostly blank or "—" entries.

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet1	—	✓ Up	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet11	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet12	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet13	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet14	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet15	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet16	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet17	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet18	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet19	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet2	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet20	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet21	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet22	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet23	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet24	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet25	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet26	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet27	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet28	—	✗ Link Down	✓	—	—	—
dt-spine2	70:72:cf:ae:a0:5e	ethernet29	—	✗ Link Down	✓	—	—	—

BCF ACCESS AND CONFIGURATION

GUI – Walk Through



The screenshot shows the Big Cloud Fabric™ GUI with the 'Fabric' tab selected. The main content area is titled 'Interfaces' and displays a table of network interfaces. A search bar at the top of the table is set to 'ethernet11'. The table has columns for Switch, Switch MAC, Interface Name, Description, Status, Spine Switch, Leaf Switch, Virtual Switch, and Interface Group. The status column shows 'Link Down' for all entries. The table header includes a 'Filtered Results' label. At the bottom of the table, there are pagination controls showing 'Show: 10 25 100 (1 - 6 / 6)'.

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet11	—	✗ Link Down	✓	—	—	—
dt-spine1	70:72:cf:ae:a5:f4	ethernet11	—	✗ Link Down	✓	—	—	—
dt-leaf2b	70:72:cf:b5:ff:ec	ethernet11	—	✗ Link Down	—	✓	—	—
dt-leaf2a	70:72:cf:b7:6d:12	ethernet11	—	✗ Link Down	—	✓	—	—
dt-leaf1b	70:72:cf:bc:cf:2c	ethernet11	—	✗ Link Down	—	✓	—	—
dt-leaf1a	70:72:cf:b7:6d:f0	ethernet11	—	✗ Link Down	—	✓	—	—

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Cloud Fabric™ interface. The top navigation bar includes links for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation is a breadcrumb trail: Big Cloud Fabric™ > Fabric > Interfaces. The main title is "Interfaces". A search bar contains the text "ethernet11". The table displays the following data:

Switch	Switch MAC	Interface Name	Description	Status	Spine Switch	Leaf Switch	Virtual Switch	Interface Group
dt-spine2	70:72:cf:ae:a0:5e	ethernet11	—	✗ Link Down	✓	—	—	—
dt-spine1	70:72:cf:ae:a5:f4	ethernet11	—	✗ Link Down	✓	—	—	—
dt-leaf2b	70:72:cf:b5:ff:ec	ethernet11	—	✗ Link Down	—	✓	—	—
dt-leaf2a	70:72:cf:b7:6d:12	ethernet11	—	✗ Link Down	—	✓	—	—
dt-leaf1b	70:72:cf:bc:cf:2c	ethernet11	—	✗ Link Down	—	✓	—	—
dt-leaf1a	70:72:cf:b7:6d:f0	ethernet11	—	✗ Link Down	—	✓	—	—

Below the table, the text "Nov 4, 2016, 2:00:27pm PDT" is displayed. To the right, there are buttons for "Show: 10 25 100" and "(1 - 6 / 6)". The bottom navigation bar includes links for Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Switch Network (BCF) GUI interface. At the top, there is a navigation bar with tabs: Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation bar, the path is shown as Fabric • Switches • dt-leaf1a. The main title is LEAF SWITCH dt-leaf1a [ACCTON AS5710].

The central part of the screen displays a 3D rendering of an ACCTON AS5710 switch. The switch has two horizontal rows of ports, each labeled with a number from 1 to 54. The ports are color-coded: green for active ports and yellow for ports that are either disabled or have a warning. A legend at the bottom of the switch image provides a key for these colors.

Below the switch image is a detailed properties panel. The top row of the panel contains several tabs: Diagnostic, Properties, Actions, Environment, Interfaces, Links, Lags, vSwitches, Leaf Group, Interface Groups, and sFlow. The 'Properties' tab is currently selected.

The properties section is divided into two columns:

Name	Connected	MAC Address	Storm Control Profile				
dt-leaf1a	✓	70:72:cf:b7:6d:f0	—				
Fabric Role	Leaf	Admin Status	Up	Management IP Address	fe80::7272:cfffe:feb7:6df0%3	Storm Control State	NA
Leaf Group	rack1	Fabric Status	✓	Management Port	44973		
Platform	powerpc-accton-as5710-54x-r0b	Interfaces	55	Allocated IP Address	10.1.6.50/24		
Serial Number	571054X1419022						

At the bottom of the properties panel, there is a footer with links: Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the Big Switch Network (BCF) GUI interface. At the top, there is a navigation bar with icons for Fabric, Logical, Edge, Visibility, Settings, Integration, Security, and Default admin. Below the navigation bar, the path is shown as Fabric > Switches > dt-leaf1a. The main title is "LEAF SWITCH dt-leaf1a [ACCTON AS5710]".

The central part of the screen displays a 3D rendering of an ACCTON AS5710 switch. The switch has two front panels, each with 28 ports. The ports are numbered 1 through 54. Some ports are highlighted in yellow, indicating specific configurations or status.

Below the switch image, there is a toolbar with several tabs: Diagnostic, Properties, Actions, Environment, Interfaces, Links, Lags, vSwitches, Leaf Group, Interface Groups, and sFlow. The "Actions" tab is currently selected, indicated by a blue background.

To the left of the toolbar, there is a list of actions with corresponding icons:

- Configure
- Clear Configuration
- Split From Leaf Group
- Reset Connection
- Reboot
- Shutdown
- Beacon
- Set Storm Control Profile

At the bottom of the screen, there is a footer navigation bar with links to Home, Fabric, Logical, Visibility, Settings, and Security.

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

Fabric Logical Edge Visibility Settings Integration Security Default admin

Fabric • Switches • dt-leaf1a

LEAF SWITCH dt-leaf1a [ACCTON AS5710]



Diagnostic Properties Actions **Environment** Interfaces Links Lags vSwitches Leaf Group Interface Groups sFlow

Fans

Label	State	Status	Speed	Airflow	PSU Fan
Chassis Fan 1	Present	Running	High	Front-to-Back	—
Chassis Fan 2	Present	Running	High	Front-to-Back	—
Chassis Fan 3	Present	Running	High	Front-to-Back	—
Chassis Fan 4	Present	Running	High	Front-to-Back	—
Chassis Fan 5	Present	Running	High	Front-to-Back	—
Chassis PSU-1 Fan 1	Present	Running	Medium	Front-to-Back	✓

Thermal Sensors

Label	Status	Temperature (°C)	PSU Sensor
Chassis Thermal Sensor 1 (Sensor on CPU board)	Functional	28.0	—
Chassis Thermal Sensor 2 (Measurement point on CPU)	Functional	51.0	—
Chassis Thermal Sensor 3 (Front middle)	Functional	27.5	—
Chassis Thermal Sensor 4 (Rear right)	Functional	25.5	—
Chassis Thermal Sensor 5 (Front right)	Functional	21.5	—
PSU-1 Thermal Sensor 1	Functional	27.0	✓

Power Supply Units

Label	State	Status	Model	Type	Voltage (In)	Voltage (Out)	Current (In)	Current (Out)	Power (In)	Power (Out)	Fans	Thermal Sensors
PSU-1	Present	Running	CPR-4011-4M11	AC	206.50	12.00	0.50	8.10	111.00	98.00	Chassis PSU-1 Fan 1	PSU-1 Thermal Sensor 1
PSU-2	Present	Not Running	—	—	—	—	—	—	—	—	—	—

Home • Fabric • Logical • Visibility • Settings • Security

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

Fabric Logical Edge Visibility Settings Integration Security Default admin

Fabric • Switches • dt-leaf1a

LEAF SWITCH dt-leaf1a [ACCTON AS5710]



Diagnostic Properties Actions Environment Interfaces **Links** Lags vSwitches Leaf Group Interface Groups sFlow

Source	Destination	Link Type	Link Direction
dt-leaf1a / ethernet48	dt-leaf1b / ethernet47	peer	bidirectional
dt-spine1 / ethernet1	dt-leaf1a / ethernet49	leaf-spine	bidirectional
dt-spine2 / ethernet1	dt-leaf1a / ethernet51	leaf-spine	bidirectional

Show: [10](#) [25](#) [100](#) [All](#) (1 - 3 / 3)

Home • Fabric • Logical • Visibility • Settings • Security

BCF ACCESS AND CONFIGURATION

GUI – Walk Through

The screenshot shows the BIG-IP Configuration Utility interface for a leaf switch named "dt-leaf1a".

Header: Fabric, Logical, Edge, Visibility, Settings, Integration, Security, Default admin.

Breadcrumbs: Fabric • Switches • dt-leaf1a

Switch Model: LEAF SWITCH dt-leaf1a [ACCTON AS5710]

Switch Image: A 48-port ACCTON AS5710 leaf switch with ports numbered 1 to 54. Port 51 is highlighted in blue.

Table: Shows network links between the switch and other devices.

Source	Destination	Link Type	Link Direction
dt-leaf1a / ethernet48	dt-leaf1b / ethernet47	peer	bidirectional
dt-spine1 / ethernet1	dt-leaf1a / ethernet49	leaf-spine	bidirectional
dt-spine2 / ethernet1	dt-leaf1a / ethernet51	leaf-spine	bidirectional

Show: 10 25 100 All (1 - 3 / 3)

Interface Details: A modal window displays details for interface "ethernet51".

Name	ethernet51
Port Number	51
Hardware Address	5c:16:c7:12:dc:c9

Buttons: Diagnostic, Properties, Actions, Environment.

Details:
Admin Status: ✓ Up
Link Status: ✓ Up
Interface Group: —
Link Speed: 40 Gbit/s
Mode: Full Duplex
Link Type: Copper
Breakout: —
Auto-Negotiation: NA

Footer: Home • Fabric • Logical • Visibility • Settings • Security

BCF ACCESS AND CONFIGURATION

CLI – Overview

- CLI is similar to commonly used model in the industry
- Hierarchical configuration
- Tree-like traversal
- Context sensitive help
- Client of REST API
- Access to standard Linux utilities
- Case-sensitive

```
! tenant
tenant BLUE
logical-router
  route 0.0.0.0/24 tenant system
  interface segment web
    ip address 10.1.1.254/24

  interface segment app
    ip address 10.1.3.254/24

  segment web
    member interface-group ig-bm0 vlan 20

  segment app
    member interface-group ig-bm5 vlan 40
```

BCF ACCESS AND CONFIGURATION

CLI – Access

- Logging in:
 - Use console window of the controller
 - Use SSH to remotely access the controller
 - Use management IP address set during installation
 - May use SSH keys for logging in without password

```
my-desktop$ ssh admin@10.2.24.131
Big Cloud Fabric Appliance 3.1.0 (bcf-3.1.0 #166)
Log in as 'admin' to configure

admin@10.2.24.131's password:
Last login: Wed Dec 16 18:50:53 2015 from 10.1.2.106
Big Cloud Fabric Appliance 3.1.0 (bcf-3.1.0 #166)
Logged in as admin, 2015-12-16 18:53:28.602000 UTC, auth from 10.1.2.106
controller>
```

BCF ACCESS AND CONFIGURATION

CLI – Modes

- CLI operates in one of three modes:
 - **Login**: Initial login mode. Prompt ends with ">" sign

```
controller>  
controller>
```

- **Enable**: Superset of Login mode. Prompts ends with "#" sign

```
controller> enable  
controller#
```

- **Config**: Superset of Enable mode. Prompt ends with "(config)#"

```
controller# config  
controller(config)#
```

- **Config-sub-mode**: Superset of Config mode. Prompt indicates sub-mode

```
controller(config)# tenant blue  
controller(config-tenant)# logical-router  
controller(config-tenant-lrouter)# policy-list abc  
controller(config-tenant-lrouter-policy)#
```

BCF ACCESS AND CONFIGURATION

CLI – Modes

- Exit current mode or sub-mode using “**exit**” keyword

```
controller(config-tenant-lrouter-policy)# exit  
controller(config-tenant-lrouter)# exit  
controller(config-tenant)# exit  
controller(config)# exit  
controller# exit  
controller> exit  
Connection to 10.2.24.131 closed.
```

- Exit config mode from any sub-mode using “**end**” keyword

```
controller(config-tenant-lrouter-policy)# end  
controller#
```

BCF ACCESS AND CONFIGURATION

CLI – Help

- Context sensitive help
 - “**help or ?**”:
 - All available commands – cmd list from all previous modes traversed
 - Commands – cmd list specific to current mode/sub-mode

```
controller(config)# help
For help on specific commands: help <command>
All available commands:
    ... <list of commands from all previously traversed modes> ...
Commands:
    ... <list of commands specific to current mode/sub-mode> ...
controller(config)#

```

- “**help <command>**”: displays help text for a specific command
- “**cmd ?**”: displays completion options for the given command

BCF ACCESS AND CONFIGURATION

CLI – Navigation

- Control-key sequences for the command line:
 - <CTRL>-B: back one character (or left arrow key)
 - <CTRL>-F: forward one character (or right arrow key)
 - <CTRL>-P: previous command (or up arrow key)
 - <CTRL>-N: next command (or down arrow key)
 - <CTRL>-A: beginning of line
 - <CTRL>-E: end of line
 - <CTRL>-W: delete last word
 - <CTRL>-R: search for text in history
 - <CTRL>-U: erase line

BCF ACCESS AND CONFIGURATION

CLI – Redirecting output, multiple commands

- Redirect output to Linux shell utilities using the “ | ” character
 - **grep, awk, wc, tail, more, less, sort, ...**

```
controller# show running-config | grep snmp
! snmp-server
snmp-server host 10.2.24.255
snmp-server enable traps
snmp-server community ro public
```

- Redirect output to a file using the ">" character

```
controller# show running-config > config1.txt
```

- Issue multiple commands separated by a semi-colon

```
controller# config ; local node ; hostname bcf-controller-1 ; end
bcf-controller-1#
```

BCF ACCESS AND CONFIGURATION

CLI – Managing the CLI session

- Configure CLI parameters
 - Paginate all output (ssh connections only)

```
controller> terminal length term
```

- Remove output pagination

```
controller> terminal length 0
```

- Clear and reset terminal screen

```
controller> terminal clear
```

- CLI session times-out after two hours of inactivity
 - Prompt indicates when re-authentication is required

```
reauth controller> reauth
```

BCF ACCESS AND CONFIGURATION

REST – Overview

- BCF managed via REST API calls
- CLI and GUI are REST clients
- CLI and GUI provide the ability to see all REST calls
- Helpful in writing automation scripts
- Automation of REST calls via Python, Ruby, Perl, . . .

BCF ACCESS AND CONFIGURATION

REST – CLI

- CLI REST calls – issue command “debug rest”

```
controller# debug rest
***** Enabled display rest mode *****
REST-SIMPLE: GET http://127.0.0.1:8080/api/v1/data/controller/core/controller/role
REST-SIMPLE: http://127.0.0.1:8080/api/v1/data/controller/core/controller/role done, 0:00:00.002529

controller# show version
REST-SIMPLE: GET http://127.0.0.1:8080/api/v1/data/controller/core/version/appliance
REST-SIMPLE: http://127.0.0.1:8080/api/v1/data/controller/core/version/appliance done, 0:00:00.002192
~~~~~ Appliance ~~~~~
Name : Big Cloud Fabric Appliance
Build date : 2015-10-21 12:37:23 PDT
Build user : bsn
Ci build number : 166
Ci job name : bcf-3.1.0
Release string : Big Cloud Fabric Appliance 3.1.0 (bcf-3.1.0 #166)
Version : 3.1.0
REST-SIMPLE: GET http://127.0.0.1:8080/api/v1/data/controller/core/controller/role
REST-SIMPLE: http://127.0.0.1:8080/api/v1/data/controller/core/controller/role done, 0:00:00.002020

controller# no debug rest
***** Disabled display rest mode *****
controller#
```

BCF ACCESS AND CONFIGURATION

REST – GUI

- GUI REST calls – click “dragonfly” in top left corner

The screenshot shows the Big Switch Network GUI interface. At the top, there is a navigation bar with tabs: Fabric, Logical, Endpoints, Visibility, Settings, Integration, Security, and Default admin. A red circle highlights the "Fabric" tab. Below the navigation bar is a button labeled "Show API requests and responses".

Underneath the button is a table titled "Filter table rows" with columns: Timestamp, Method, Path, Port Override, Host Override, Timeout (s), Status, and Response Size (Chars). The table lists several API requests from today:

Timestamp	Method	Path	Port Override	Host Override	Timeout (s)	Status	Response Size (Chars)
Today, 2:49:27pm PST	GET	/api/v1/data/controller/applications/bcf/info?select=errors&select=warnings&single=true	—	—	10	200 (OK)	7,635
Today, 2:49:26pm PST	GET	/api/v1/data/controller/core/service/entry	—	—	10	200 (OK)	8,354
Today, 2:49:26pm PST	GET	/api/v1/data/controller/cluster?single=true	—	—	10	200 (OK)	1,140
Today, 2:48:26pm PST	GET	/api/v1/data/controller/applications/bcf/info?select=errors&select=warnings&single=true	—	—	10	200 (OK)	7,635
Today, 2:47:27pm PST	GET	/api/v1/data/controller/core/switch?select=host-stats	—	—	10	200 (OK)	6,188

Below the table are two cards: "Attributes" and "Controller". The "Attributes" card displays system information such as Mode (HA), IP Addresses (Virtual - None -), Active Controller (10.2.24.131), Standby Controller (10.2.24.132), Version (3.1.0), Controller Uptime (1 week, 5 days), and Cluster Uptime (1 week, 5 days). The "Controller" card shows the active controller status (Active) and memory usage metrics for Memory, CPU, Swap Used, Root Disk (/), and Log Disk (/log).

On the right side, there is a pie chart titled "Memory" showing the distribution of memory usage between idle, floodlight, and other categories.

BCF ACCESS AND CONFIGURATION

REST – CURL

- Using curl
 - Login and get a session cookie

```
curl -g --insecure -H "Content-type: application/json" -d '{"user": "admin",  
"password": "<password>"}' -s https://10.2.24.141:8443/api/v1/auth/login
```

- Use cookie to execute config or show commands

- show switch

```
curl -s -H "Cookie:session_cookie=<cookie>" -g -X GET --insecure  
'https://10.2.24.141:8443/api/v1/data/controller/core/switch'
```

- show version

```
curl -s -H "Cookie:session_cookie=<cookie>" -g -X GET --insecure  
'https://10.2.24.141:8443/api/v1/data/controller/core/version/appliance'
```

Questions?

BCF ACCESS AND CONFIGURATION

Configuration Management

- Running config
 - Updates take effect immediately
 - Configuration saved immediately (no startup config)
 - List complete running config or subset of running config
 - List running config with default values

```
controller# show running-config
controller# show running-config controller|aaa|snmp|...
controller# show running-config details
```

- View configuration of current hierarchy

```
controller(config-switch)# show this
```

BCF ACCESS AND CONFIGURATION

Configuration Management

- Configuration save options:

- In the snapshot database
- To a local text file
- To a remote scp server

```
ctrl# copy running-config snapshot://config  
ctrl# copy running-config file://config.txt  
ctrl# copy running-config scp://user@host:config.txt
```

- Configuration load options:

- From the snapshot database
- From a local text file
- From a remote text file via scp
- From a remote text file via http

```
ctrl# copy snapshot://config running-config  
ctrl# copy file://config.txt running-config  
ctrl# copy scp://aj@host:config.txt running-config  
ctrl# copy http://www-host:config.txt running-config
```

BCF ACCESS AND CONFIGURATION

Configuration Management

- Snapshots
 - Saved in an internal database (json format)
 - Can be exported using scp
 - Can import a json formatted file
 - **Replaces the config when copied to running-config**
- Text Files
 - Saved in a directory on the controller or remote server
 - Edit with any text editor
 - **Merges the config when copied to running-config**

BCF ACCESS AND CONFIGURATION

File Management

- Copy files (including configurations):

```
controller# copy <source> <destination>
```

- Possible URLs (source or destination):

– Current config:	“running-config”
– External servers:	scp://<user@host>:path
– Config repository:	snapshot://<config-name>
– File system:	file://<file-name>
– HTTP servers:	http://<file-name> (source URL only)
– SW repository:	image:// (destination URL only)

BCF ACCESS AND CONFIGURATION

File Management

- View files on the controller:

```
controller# show file <file-name>
controller# show snapshot <config-name>
```

- Delete files on the controller:

```
controller# delete file <file-name>
controller# delete image <image-name>
controller# delete snapshot <config-name>
```

BCF ACCESS AND CONFIGURATION

Users and Groups

- Access enabled through user and group configurations
- Supports two groups: admin and read-only
- Admin user and group created during install
- Create local user

```
controller(config)# user user1
controller(config-local-user)# full-name "Test User 1"
controller(config-local-user)# password test
```

- Associate user with a group

```
controller(config)# group admin
controller(config-group)# associate user user1
```

BCF ACCESS AND CONFIGURATION

AAA – Overview

- Defaults to controller (local)
- Enable a TACACS+ or RADIUS server for AAA functions:

```
controller(config)# tacacs server host 192.168.17.201 key secret  
controller(config)# radius server host 192.168.17.201 key secret
```

- Supports up to 4 servers
- Control the number of active sessions & session expiration

```
controller(config)# aaa concurrent-limit 5  
controller(config)# aaa session-expiration <minutes>
```

BCF ACCESS AND CONFIGURATION

AAA – Authentication

- Authentication

- Local

```
controller(config)# aaa authentication login default local
```

- First Local then Remote (TACACS+ or Radius)

```
controller(config)# aaa authentication login default local group tacacs+
```

- First Remote then Local

```
controller(config)# aaa authentication login default group tacacs+ local
```

- Remote

```
controller(config)# aaa authentication login default group tacacs+
```

BCF ACCESS AND CONFIGURATION

AAA – Authorization

- Authorization

- Local

```
controller(config)# aaa authorization exec default local
```

- First Local then Remote (TACACS+ or Radius)

```
controller(config)# aaa authorization exec default local group tacacs+
```

- First Remote then Local

```
controller(config)# aaa authorization exec default group tacacs+ local
```

- Remote

```
controller(config)# aaa authorization exec default group tacacs+
```

BCF ACCESS AND CONFIGURATION

AAA – Accounting

- Accounting

- Local

```
controller(config)# aaa accounting exec default start-stop local
```

- Remote (TACACS+ or Radius)

```
controller(config)# aaa accounting exec default start-stop group tacacs+
```

- Local and Remote

```
controller(config)# aaa accounting exec default start-stop local group tacacs+
```

- Accounting Logs

- Local Logs

```
controller# show logging audit
```

- Remote logs on remote server

BCF ACCESS AND CONFIGURATION

Logging

- Controller logs
 - Local logging always enabled
 - Enable remote logging
- Switch logs
 - Switches forward all logs to controllers
 - Optionally send to remote server instead of controllers
- To view logs:

```
controller(config)# logging remote 192.168.17.1
```

```
controller(config)# logging switch-remote
```

```
controller# show logging
audit      Show audit file contents
controller Show log contents for the controller floodlight process
syslog     Show syslog file contents
```

BCF ACCESS AND CONFIGURATION

SNMP

- Configure SNMP parameters

```
controller(config)# snmp-server community ro bsn-secret  
controller(config)# snmp-server location "Santa Clara, CA"  
controller(config)# snmp-server contact "Lab Administrator"
```

- Allow SNMP access via controller access-list
 - Disabled by default

```
controller(config)# controller  
controller(config-controller)# access-control  
controller(config-controller-access)# access-list snmp  
controller(config-controller-access-list)# 1 permit from 10.0.0.0/8
```

BCF ACCESS AND CONFIGURATION

SNMP

- Enable SNMP traps (disabled by default)

```
controller(config)# snmp-server enable traps
```

- Configure SNMP trap receiver

```
controller(config)# snmp-server host 192.168.17.150
```

- View SNMP Config

```
controller# show running-config snmp  
controller# show running-config controller
```

BCF ACCESS AND CONFIGURATION

Other Config Commands

- Configure NTP
 - Switches synchronize from NTP servers directly (default)
 - Manually configure to synchronize from controller

```
controller(config)# ntp server 0.bigswitch.pool.ntp.org  
controller(config)# ntp time-zone US/Pacific  
controller(config)# ntp controller-source
```

- Configure login banner

```
controller(config)# banner "Welcome to Big Switch.\nAuthorized users only."
```

- Delete configurations using the “**no**” keyword

```
controller(config)# no snmp-server host 1.2.3.4  
controller(config)# no ntp server 0.bigswitch.pool.ntp.org
```

BCF ACCESS AND CONFIGURATION

Other CLI Commands

- Compare configurations

```
controller# compare snapshot://config1 file://config1.txt
```

- Generate tech-support bundle

```
controller# support
```

- List users logged into BCF

```
controller# show session
```

- Run command at every 2 second interval

```
controller# watch show switch all endpoint | grep "00:50:56:aa:f2:5d"
controller# watch show switch r111 interface ethernet37 counters
```

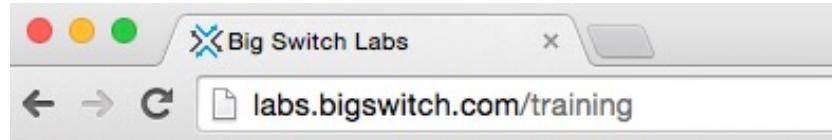
Questions?



Using BSN Labs

CONNECTING TO BSN LABS

- Direct your browser to <http://labs.bigswitch.com/training>



- Log in using the provided user account

LOGIN

ACCESS BSN LABS

EMAIL

PASSWORD

By registering and logging in you agree to our [terms of service](#) and [privacy policy](#).

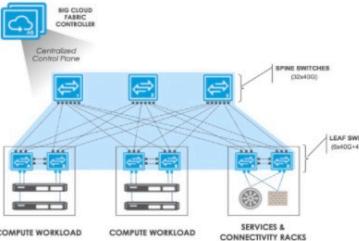
SIGN IN

LAUNCH MODULE

- Launch Module 1 under “Big Cloud Fabric” tab
 - It may take up to 15 minutes for module to launch

 Big Cloud Fabric

 Big Monitoring Fabric

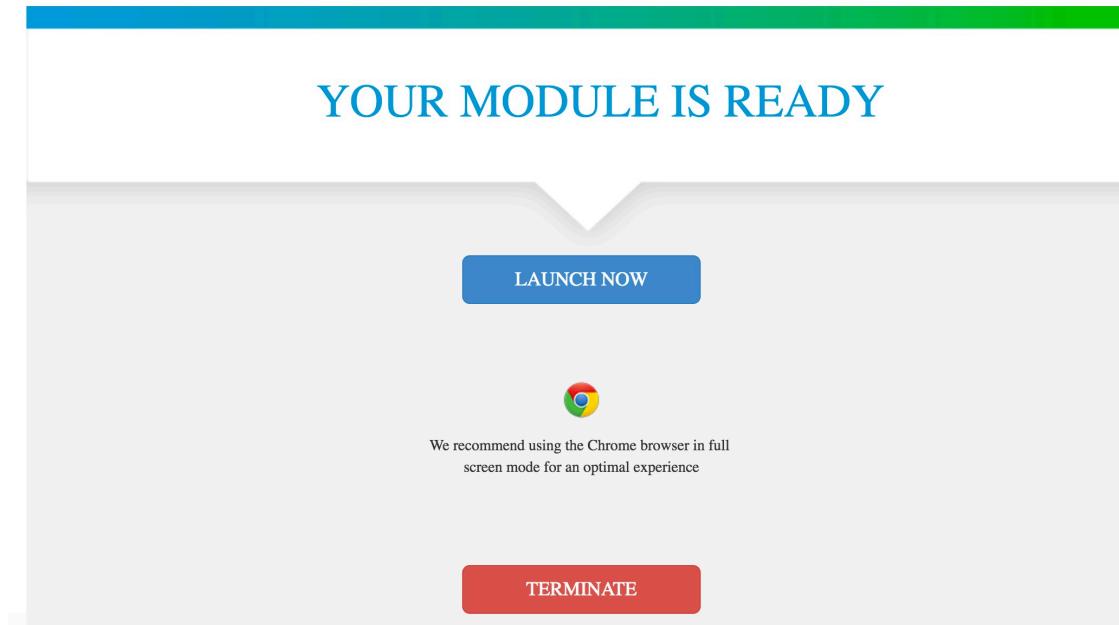


MODULE 1
Big Cloud Fabric 3.5 P-Edition Training
Introduction to the fundamental concepts and the architecture of Big Cloud Fabric.

LAUNCH

LAUNCH MODULE

- Click “LAUNCH NOW” to access the module



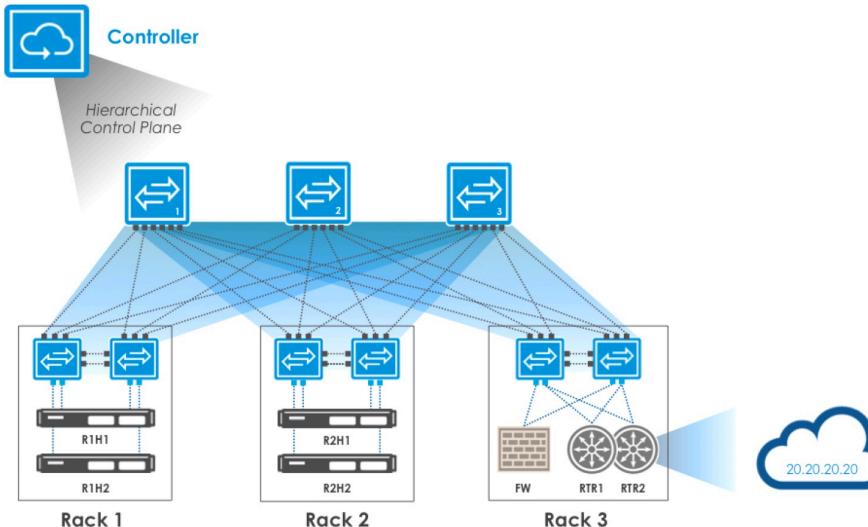
LAB TOPOLOGY

- Access to topology available from “DEVICES” menu



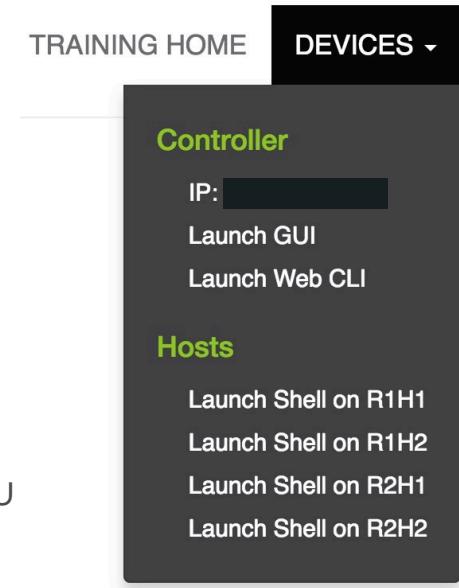
TRAINING HOME DEVICES ▾

Big Cloud Fabric Topology



ACCESSING THE CONTROLLER

- GUI
 - Launch GUI from the “DEVICES” menu
 - Or use IP address displayed in “DEVICES” menu
- CLI
 - Preferred access method is via a laptop client
 - ❖ Ex: PuTTY, SecureCRT, or other
 - ❖ Use IP address displayed in the “Devices” menu
 - Alternatively, use Web CLI from the “Devices” menu
- Login credentials – User: **admin** Password: **bsn123**





Lab – Hands-on BCF Environment

LAB OBJECTIVES

- Become familiar with CLI & GUI
- Edit configurations
- Save and restore configurations
- Understand CLI tools & capabilities
- Build support packages
- Perform tasks using GUI

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (1)

1. Start a ssh session to controller

1. Use secureCRT, PuTTY, or other client to access controller
2. Use the IP address shown in “DEVICES” menu (Figure 1)
3. Use Web CLI if laptop client is not available
4. Login credentials – User: **admin** Password: **bsn123**

2. Check version

1. show version

3. Check controller status

1. show controller

4. Check all users signed into BCF

1. show session

5. View running configuration

1. show running-config

Note: Use ? to get help on commands. Ex: show controller ?

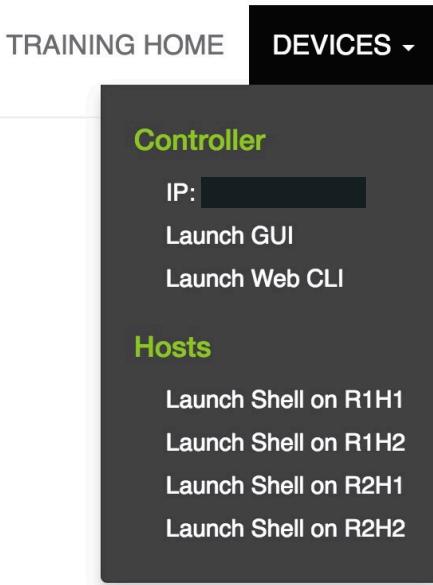


Figure 1

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (2)

1. Save a snapshot of running configuration (Lab-1-Config-1)

1. enable ; copy running-config snapshot://Lab-1-Config-1
Note: semicolon in configuration statements should be replaced with carriage returns.
2. show snapshot
3. View contents of snapshot: show snapshot Lab-1-Config-1

2. Add a new user John Smith

1. Enter config mode: config
2. Use admin user configuration as an example: show running user
3. Use "?" to get help on available commands in current mode
4. User id: john Full name: "John Smith" Password: bsn123
5. Use "show running user" to see all user configurations
6. Use "show running user john" to see only user configuration
7. Use "show this" to see user configuration from within config mode (for current user)
8. Associate john with group "admin"
9. Use admin group configuration as an example: show running group

Note: Use <tab> to complete partially typed commands. If command does not complete, it may be typed incorrectly.

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (3)

1. Update controller hostname to “bcf-<your name>”

1. local node ; ...
2. Use "?" to get online help and identify "Commands" available in current mode

2. Update controller DNS parameters

1. local node ; interface eth0 ; ipv4 ; ...
2. Add Google DNS servers: 8.8.8.8 and 8.8.4.4
3. Add DNS search domain "bigswitch.com"

3. Add NTP server

1. 0.us.pool.ntp.org

4. Save another snapshot of running config (Lab-1-Config-2)

5. Save a copy of running configuration in a text file (Lab-1-Config-3.txt)

1. copy running-config file://Lab-1-Config-3.txt
2. show file

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (4)

1. Add SNMP config (host, enable, community, contact, location)

1. snmp . . .
2. Community – public. Contact – your name. Location – BSN Labs
3. Enable traps to host 10.10.10.99
4. Check snmp configuration – show running snmp

2. Compare running config with snapshot Lab-1-Config-2

1. compare running-config ...
2. "<" represents differences in first file (running-config)
3. ">" represents differences in second file (Lab-1-Snapshot-2)
4. The differences should be snmp and comments

3. Compare snapshots and text files

1. Compare snapshots Lab-1-Config-1 and Lab-1-Config-2
2. Compare snapshot Lab-1-Config-2 and text file Lab-1-Config-3.txt

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (5)

1. Delete NTP servers
 1. Delete 0.bigswitch.pool.ntp.org and 1.bigswitch.pool.ntp.org
2. Save another snapshot of running config (Lab-1-Config-4)
3. Save snapshot Lab-1-Config-4 as text file (Lab-1-Config-5.txt)
4. Load snapshot configurations
 1. copy snapshot://Lab-1-Config-1 running-config
 2. All changes made so far should now be removed (snmp, ntp, user) except for local node
 3. Load configuration from snapshot Lab-1-Config-4
 4. All changes made in this lab should now be restored
 5. Local node configuration is controller vs cluster specific and is not replaced during copy
5. Load config from text file Lab-1-Config-3.txt
 1. Because this is a merge operation, result is much different than in previous task
 2. The two NTP servers deleted after saving Config-3 are restored
 3. The SNMP configuration added after saving Config-3 is not removed

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (6)

1. Search for the word “server” in running-config

1. show ... | grep server

2. Generate tech support package

1. support
2. List support files including URLs to download file

3. Configure CLI settings

1. Enable output pagination: terminal length term
2. Verify: show running-config or show logging controller
3. Restore normal output: terminal length 0
4. Verify output is no longer paginated: show logging controller

4. Paginate output using other utilities

1. show running-config | less (Enter “h” for help; “q” to exit)
2. show running-config | more

5. View “history” of all the previously entered commands

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (7)

1. Search for a previously typed command using ^r (control R)
 1. Search for “running”
 2. Use ^r repeatedly to find older commands containing this keyword
2. Use watch command to continuously run another command
 1. Ex: `watch show clock` (^c to exit)
 2. Use watch to monitor controller interface statistics
 3. `watch show local node interfaces eth0 stats`
3. Enable debug REST mode to see all REST API calls
 1. `debug rest`
 2. Issue commands: `show version`, `show controller`, `show switch`
 3. Exit this mode: `no debug rest`
4. Delete the following snapshots and files
 1. `Lab-1-Config-1`, `Lab-1-Config-2`, `Lab-1-Config-4`, `Lab-1-Config-3.txt`, `Lab-1-Config-5.txt`

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (8)

1. Perform all remaining tasks using GUI
2. Check switch status and properties
 1. List all switches (Menu: Fabric → Switches)
 2. Click on S1
 3. Hover mouse over each switch interface to get status
 4. Select “Properties”, “Action”, “Interface”, “Links” for more information and actions
3. Check interface status
 1. List all interfaces (Menu: Fabric → Interfaces)
 2. Select 100 at the bottom of page to see all interfaces on one page (Figure 1)
 3. Add “Admin Status” to list of visible columns (Figure 2)
 4. Filter all interfaces with “eth6” in their names using “Filter table rows” at the top of table
4. View BCF Topology
 1. Click on menu entry “Fabric”
 2. Hover mouse over any switch to get switch status

Interfaces		
Filter table rows		
Show/Hide Columns		
Shutdown Selected Interfaces		
Startup Selected Interfaces		
<input type="checkbox"/> dt-spine2	70:72:cf:ae:a0:5e	ethernet1
<input type="checkbox"/> dt-spine2	70:72:cf:ae:a0:5e	ethernet11
<input type="checkbox"/> dt-spine2	70:72:cf:ae:a0:5e	ethernet12
<input type="checkbox"/> dt-spine2	70:72:cf:ae:a0:5e	ethernet13
<input type="checkbox"/> dt-spine2	70:72:cf:ae:a0:5e	ethernet14

Figure 2

–	–	–
Show:	<u>10</u>	<u>25</u>
(1 - 65 / 65)		

Figure 1

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (9)

1. View BCF Topology (Cont'd)

1. Click on “show/hide links” icon (Figure 1)
2. Click on a spine switch to see all fabric links from this switch
3. Click on a leaf switch to see all fabric links from this switch
4. Hover over a fabric link to get information on link
5. List all switches (Fabric → Switches)
6. Shutdown R1L1 using menu (Figure 2)
7. Go back to BCF Topology (click on menu entry “Fabric”)
8. R1L1 leaf switch should show suspended status since it was shutdown
9. Hover over the green/yellow lights for details of status
10. Click on “show/hide links” icon
11. Select spine and leaf switches to see change in fabric topology
12. List all interfaces in BCF (Fabric → Interfaces)
13. Filter all interfaces with “eth2” in their names
14. Shutdown R2L1-eth2 using menu



Figure 1

Name	MAC	Connected
R1L1	00:00:00:02:00:01	✓
R1L2	00:00:00:02:00:02	✓

Figure 2

LAB – HANDS-ON BCF ENVIRONMENT – PAGE (10)

1. View BCF Topology (Cont'd)

1. Go back to BCF Topology (click on menu entry "Fabric")
2. Click on "show/hide links" icon
3. Click on R2L1 leaf switch to see the change in fabric links
4. Click on S1 spine switch to see the impact from this switch
5. S1 is only connected to 4 leaf switches while S2 and S3 are connected to 5

2. Generate Support Bundle (Visibility → Support Bundles)

3. Track GUI REST calls

1. Go to dashboard (click on Big Switch logo in top left corner)
2. Click on dragonfly just below the Big Switch logo (Figure 1)
3. Expand window that just opened if needed
4. Clear window contents
5. Refresh any part of the screen to see REST calls made by GUI
6. Close REST call window



Figure 1

Questions?

Please stop here and let your instructor know that you have finished.

Module 3 – BCF Deployment

BCF DEPLOYMENT

Module Outline

- Components
- Management & Control Networks
- Bring up Controllers
- Bring up Switches
- Configure Fabric
- **Lab – Working with Controller and Switches**

BCF DEPLOYMENT

Components

Controller

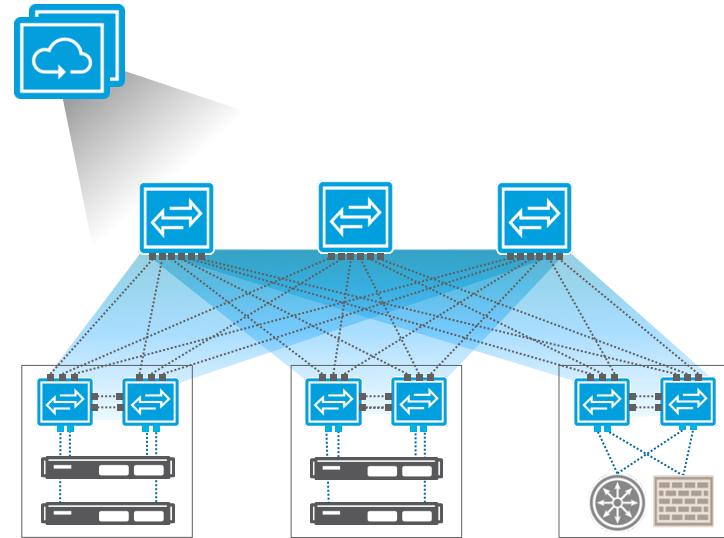
- X86 device or VM that manages the Big Cloud Fabric (fabric components, discovery, configuration)
- Pair of controllers form a cluster

Switch (Spine)

- Backbone of the fabric
- Only fabric links (no links to external devices)
- No connection to other spines
- Must connect to each leaf

Switch (Leaf)

- Ingress and egress point for BCF traffic
- Connects to external devices (servers, routers, firewall, ...)
- Must connect to each spine



BCF DEPLOYMENT

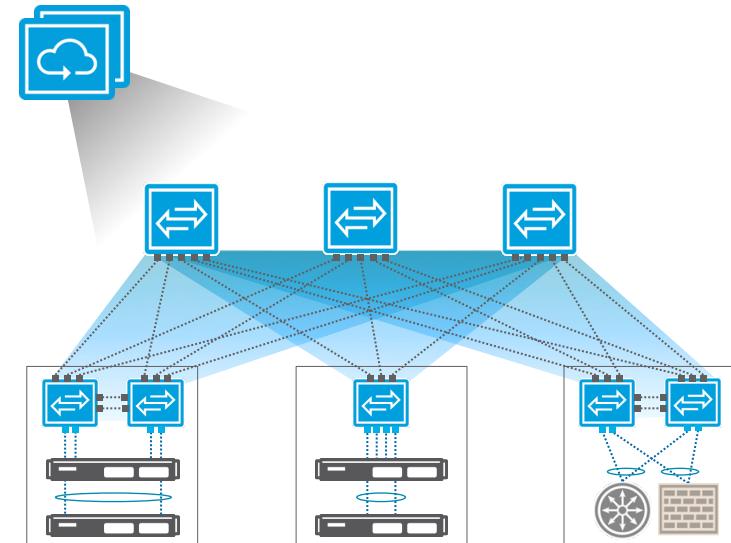
Components

Leaf Group

- Consists of one or two leaf switches
- May not have more than two leaf switches
- Every leaf switch belongs to a leaf group
- Peer link connects the two leaf switches
- Links across leaf groups not permitted

Interface Group

- Same as LAG (Link Aggregation Group)
- Members (interfaces) must belong to same leaf group
- May not span across leaf groups



Note: During this training, Interface Group / LAG is implied in all Leaf Group topologies unless explicitly noted as on this slide.

BCF DEPLOYMENT

Management & Control Networks

Management Network

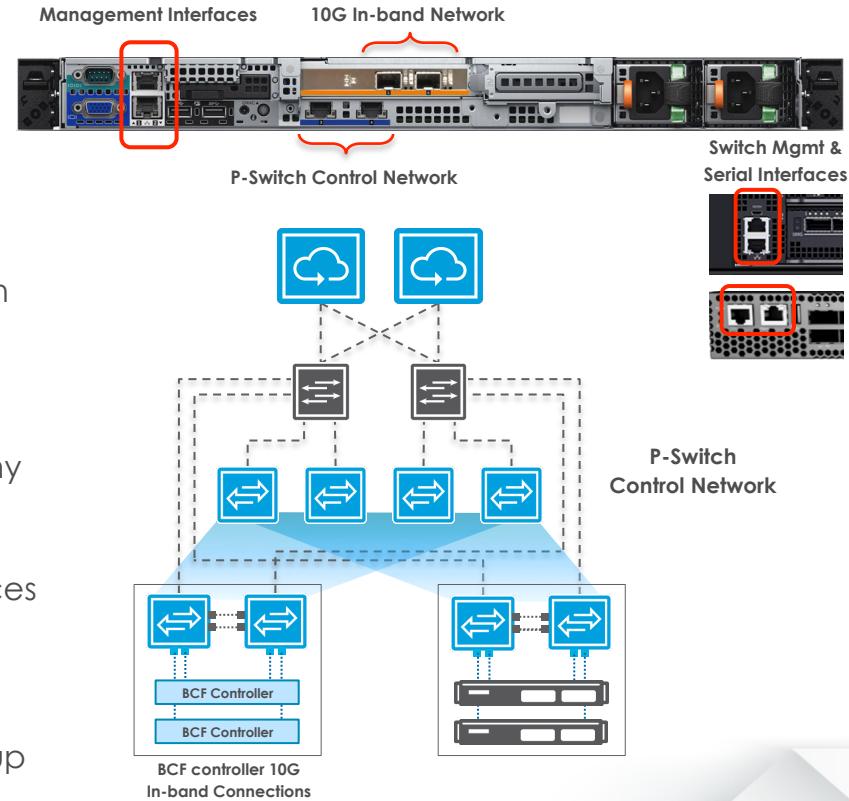
- Access BCF controllers via SSH / GUI / REST API
- Only controllers connect to Management Network

P-Switch Control Network

- 1G Copper based OOB network for controller/switch communications
- Uses two dedicated interfaces on controller
- Uses management interface on switch
- Option to secure communication using cryptography

In-band Control Network

- Controller connects to customer facing 10G interfaces on leaf switches
- Both controllers may connect to same leaf group or different leaf groups
- Both 10G interfaces must connect to same leaf group



BCF DEPLOYMENT

Controller First Boot Information

First Boot

- First time controller is booted after Switch Light OS install
- State after controller is reset to factory-default
- Allows login as “admin” without password
- Requests information for initial configuration

Controller First Boot Table

First Boot Parameter	Value	Description
Controller 1 hostname		
Controller 2 hostname		
Controller 1 IP		IPv4, IPv6, or both
Controller 2 IP		IPv4, IPv6, or both
Gateway		
Cluster name		Controller cluster
Cluster description		Optional
Admin password		For cluster access
Recovery password		For host access/recovery
NTP servers		Optional (default servers populated)
DNS Servers		Optional
DNS Search Domain		Optional

BCF DEPLOYMENT

Controller Preparations & Console Access

Management & Control Networks

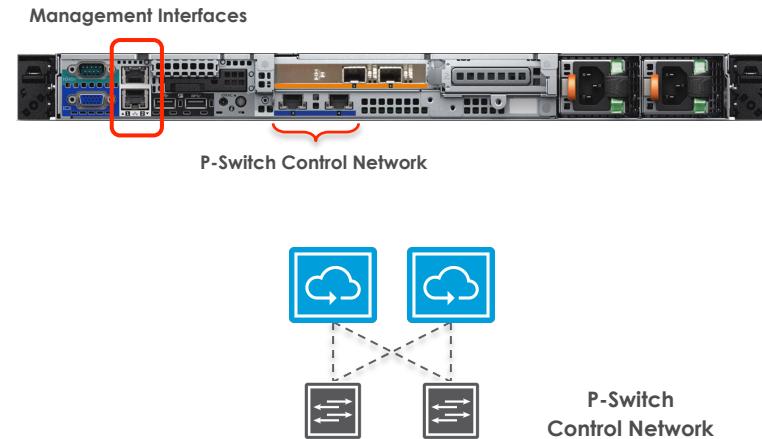
- Connect controllers to management network
- Connect controllers to P-Switch control network

Appliance Prep

- Big Cloud Fabric OS preloaded to latest version
- Connect keyboard and monitor or serial console
- Power on the appliance

VM Prep

- Switch Light OS based on the VM package downloaded
- Install VM in the appropriate hypervisor
- .ova for ESXi hosts and .qcow2 for Linux hosts
- Power up the VM
- Connect console to the VM



BCF DEPLOYMENT

Controller First Boot

Login as “admin” to configure

- No password required
- Accept EULA and answer prompts with information from “Controller First Boot Table”

Management Address Options

- Three options: IPv4, IPv6, or IPv4/IPv6

Cluster Options

- Controller 1 will start a new cluster
- Controller 2 will join the existing cluster

Login as “admin”

- Use admin password just configured
- Check controller version and status

```
controller# show version
controller# show controller
```

Please choose an option:

- | | |
|---------------------------------|-----------------------|
| [1] Apply settings | (*****) |
| [2] Reset and start over | (BCF-Controller-1) |
| [3] Update Recovery Password | (IPv4 only) |
| [4] Update Hostname | (10.9.19.78/23) |
| [5] Update IP Option | (10.9.18.1) |
| [6] Update IPv4 Address | (10.3.0.4) |
| [7] Update IPv4 Gateway | (<none>) |
| [8] Update DNS Server 1 | (bigswitch.com) |
| [9] Update DNS Server 2 | (Start a new cluster) |
| [10] Update DNS Search Domain | (BCF-Lab-Cluster) |
| [11] Update Cluster Option | (<none>) |
| [12] Update Cluster Name | (*****) |
| [13] Update Cluster Description | (Default NTP servers) |
| [14] Update Admin Password | |
| [15] Update NTP Option | |

[1] >

BCF DEPLOYMENT

Controller First Boot (2nd Controller)

First Boot – 2nd Controller

- Repeat same procedure with following differences
- During the cluster option, join existing cluster

```
[1] Start a new cluster  
[2] Join an existing cluster
```

- Provide IP address of the cluster (1st controller) and admin password

Controller Versions

- Both controllers must have the same version of Big Cloud Fabric OS

Repeat First Boot

- If needed, first boot process can be repeated

```
controller# boot factory-default
```

BCF DEPLOYMENT

Controller Node Configuration

Node Configuration (local node)

- Unique on each controller
- Configure individually if needed after first-boot
 - Hostname
 - IP configuration
 - DNS domain and server
- Snapshot does not apply this configuration
 - Allows configuration to be portable

Controller 1

```
local node
hostname bcf-controller-1
interface management
  ipv4
    10.2.24.97/24 gateway 10.2.24.1
    dns server 8.8.8.8
    dns search bigswitch.com
    method manual
```

Controller 2

```
local node
hostname bcf-controller-2
interface management
  ipv4
    10.2.24.98/24 gateway 10.2.24.1
    dns server 10.2.25.57
    dns search bigswitch.com
    method manual
```

BCF DEPLOYMENT

Controller Configuration

Cluster Configuration (controller)

- Common on both controllers
 - Name, Description
 - Virtual IP, ACLs

Access Control Lists (ACLs)

- Manages IPv4 and IPv6 access
- Provides CIDR level access control
- Also applies to switches
- Default access allowed for all three modes of access: API, GUI, & SSH
- SNMP disabled by default
- SNMP access must be enabled explicitly if SNMP configured

```
controller
  description 'Cluster for test lab'
  name test-lab
  virtual-ip 10.2.24.140
  access-control

  access-list api
    1 permit from ::/0
    2 permit from 0.0.0.0/0

  access-list gui
    1 permit from ::/0
    2 permit from 0.0.0.0/0

  access-list snmp
    1 permit from 10.2.24.102/32

  access-list ssh
    1 permit from ::/0
    2 permit from 0.0.0.0/0
```

BCF DEPLOYMENT

Controller Configuration

System Configuration

- Other system level parameters that can be configured
 - AAA
 - Banner
 - Local users
 - Local groups
 - Logging
 - NTP
 - SNMP
 - Radius
 - TACACS+

```
ntp server 0.bigswitch.pool.ntp.org

aaa accounting exec default start-stop local

local node
    hostname BCF-Pod2-ctlr-1
    interface management
        ipv4
            dns server 10.1.5.200
            ip 10.2.24.18/24 gateway 10.2.24.1
            method manual
        ipv6
            method auto

user admin
    full-name 'Default admin'
group admin
    associate user admin
```

BCF DEPLOYMENT

Controller Operational Commands

Status Checks

- Overall status & uptime
- CPU & memory utilization
- Interface MAC, IP, & state
- Interface utilization
- Firewall rules
- Filesystem utilization
- Running processes
- Active & listening network ports

```
controller# show controller details
controller# show controller localhost
controller# show local node interfaces
controller# show local node interfaces all stats
controller# show controller access-control
controller# show system storage
controller# show system process
controller# show controller service-addresses
```

Controller Operations

- Reload (soft boot ~30 secs)
- Reboot (hard boot ~3 mins)
- Halt / Shutdown
- Failover active to standby
- Remove from cluster

```
controller# system reload controller
controller# system reboot controller
controller# system shutdown controller
controller# system failover
controller# system remove-node
```

Questions?

BCF DEPLOYMENT

Switch Bring Up

Connect Switch to P-Switch Control Network

- Connect using switch management interface

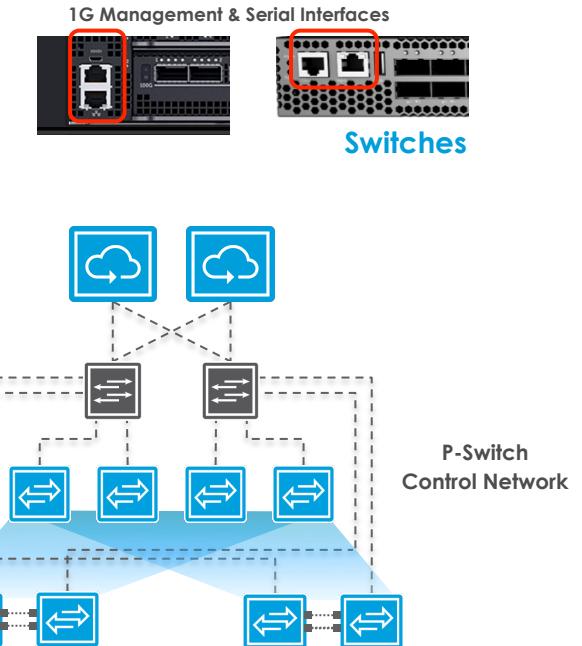
Enter Switch Configuration on Controller

- Configure MAC
- Configure Role

Power up the switches

- That's it!
- ZTF (Zero Touch Fabric) automatically builds the fabric
 - Switches automatically discover controllers
 - Switches automatically download latest software
 - Switches automatically download latest configuration

Configure fabric



BCF DEPLOYMENT

Configure Switch and Switch Roles

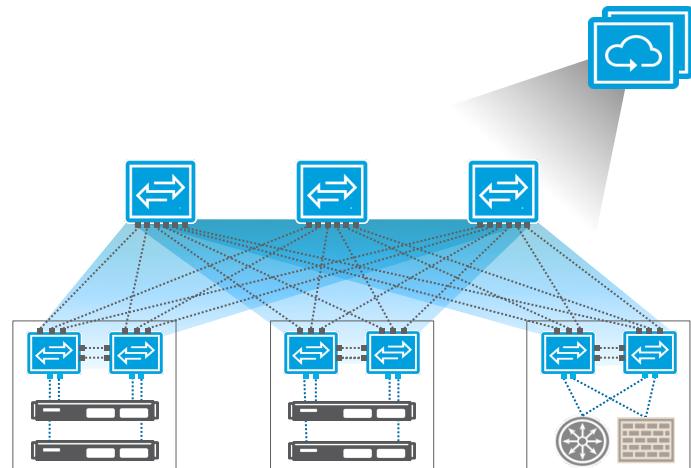
Switch Configuration

- Minimum configuration required to join fabric
 - Name
 - MAC Address
 - Role (spine or leaf)

```
switch S1
  mac 00:00:00:01:00:01
  fabric-role spine

switch R1L1
  mac 00:00:00:02:00:01
  fabric-role leaf

Switch R1L2
  mac 00:00:00:02:00:02
  fabric-role leaf
```



BCF DEPLOYMENT

Configuring Leaf Groups

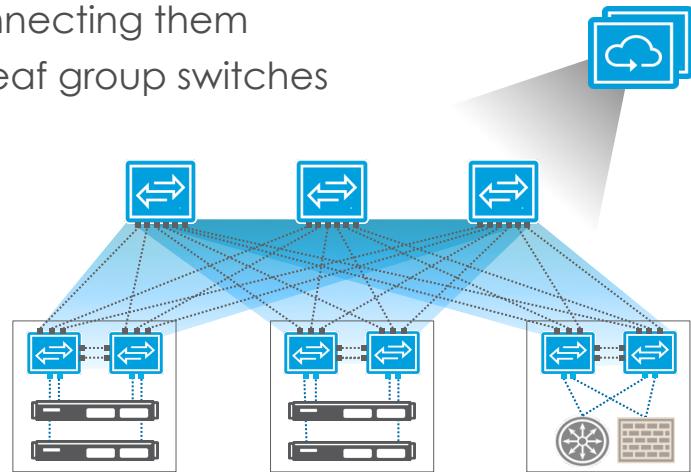
Leaf Groups

- Every leaf switch is a member of a leaf group
- May have one or two members
- Members of the leaf group must have a link connecting them
- Leaf group switches can not connect to other leaf group switches

```
switch R1L1
  mac 00:00:00:02:00:01
  fabric-role leaf
  leaf-group R1

switch R1L2
  mac 00:00:00:02:00:02
  fabric-role leaf
  leaf-group R1

switch R2L1
  mac 00:00:00:02:00:03
  fabric-role leaf
```



BCF DEPLOYMENT

Switch Status

Switch Status

- Every leaf switch is a member of a leaf group
- Leaf switches not configured in a leaf group are auto assigned a leaf group
- Switches automatically assign link local IPv6 address
- Controller and switch use IPv6 to establish connectivity

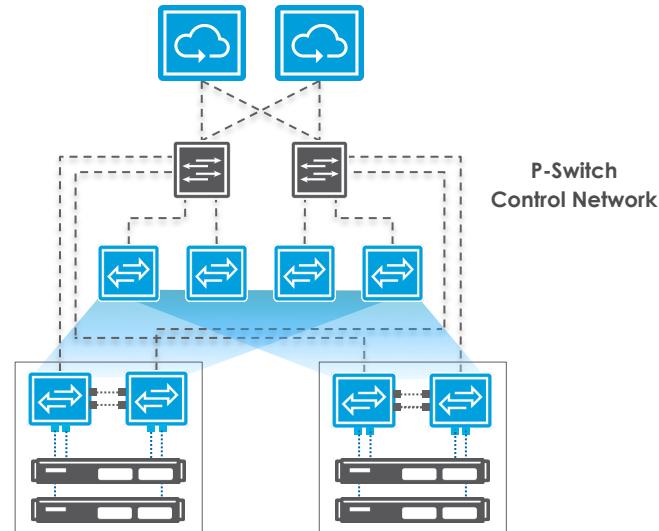
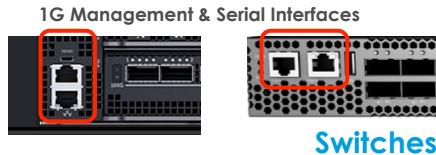
```
controller-1(config-switch)# show switch
# Switch Name Switch MAC Address IP Address          Connection State Fabric Role Leaf Group
- |-----|-----|-----|-----|-----|-----|-----|
1 R1L1    00:00:00:02:00:01  fe80:0:0:0:8ce8:16ff:fe6a:6133%3 connected      leaf      R1
2 R1L2    00:00:00:02:00:02  fe80:0:0:0:74fa:14ff:fe34:a987%3 connected      leaf      R1
3 R2L1    00:00:00:02:00:03  fe80:0:0:0:60ec:33ff:fe1b:3791%3 connected      leaf      00:00:00:00:00:02:00:03
4 R2L2    00:00:00:02:00:04  fe80:0:0:0:309b:7ff:fe4d:931b%3 connected      leaf      R2
5 R3L1    00:00:00:02:00:05  fe80:0:0:0:c482:dbff:fee3:a7c9%3 connected      leaf      R3
6 R3L2    00:00:00:02:00:06  fe80:0:0:0:a40f:b9ff:fea9:e9bc%3 connected      leaf      R3
7 S1     00:00:00:01:00:01  fe80:0:0:0:a8d9:e7ff:fe2b:821c%3 connected      spine
8 S2     00:00:00:01:00:02  fe80:0:0:0:d892:c1ff:fead:c0bb%3 connected      spine
9 S3     00:00:00:01:00:03  fe80:0:0:0:4c1c:12ff:fefe:9232%3 connected      spine
```

BCF DEPLOYMENT

Switch Bring Up – Troubleshooting

Issues / Troubleshooting

- Need serial console access
- Check for typos in MAC address
- Check P-Switch Control Network
 - Controller and switches need to be on same VLAN
- Check for different OS on the system
 - Delete existing OS
- Check for outdated ONIE
 - Update ONIE from ONIE Rescue Mode
- Check log messages for MAC address

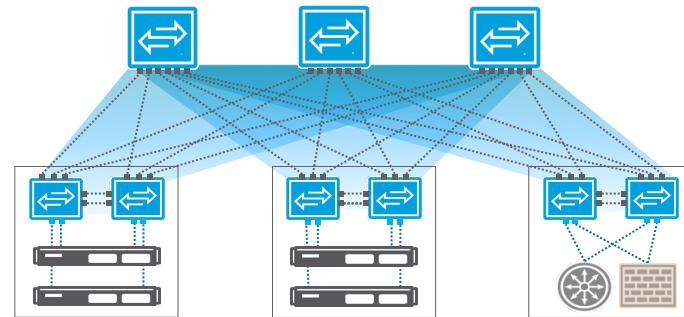


BCF DEPLOYMENT

Discover BCF Fabric

Discover fabric

- No configuration required
- Uses LLDP to discover inter-switch connectivity
- Connections required for topology enabled
- Extraneous connections disabled
 - Spine to spine
 - leaf to leaf (across leaf groups)



Check fabric interfaces

- Spine to leaf
- Leaf to leaf (within leaf group)

```
controller# show link
```

#	Switch Name	IF Name	Switch Name	IF Name	Link Type
1	R1L1	ethernet48	R1L2	ethernet48	peer
2	R1L2	ethernet48	R1L1	ethernet48	peer
3	Spine1	ethernet1	R1L1	ethernet50	leaf-spine
4	Spine2	ethernet1	R1L1	ethernet51	leaf-spine
...					

BCF DEPLOYMENT

Switch IP Addresses

IPv6 Addresses

- Link local addresses used for all controller to switch communications

IPv4 addresses

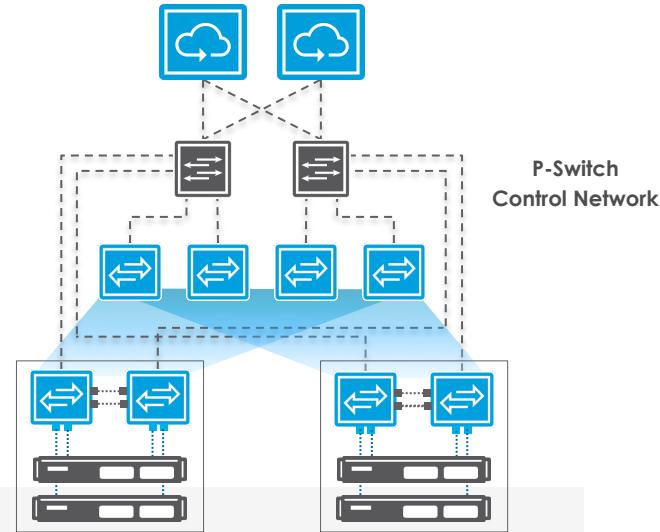
- Required for SNMP, Syslog, and possibly NTP
- Assigned by controller using IPAM

IPAM (IP Address Management)

- Can allocate non-contiguous address blocks within the same subnet

```
controller# show fabric ipam switch
```

```
fabric
  ipam switch
    allocate
      gateway 10.2.24.1
      ip-range 10.2.24.22 10.2.24.25 subnet-mask-length 24
      ip-range 10.2.24.32 10.2.24.33 subnet-mask-length 24
      ip-range 10.2.24.77 10.2.24.85 subnet-mask-length 24
```



BCF DEPLOYMENT

Shut Down a Switch

Shutdown a switch

- Gracefully remove from fabric
- Similar to admin shutdown for interfaces
- Ex: RMA or switch upgrade

```
switch R2L1
  mac 00:00:00:02:00:03
  role leaf
  leaf-group R2
  shutdown

switch R2L2
  mac 00:00:00:02:00:04
  role leaf
  leaf-group R2
```

controller-1(config-switch)# show switch				Connection State	Fabric	Role	Leaf Group
#	Switch Name	Switch MAC Address	IP Address				
1	R1L1	00:00:00:02:00:01	fe80:0:0:0:8ce8:16ff:fe6a:6133%3	connected		leaf	R1
2	R1L2	00:00:00:02:00:02	fe80:0:0:0:74fa:14ff:fe34:a987%3	connected		leaf	R1
3	R2L1	00:00:00:02:00:03	fe80:0:0:0:60ec:33ff:fe1b:3791%3	suspended (Admin shutdown)	leaf		R2
4	R2L2	00:00:00:02:00:04	fe80:0:0:0:309b:7ff:fe4d:931b%3	connected	leaf		R2

BCF DEPLOYMENT

Interfaces

Switch Interfaces

- Default – All interfaces enabled and ready to receive traffic
- Type – edge, leaf, spine
 - Any interface not leaf or spine is edge
- Admin shutdown of the interface
 - Same as a traditional switch

```
switch R1L1
  interface ethernet10
    shutdown
```

controller# show switch R1L1 interface all											
#	Switch	IF Name	IF Type	Phy. State	Op. State	IF Down Reason	LACP	State	BPDU-Guard	State	Curr Features
1	R1L1	ethernet1	edge	up	up	None	active	inactive	inactive	10gb-fd, copper	
3	R1L1	ethernet3	unknown	down	down	Link Down	inactive	inactive	inactive	1gb-fd, copper	
10	R1L1	ethernet10	unknown	down	down	Interface Admin Shutdown	inactive	inactive	inactive	1gb-fd, copper	
48	R1L1	ethernet48	leaf	up	up	None	inactive	inactive	inactive	10gb-fd, copper	
49	R1L1	ethernet49	spine	up	up	None	inactive	inactive	inactive	copper, 40gb-fd	

BCF DEPLOYMENT

Configure Breakout Interfaces

Breakout or Splitter Cables

- QSFP128 splitter cable
 - Splits 100G to four 25G (breakout fiber or DAC)
- QSFP+ splitter cable
 - Splits 100G/40G to four 10G (breakout fiber, DAC, or AOC)
- Not all interfaces support breakout (switch dependent)
 - Check HCL or use CLI command

```
controller# show switch Spine1 interface properties
```

Supported Cables

- Auto detected
- DAC cables default to 4x10G breakout
- Use “breakout” command for cables not auto-detected
- 4x25G breakout requires explicit configuration

```
switch R1L1
  interface ethernet49
    breakout
switch Spine1
  interface ethernet10
    breakout mode 4x25G
```

BCF DEPLOYMENT

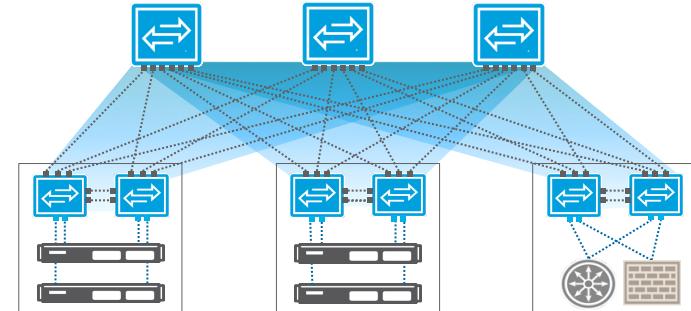
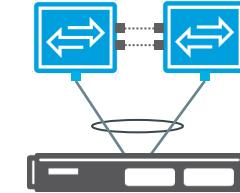
Interface Groups – Overview

Interface Groups

- Aggregate multiple links (LAG)
- All links must be part of same leaf group
- Links operate as active / active
- Option to have links operate as active / standby
- Up to 24 members (switch/hardware dependent)

Modes

- Static (default mode)
- LACP (fast mode)
- CDP, LLDP (to auto discovers hosts)
- Lacp-fallback-individual (for PXE boot servers)
- Inter-pod (for L2 Inter-pod connectivity)
- Span-fabric (for Fabric SPAN)



BCF DEPLOYMENT

Interface Groups – Configuration and Status

Interface Group Configuration

- G1 – Static based
- G2 – LACP based
- G3 – LLDP based
 - Any switch interface on which LLDP is received where system name is “compute-node1” and interface that sent the LLDP is “em0” or “em1” will join this interface group

```
interface-group G1
    member switch R1L1 interface ethernet12
    member switch R1L2 interface ethernet23

interface-group G2
    mode lacp
    member switch R1L1 interface ethernet35
    member switch R1L2 interface ethernet35

interface-group G3
    mode lldp
    member host compute-node1 interface em0
    member host compute-node1 interface em1
```

Interface Group Status

```
controller# show interface-group G2 members
# Name Mode Switch Name Interface Name Leaf Group IF State IF Down Reason
-|----|----|-----|-----|-----|-----|-----|-----|
1 G2   lacp  R1L1      ethernet35   R1      up     None
2 G3   lacp  R1L2      ethernet35   R1      up     None
```

BCF DEPLOYMENT

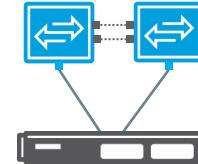
LACP-Fallback-Individual Interface Groups

LACP-Fallback-Individual Mode

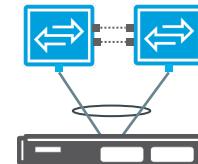
- PXE boot servers using same set of interfaces as server uplinks
- No LACP packets – interfaces operate in individual mode
- LACP packets – becomes a LAG (Interface Group)

```
interface-group G1
  mode lacp-fallback-individual
  member switch R1L1 interface ethernet7
  member switch R1L2 interface ethernet7
```

- Interface group status
 - Interface in individual mode until LACP packets are seen



No LACP packets – Interfaces individual mode



LACP packets – Interfaces form LAG

```
controller1# show interface-group G1 members
# Name Mode          Switch Interface Phy. State Op. State      Leaf Group IF Down Reason
-|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
1 G1  lacp-fallback... R1L1  ethernet7 up        down       rack0    inactive interface
2 G1  lacp-fallback... R1L2  ethernet7 up        up(designate)  rack0    None
```

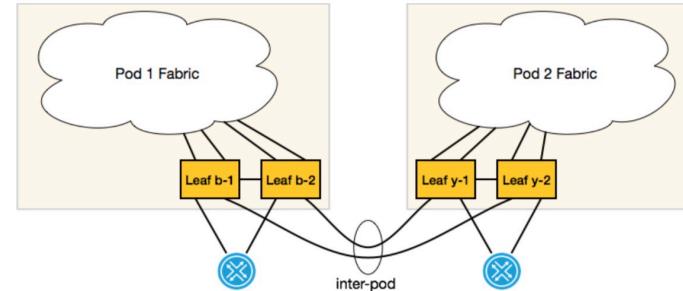
BCF DEPLOYMENT

Primary / Backup Interface Groups

Primary / Backup Interface Group

- Interface group operates as active / standby
- Backup members become active when all primary members are down
- Add backup members to create a primary / backup interface group
- Supports mode static, LACP, and inter-pod
- Example – Connect two PODs
 - Primary links connect the two PODs directly using dark fiber
 - Backup links connect the two PODs using L3 network
 - Primary / Backup links may operate in different modes

```
interface-group pod1-pod2
  mode inter-pod
  member switch leafb-1 interface ethernet1
  member switch leafb-2 interface ethernet1
  backup-mode lacp
  backup-member switch leafb-1 interface ethernet2
  backup-member switch leafb-2 interface ethernet2
  preempt
```



BCF DEPLOYMENT

Switch Operational Commands

Switch related operational commands

- Check status & connection time
- Check environment
 - Fans, power supplies, temperatures, system info
- Check switch configuration
- Check interface optics
- Check interface status
- Check interface stats / utilization

- Connect to switch (inband connection)
- Enable beacon to identify switch
- Reboot switch

```
controller# show switch all details
controller# show switch <name> environment
controller# show switch <name> running-config
controller# show switch <name> inventory
controller# show switch <name> interface
controller# show switch <name> interface stats
```

```
controller# connect switch <name>
controller# system beacon switch <name>
controller# system reboot switch <name>
```

BCF DEPLOYMENT

Fabric Status

Controller Status

- Node & Cluster information, redundancy status
- Active/Standby state, uptime, failover reason

Switch Status

- Switch role, member of leaf group
- Connection state and connection time

Fabric Status

- Overall status and inventory
- Detailed errors and warning

Fabric Links

- Spine to Leaf links
- Peer links (leaf to leaf)

```
controller# show controller  
  
controller# show switch  
  
controller# show fabric  
  
controller# show fabric error  
  
controller# show fabric warning  
  
controller# show link
```

Questions?



Lab – Working with Controllers and Switches

LAB OBJECTIVES

- Configure controller parameters
- Configure switches and switch roles
- Discover fabric topology
- Configure interface-groups
- Determine status of fabric, controller, and switches
- Perform shutdown and bring-up operations

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(1)

1. Load configuration for this lab

1. copy snapshot://BCF-Config2 running-config

2. Check controller status & configuration

1. show controller
2. Print only the controller configuration from config and non-config mode

3. Check switch status & configuration

1. show switch
2. Switches are in suspended state since there is no configuration in controller

4. Check fabric status

1. show fabric
2. Get details of fabric errors: show fabric ...

5. Use controller GUI to get controller, switches, and fabric status

1. Fabric -> Switches, Visibility -> Fabric Summary

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(2)

1. Update controller cluster configuration

1. config ; controller ; ...
2. Change the cluster name to "bcf-lab-cluster"
3. Add description for the controller cluster
4. Add VIP address for the cluster: 10.10.12.25
5. Check configuration: show this
6. Verify updates: show controller

2. Add spine "S1" to fabric

1. config ; switch S1 ; ...
2. Add MAC address 00:00:00:01:00:01
3. Assign fabric-role spine
4. Check configuration: show this
5. Check switch status: show switch

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(3)

1. Add leaf “R1L1” to fabric

1. config ; switch R1L1 ; ...
2. Add MAC address 00:00:00:02:00:01
3. Assign fabric-role leaf
4. Add to leaf group R1
5. Check configuration using “show this” and “show running switch”
6. Check switch status

2. Add remaining switches to fabric

1.	Name: S2	MAC: 00:00:00:01:00:02	Role: Spine	
2.	Name: S3	MAC: 00:00:00:01:00:03	Role: Spine	
3.	Name: R1L2	MAC: 00:00:00:02:00:02	Role: Leaf	Leaf Group: R1
4.	Name: R2L1	MAC: 00:00:00:02:00:03	Role: Leaf	Leaf Group: R2
5.	Name: R2L2	MAC: 00:00:00:02:00:04	Role: Leaf	Leaf Group: R2
6.	Name: R3L1	MAC: 00:00:00:02:00:05	Role: Leaf	Leaf Group: R3
7.	Name: R3L2	MAC: 00:00:00:02:00:06	Role: Leaf	Leaf Group: R3

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(4)

1. Check fabric status

1. show fabric
2. Get details of any fabric errors and warnings (Hint: show fabric ...)

2. Assign IPv4 addresses to each switch

1. Check fabric errors before configuring IPv4 addresses for switches
2. Check IPv4 addresses on switches: show fabric ipam switch
3. config ; fabric ; ipam switch ; ...
4. Configure IP address range 10.10.99.10 to 10.10.99.15 within /24 subnet
5. Configure gateway 10.10.99.1
6. Enable IPAM: allocate
7. Check IPv4 addresses: show fabric ipam switch
8. Check fabric errors again. There should be fewer errors.
9. Add more IP addresses from the same subnet: 10.10.99.50 to 10.10.99.55
10. Check IPv4 addresses again
11. Check fabric errors. There should be no errors.

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(5)

1. Save a snapshot of running config (Lab-2-Config-1)

2. Discover fabric topology

1. show switch
2. show link
3. Identify all the fabric links that have been discovered
4. First identify all switches (spines, leaf switches, leaf-groups)
5. Then map out links from each spine (to leaf switches)
6. show link | grep ...
7. Then map out links from each leaf switch (to spines and peer)

3. Pre-provision a new spine switch S4 (switch does not exist in the topology)

1. Use “show switch” to see status change as you add each line of configuration
2. MAC address 00:00:00:01:00:04
3. Fabric-role spine
4. Check fabric status and errors
5. Remove fabric role and check errors again

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(6)

1. Pre-provision a new leaf switch R4L1

1. Use "show switch" to see status change as you add each line of configuration
2. MAC address 00:00:00:02:00:07
3. Fabric-role leaf
4. Leaf group R4
5. Check fabric status and errors

2. Pre-provision a second leaf switch R4L2 for leaf group R4

1. MAC address 00:00:00:02:00:08
2. Check fabric errors?
3. There should be a new error about peer link

3. Move switch R3L1 to a new leaf-group R5

1. Use "show link | grep R3L1" to note all fabric links on R3L1 before making the change
2. Change the leaf-group name under switch R3L1 to create the new leaf-group
3. The fabric now has 5 leaf-groups
4. **What fabric links have changed and what is the new error?**

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(7)

1. Swap R1L2 and R2L2 from their current leaf groups
 1. Move R1L2 to leaf group R2 (Delete the leaf group configuration for both switches first)
 2. Move R2L2 to leaf group R1
 3. Check fabric status and errors
 4. Are there any errors for leaf group R1 and R2?
 5. Check links. (show link) Have some links gone away? Are there any peer links?
2. Remove R1L1 from its leaf groups?
 1. Check switch status
 2. **How many leaf groups are there now?**
3. Delete leaf switches R4L1 and R4L2
4. Restore configuration from snapshot Lab-2-Config-1
5. Shutdown spine S1
 1. Check switch status, fabric status, and links

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(8)

1. Shutdown leaf switch R1L1
 1. Check switch status, fabric status, and links
2. Delete spine switch S2
 1. Check switch status, fabric status, and links
 2. Verify configuration - show running switch
3. Restore configuration from snapshot Lab-2-Config-1
4. Shutdown interface R1L1-eth2 on switch R1L1
 1. config ; switch R1L1 ; interface ...
 2. Check interface status: show switch <> interface
 3. Check fabric topology: show link | grep R1L1
 4. There's no interface to S1 spine switch
 5. Check fabric status and links

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(9)

1. Configure interface group for host R1H1

1. config ; interface-group <> ; member switch ...
2. Interface group name: PG-R1H1
3. Add switch interfaces as members of this interface group
4. Members: R1L1 (R1L1-eth5) and R1L2 (R1L2-eth5)
5. Verify config: show this
6. Verify status: show interface-group ; show interface-group PG-R1H1 members

2. Configure remaining Interface groups

1. Interface Group: PG-R1H2 Members: R1L1 (R1L1-eth6), R1L2 (R1L2-eth6)
2. Interface Group: PG-R2H1 Members: R2L1 (R2L1-eth5), R2L2 (R2L2-eth5)
3. Interface Group: PG-R2H2 Members: R2L1 (R2L1-eth6), R2L2 (R2L2-eth6)

3. Pre-provision a new interface group “PG-test” with 6 members

1. Switch: R4L1 Interfaces: R4L1-eth1, R4L1-eth2, R4L1-eth3
2. Switch: R4L2 Interfaces: R4L2-eth1, R4L2-eth2, R4L2-eth3

4. Check all interface group configurations and interface group status

LAB – WORKING WITH CONTROLLERS & SWITCHES – P(10)

1. Perform the following operations using GUI

1. Delete (clear configurations for) switch S1 and R1L1
2. Check fabric status (Visibility → Fabric Summary or click on errors) (Figure 1)
3. Provision the deleted switches (S1 – 00:00:00:01:00:01, R1L1 – 00:00:00:02:00:01)
4. Pre-provision a new spine S9 – 00:00:00:01:00:09
5. Update switch view so that all SW versions are visible (Figure 2)
6. Discover links from S1 and R2L1 to all other switches
7. Shutdown 5 interfaces in a single operation across R1 and R2 leaf-groups
8. Confirm interface status from CLI
9. Check fabric status from GUI
10. Provision 3 additional interface-groups (Figure 3)
11. Validate changes from CLI
12. Bring-up all interfaces shutdown in R1 and R2 leaf-groups
13. Delete spine switch S9



Figure 1

Name	MAC	Connected	Fabric Status	Fabric Role
R1L2	00:00:00:02:00:02	✓	✓	Leaf
R2L1	00:00:00:02:00:03	✓	✓	Leaf

Figure 2

Interface-Group	Switch-Interface
PG-FW	R3L1-eth5 and R3L2-eth5
PG-RTR1	R3L1-eth6 and R3L2-eth6
PG-RTR2	R3L1-eth7 and R3L2-eth7

Figure 3

Questions?

Please stop here and let your instructor know that you have finished.

Module 4 – BCF Logical Topology

BCF LOGICAL TOPOLOGY

Module Outline

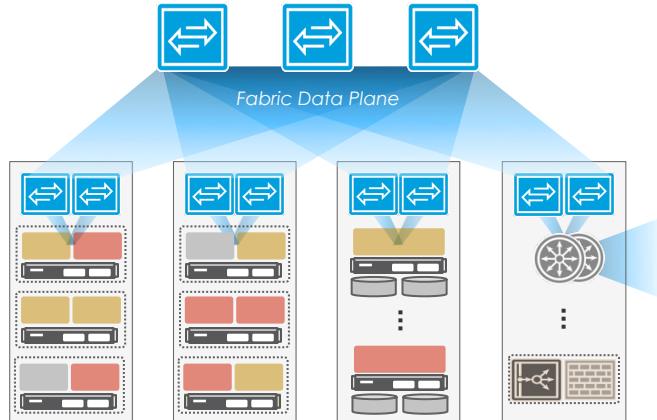
- Tenants
- Segments
- Logical Routers
- Policy
- **Lab – Configuring BCF Topologies**

BCF LOGICAL TOPOLOGY

Overview

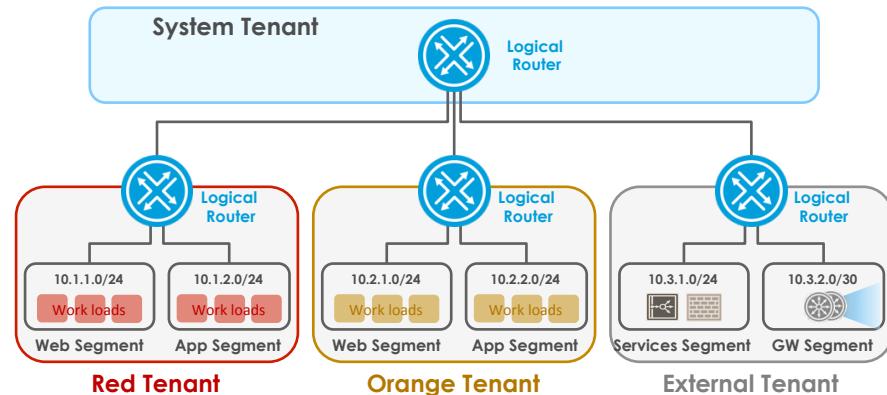
Physical Topology

- Uses spine & leaf switches



Logical Topology

- Uses tenants, segments & logical routers



BCF LOGICAL TOPOLOGY

Taxonomy & Hierarchy

Tenants

- Similar to VRF
- Establish layer 3 or domain boundary
- Support private addresses

Segments

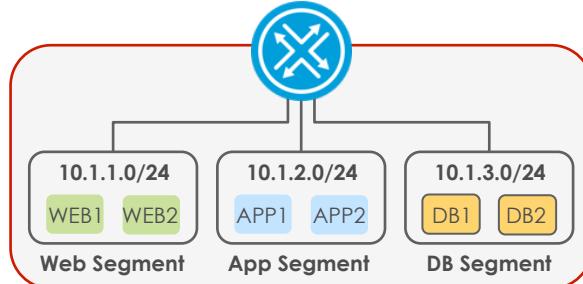
- Similar to VLAN / subnet
- Establish layer 2 boundary (broadcast domain)

End Points

- Host, VM, or network device
- Source traffic or receive traffic
- Dynamically learned or statically configured

Members

- Define attachment points where end-points are connected / learned
 - switch, interface, and VLAN
 - interface-group (LAG) and VLAN



BCF

↳ Tenants

↳ Segments

↳ Members

↳ End Points

BCF LOGICAL TOPOLOGY

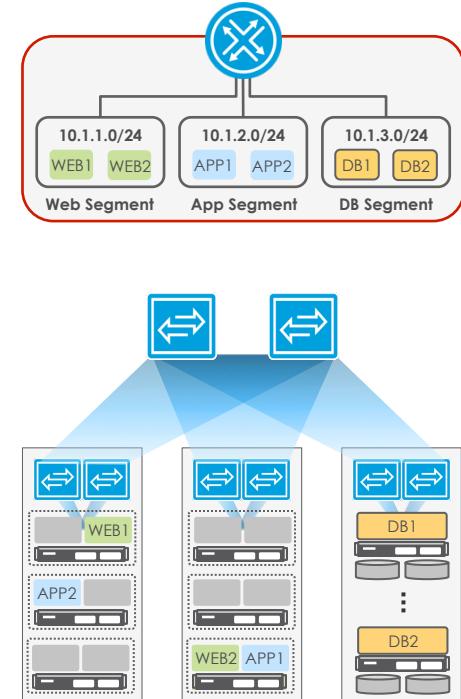
Configuring Tenants and Segments – Workloads in Racks 2 & 3

```
tenant Dev
  description "Application Development Tenant Network"

segment Web
  description "Dev Web subnet"
  member switch R2L1 interface ethernet5 vlan 100
  member switch R2L2 interface ethernet5 vlan 100

segment App
  description "Dev App subnet"
  member switch R2L1 interface ethernet5 vlan 200
  member switch R2L2 interface ethernet5 vlan 200

segment DB
  description "Dev DB subnet"
  member switch R3L1 interface ethernet10 vlan untagged
  member switch R3L2 interface ethernet10 vlan untagged
  member switch R3L1 interface ethernet20 vlan untagged
  member switch R3L2 interface ethernet20 vlan untagged
```



All servers with dual uplinks. Web2 & App1 – VMs. DB1 & DB2 – baremetal servers.

BCF LOGICAL TOPOLOGY

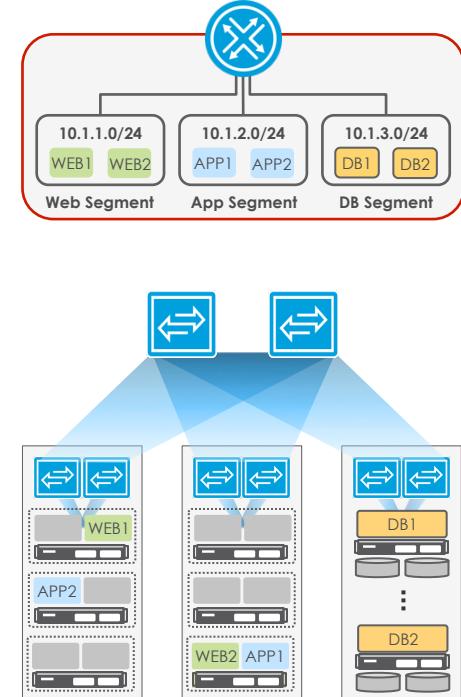
Configuring Tenants and Segments – Workloads in Racks 1 & 2

```
interface-group IG-Web1
  mode lacp
  member switch R1L1 interface ethernet12
  member switch R1L2 interface ethernet12

interface-group IG-App1
  mode lacp
  member switch R1L1 interface ethernet15
  member switch R1L2 interface ethernet15

tenant Dev
  segment Web
    description "Dev Web subnet"
    member interface-group IG-Web1 vlan untagged
    member switch R2L1 interface ethernet5 vlan untagged
    member switch R2L2 interface ethernet5 vlan untagged

  segment App
    description "Dev App subnet"
    member interface-group IG-App1 vlan 200
    member switch R2L1 interface ethernet5 vlan 200
    member switch R2L2 interface ethernet5 vlan 200
```

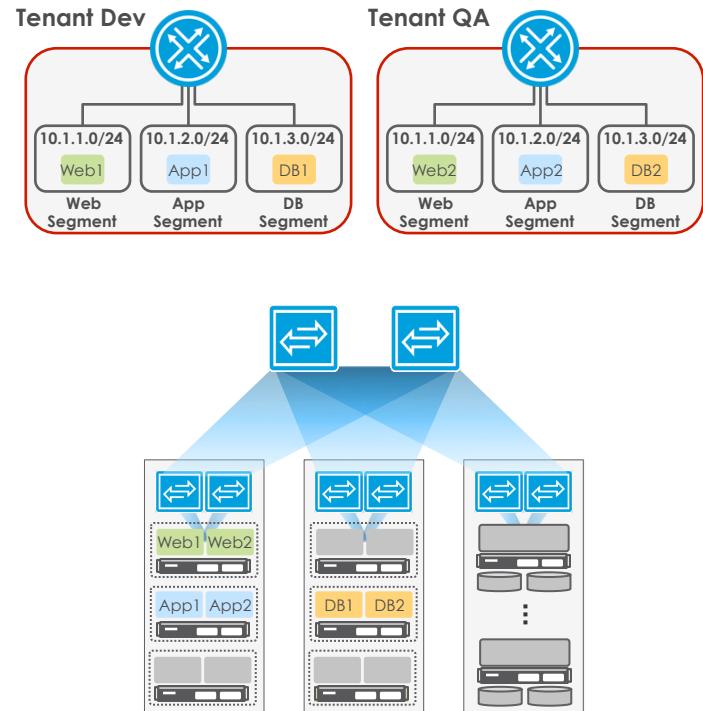


BCF LOGICAL TOPOLOGY

Configuring Tenants and Segments – Same Interface / Multiple Tenants

```
tenant Dev
  segment Web
    member switch R1L1 interface ethernet10 vlan 100
    member switch R1L2 interface ethernet10 vlan 100
  segment App
    member switch R1L1 interface ethernet20 vlan 200
    member switch R1L2 interface ethernet20 vlan 200
  segment DB
    member switch R2L1 interface ethernet30 vlan 300
    member switch R2L2 interface ethernet30 vlan 300

tenant QA
  segment Web
    member switch R1L1 interface ethernet10 vlan 500
    member switch R1L2 interface ethernet10 vlan 500
  segment App
    member switch R1L1 interface ethernet20 vlan 600
    member switch R1L2 interface ethernet20 vlan 600
  segment DB
    member switch R2L1 interface ethernet30 vlan 700
    member switch R2L2 interface ethernet30 vlan 700
```



BCF LOGICAL TOPOLOGY

Configuring Tenants and Segments – Wildcard Rules

```
tenant Dev

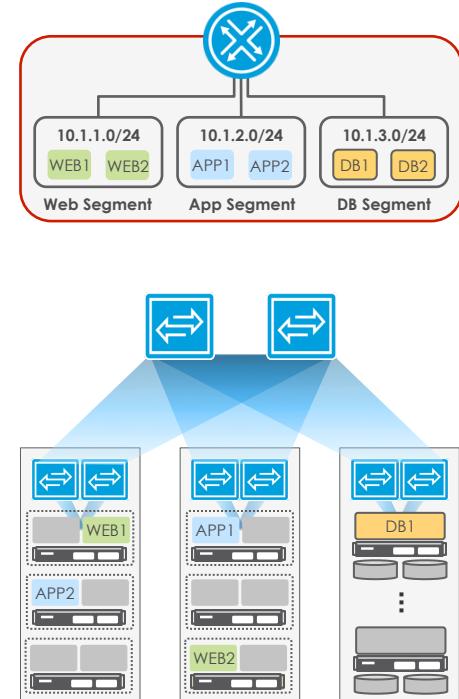
segment Web
    member interface-group any vlan untagged
    member switch any interface any vlan untagged

segment App
    member switch any interface ethernet20 vlan untagged

segment DB
    member switch R3L1 interface ethernet20 vlan untagged
    member switch R3L2 interface ethernet20 vlan untagged
```

Wildcard rules

- General (any switch; any interface)
- More specific (any switch; specific interface)
- Most specific (specific switch; specific interface)

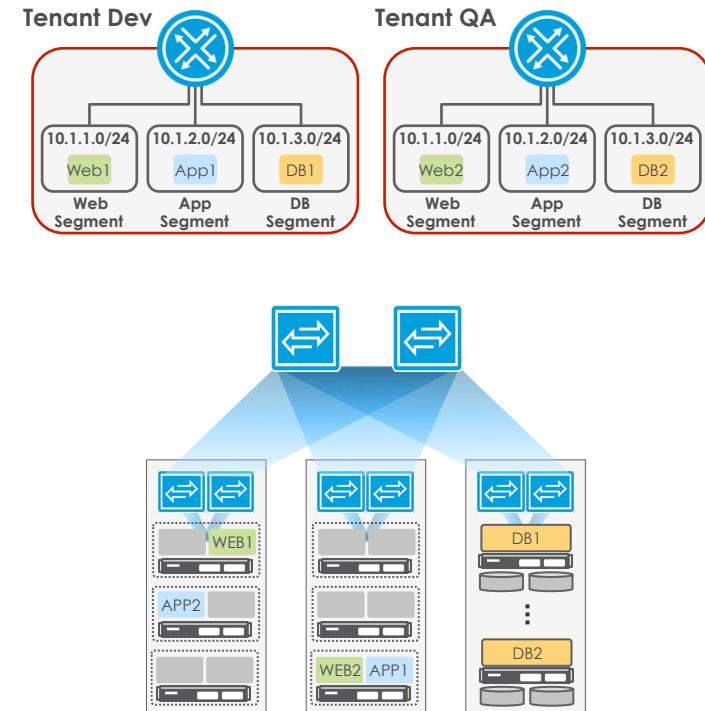


BCF LOGICAL TOPOLOGY

Configuring Tenants and Segments – Same VLAN / Multiple Segments

```
tenant Dev
  segment Web
    member switch R1L1 interface ethernet20 vlan untagged
    member switch R1L2 interface ethernet20 vlan untagged
  segment App
    member switch R2L1 interface ethernet10 vlan untagged
    member switch R2L2 interface ethernet10 vlan untagged
  segment DB
    member switch R3L1 interface ethernet22 vlan untagged
    member switch R3L2 interface ethernet22 vlan untagged
```

```
tenant QA
  segment Web
    member switch R2L1 interface ethernet10 vlan 100
    member switch R2L2 interface ethernet10 vlan 100
  segment App
    member switch R1L1 interface ethernet15 vlan 100
    member switch R1L2 interface ethernet15 vlan 100
  segment DB
    member switch R3L1 interface ethernet25 vlan 100
    member switch R3L2 interface ethernet25 vlan 100
```



BCF LOGICAL TOPOLOGY

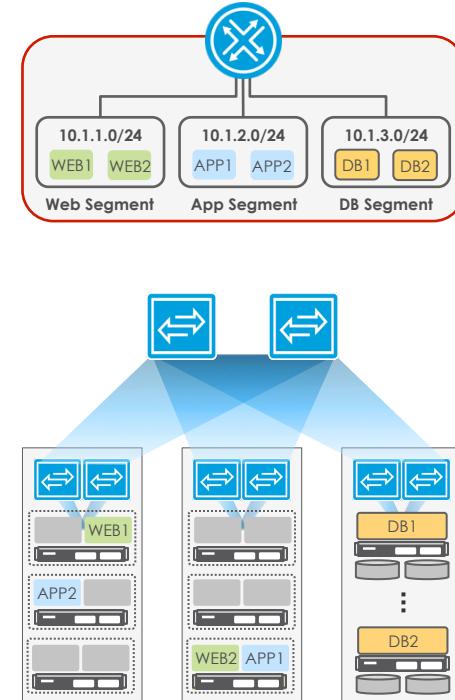
Configuring Tenants and Segments – Same Segment / Multiple VLANs

```
tenant Dev
  description "Application Development Tenant Network"

  segment Web
    description "Dev Web subnet"
    member switch R1L1 interface ethernet10 vlan 100
    member switch R1L2 interface ethernet10 vlan 100
    member switch R2L1 interface ethernet30 vlan 1100
    member switch R2L2 interface ethernet30 vlan 1100

  segment App
    description "Dev App subnet"
    member switch R1L1 interface ethernet20 vlan 200
    member switch R1L2 interface ethernet20 vlan 200
    member switch R2L1 interface ethernet30 vlan 1200
    member switch R2L2 interface ethernet30 vlan 1200

  segment DB
    description "Dev DB subnet"
    member interface-group any vlan 300
    member switch any interface any vlan 300
```



BCF LOGICAL TOPOLOGY

Configuring Tenants and Segments – Static Endpoints

```
tenant Prod
```

```
segment DB
```

```
member interface-group IG-DB1 vlan 500  
member interface-group IG-DB2 vlan 500
```

```
endpoint DB1
```

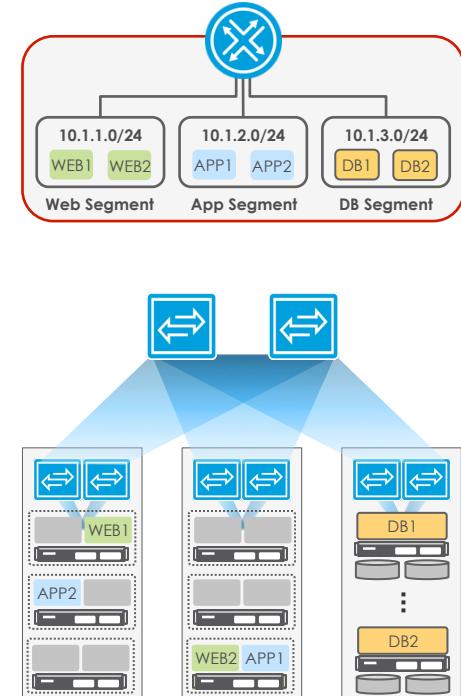
```
mac 00:00:03:01:02:03  
ip 10.1.3.12  
attachment-point interface-group IG-DB1 vlan 500
```

```
segment Web
```

```
member switch R1L1 interface ethernet10 vlan 100  
member switch R1L2 interface ethernet10 vlan 100  
member switch R2L1 interface ethernet30 vlan 100  
member switch R2L2 interface ethernet30 vlan 100
```

```
endpoint Web2
```

```
mac 00:00:01:06:07:08  
ip 10.1.1.75  
attachment-point switch R2L1 interface ethernet30 vlan 100  
attachment-point switch R2L2 interface ethernet30 vlan 100  
shutdown
```



BCF LOGICAL TOPOLOGY

End Points

End Points

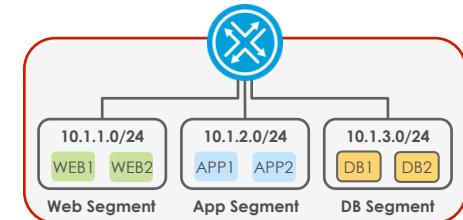
- Any host, device, or virtual machine that originates or terminates traffic
- Attached to segments
- Dynamically discovered or statically configured

Properties

- MAC Address
- IP Address
- Attachment point and type

Static configuration benefits

- MAC and IP address does not age out
- Duplicate MAC or IP cannot be learned on a different attachment point
- Reference end-point in various commands using alias
- Mis-behaving end-point may be shutdown



BCF LOGICAL TOPOLOGY

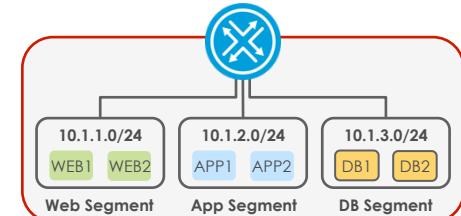
Segment Members

Members

- Define the attachment point for end-points
- Specify the traffic type to be assigned to a segment

Segment Membership

- Based on switch, interface, and VLAN
- Based on interface group and VLAN
- Based on MAC address
 - Traffic assigned to a given segment if the specified MAC address is seen anywhere on the fabric
- Wildcards allow any switch, any interface-group, or any interface
- No overlapping member rules allowed
 - Eg: VLAN 10 traffic on a given interface may only belong to one segment



BCF LOGICAL TOPOLOGY

Tenant & Segment Status

Status

- Tenant status
 - Provides information on segments, traffic counters, traffic rates, endpoints & attachment points
- Segment status
 - Provides information about traffic counters and traffic rates
 - To get endpoint related information, use tenant status command
- Membership status
 - Provides information on all the member related configurations
 - To which segment or segments is a specific VLAN mapped
 - To which segment or segments is a physical interface or interface-group mapped
- End point status
 - Provides information on the state of an endpoint
 - MAC; IP; attachment point; state; how long has it been active

```
controller# show tenant
controller# show segment
controller# show member-rule
controller# show endpoint
```

Questions?

BCF LOGICAL TOPOLOGY

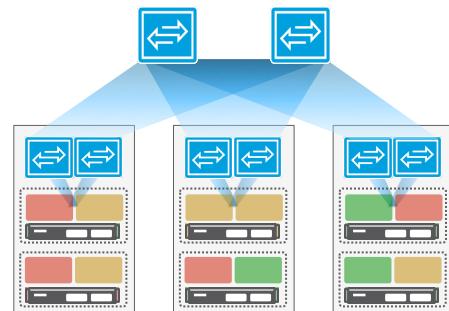
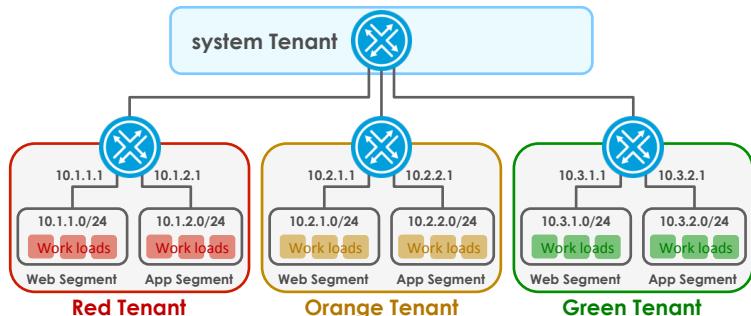
Logical Routers – Tenant and System

Tenant Logical Router

- Routes traffic within the tenant
- Supports BGP (iBGP/eBGP) & static routing
- Supports ACL and policy based routing
- Supports ECMP
- Configurable interfaces: segment and tenant
- Assign IP addresses to segment interfaces
- Relay DHCP requests
- Programmed on all leaf switches

System Logical Router

- Routes traffic across tenants
- Lives within the reserved tenant “system”
- Visibility to all networks within BCF
- Supports only static routing (no BGP)
- Supports ACL and policy based routing
- Supports redistribution of routes to tenant routers
- Configurable interfaces: tenant (unnumbered)
- Programmed on all spine switches



BCF LOGICAL TOPOLOGY

Logical Routers – Tenant and System

```
tenant Red
  logical-router
    interface tenant system

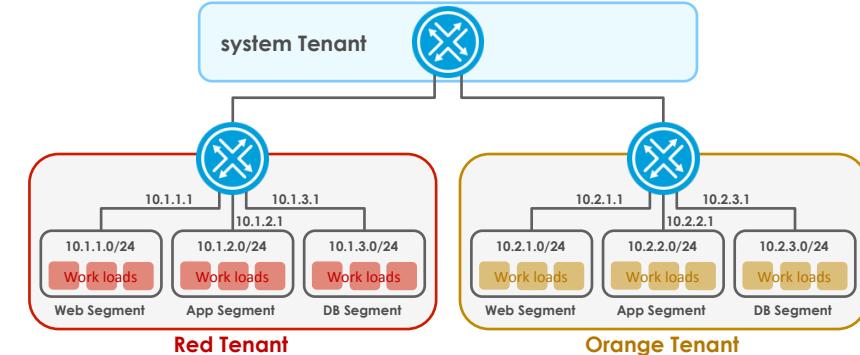
    interface segment Web
      ip address 10.1.1.1/24
      dhcp-relay server-ip 10.2.3.4

    interface segment App
      ip address 10.1.2.1/24
      ip address 10.1.174.1/24
      private

    interface segment DB
      ip address 10.1.3.1/24
      shutdown

    route 0.0.0.0/0 next-hop tenant system

tenant system
  logical-router
    interface tenant Red
    interface tenant Orange
```

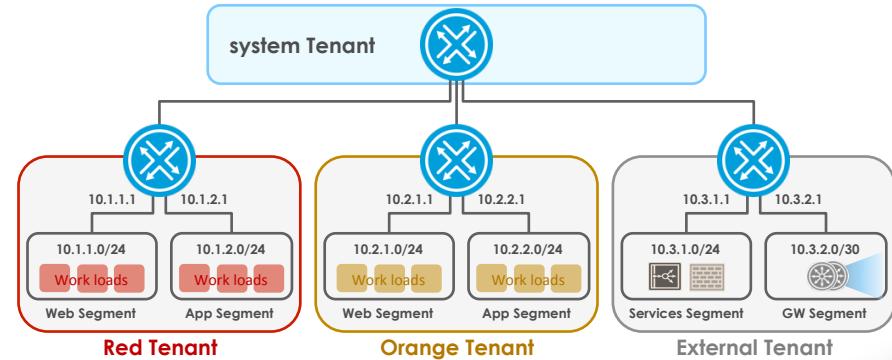
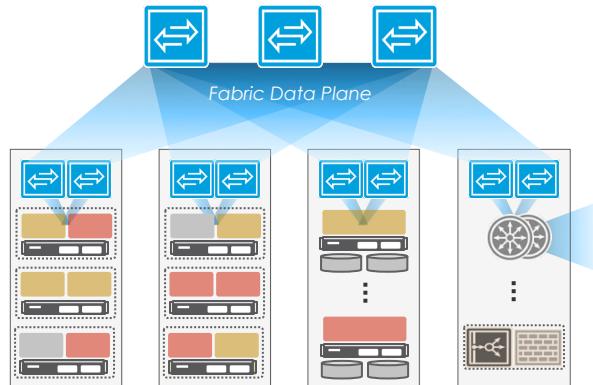


BCF LOGICAL TOPOLOGY

External Connectivity

External Connectivity

- Physically, external gateway router connects to a BCF leaf switch
- Logically, external gateway router connects to a segment within a tenant
 - Not system tenant
- Traffic from other tenants routed to this common gateway (via system tenant)

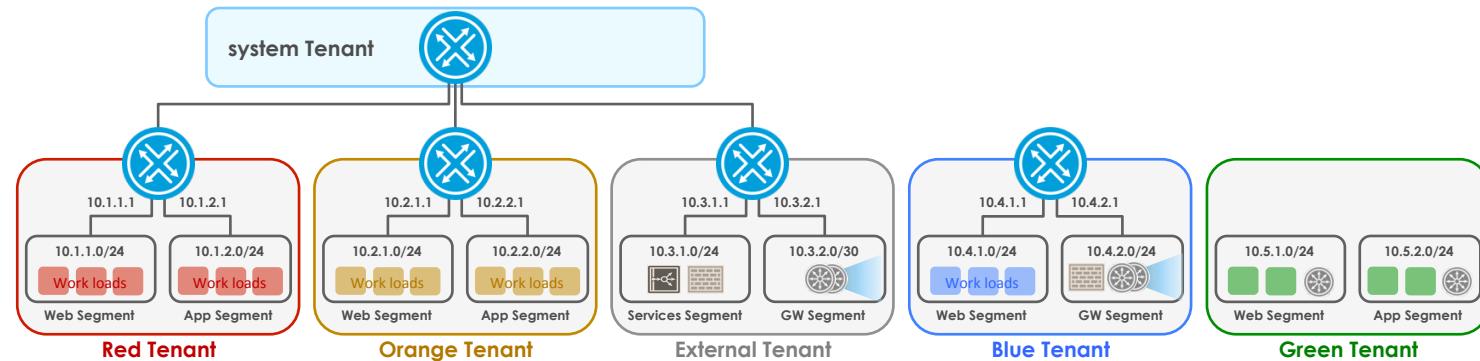


BCF LOGICAL TOPOLOGY

External Connectivity

External Connectivity Options

- Option 1: Shared external gateway (red and orange tenant)
- Option 2: External gateway per tenant (blue tenant)
- Option 3: External gateway per segment (green tenant)



BCF LOGICAL TOPOLOGY

ECMP (Equal-Cost Multi-Path)

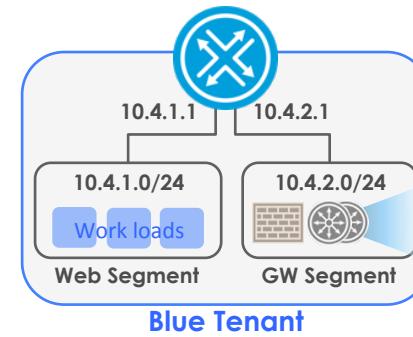
Next Hop Groups (ECMP)

- Provides multiple paths for static routes or for policy based routing
- Supports up to 16 next-hops
- Traffic load balanced across all destinations
- Provides multiple paths for increased bandwidth and availability

```
tenant Blue
  logical-router
    interface segment Web
      ip address 10.4.1.1
    interface segment GW
      ip address 10.4.2.1

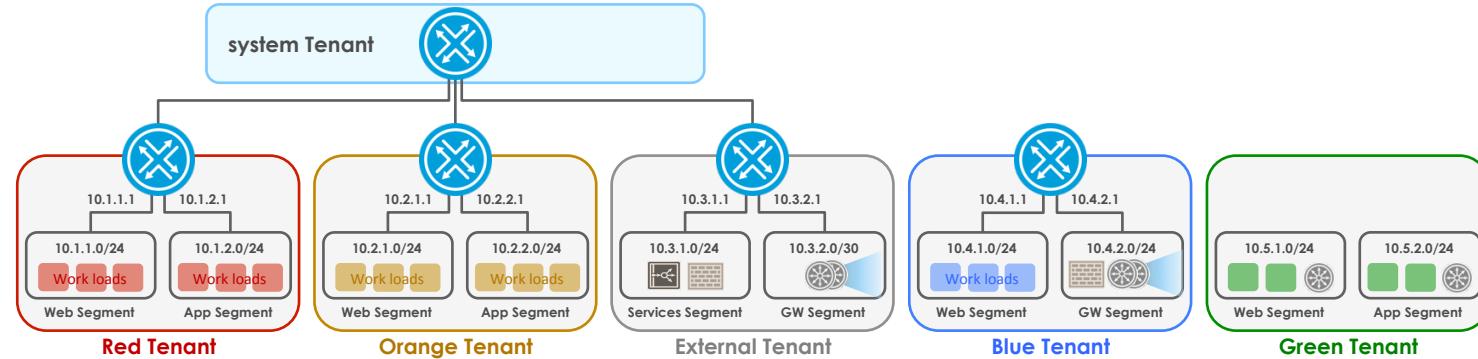
  next-hop-group gw-routers
    ip 10.4.2.2
    ip 10.4.2.3

  route 0.0.0.0/0 next-hop gw-routers
```



BCF LOGICAL TOPOLOGY

Inter-tenant Routing



```
tenant Red
logical-router
  interface tenant system
  interface segment Web
    ip address 10.1.1.1/24
  interface segment App
    ip address 10.1.2.1/24

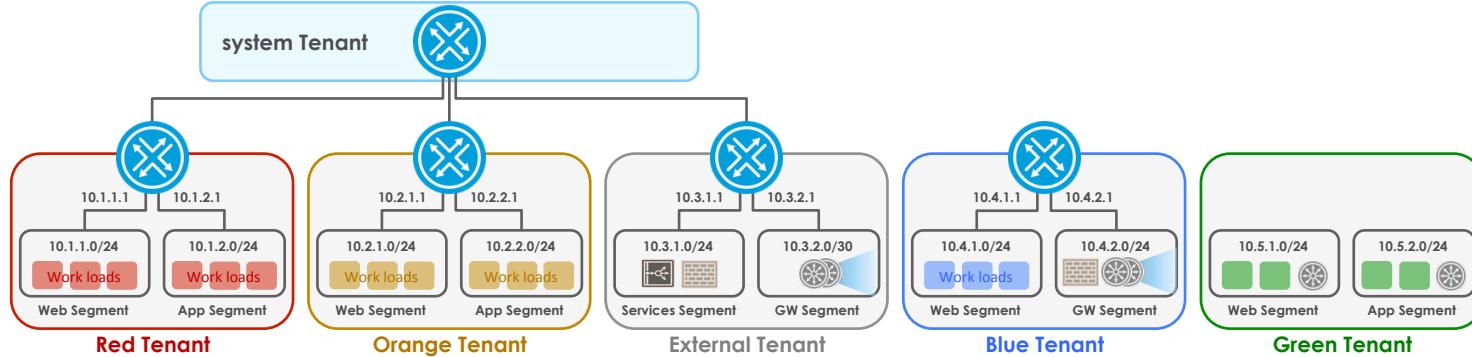
route 0.0.0.0/0 next-hop tenant system
```

```
tenant system
logical-router
  interface tenant Red
  interface tenant Orange
  interface tenant External

route 0.0.0.0/0 next-hop tenant External
```

BCF LOGICAL TOPOLOGY

Inter-tenant Routing



```
tenant External
logical-router
  interface tenant system
  interface segment Services
    ip address 10.3.1.1/24
  interface segment GW
    ip address 10.3.2.1/29
  next-hop-group gw-routers
    ip 10.3.2.2
    ip 10.3.2.3
  route 0.0.0.0/0 next-hop gw-routers
  route 10.0.0.0/8 next-hop tenant system
```

```
tenant Blue
logical-router
  interface segment Web
    ip address 10.4.1.1/24
  interface segment GW
    ip address 10.4.2.1/24
  next-hop-group ext-gw
    ip 10.4.2.2
    ip 10.4.2.3
  route 0.0.0.0/0 next-hop ext-gw
```

BCF LOGICAL TOPOLOGY

Inter-tenant Route Distribution

```
tenant system
  logical-router
    interface tenant Red
    interface tenant Orange
    interface tenant External
      export-route

    route 0.0.0.0/0 next-hop tenant External

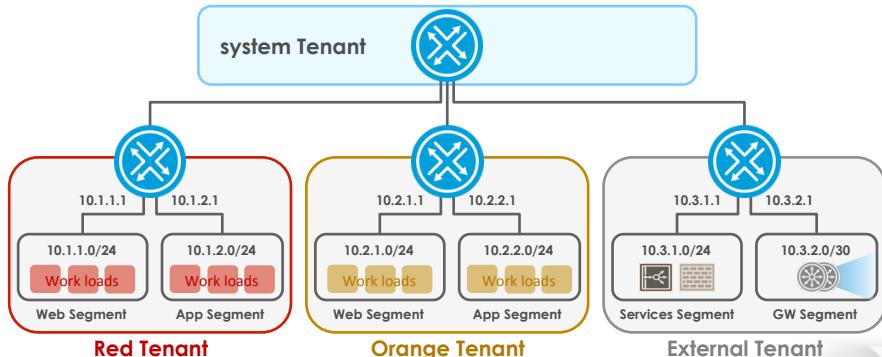
tenant External
  logical-router
    interface tenant system
      import-route

    interface segment Services
      ip address 10.3.1.1/24
    interface segment GW
      ip address 10.3.2.1/29
    next-hop-group gw-routers
      ip 10.3.2.2
      ip 10.3.2.3

    route 0.0.0.0/0 next-hop gw-routers
```

Inter-tenant Route Distribution

- System router has visibility to all BCF connected routes
 - Exception – Non-routable private tenant routes
- Logical tenant router has visibility only to tenant routes
 - Red - 10.1/16; Orange – 10.2/16; External – 10.3/16
- System router can redistribute its routes
 - System router exports; Logical router imports



Questions?

BCF LOGICAL TOPOLOGY

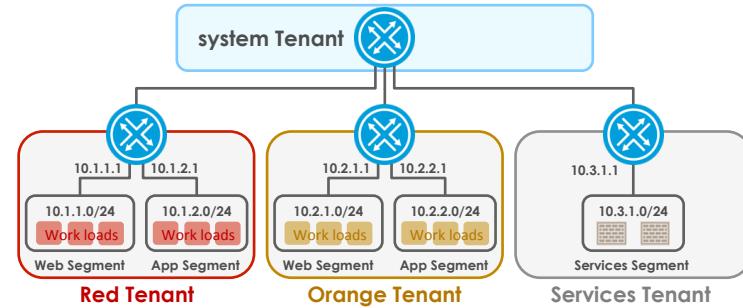
Policy – Overview

Capability

- Implement ACLs
- Implement policy based routing
- Implement service insertion
- Implement packet capture for troubleshooting

Implementation

- One policy per logical router (tenant and system)
 - Multiple policies may be defined
 - Only one policy can be applied
- Applies to logical router not interfaces
- Applies to all traffic traversing the router
- Processing stops when traffic matches a policy statement
- Policies end with implicit deny action
- Declarative mode configuration model (vs. Imperative mode)

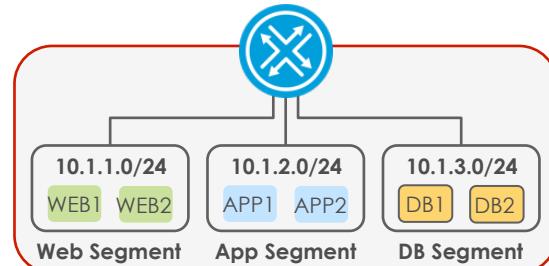


BCF LOGICAL TOPOLOGY

Policy Structure

Building policy statements

1. Sequence Numbers
 - Statement evaluated in numerical order
2. Permit or Deny action
3. Source traffic matching criteria
 - Tenant or segment interface (**optional**)
 - Tenant, Segment, IP, Protocol, TCP/UDP Port, or any
4. Keyword “to”
5. Destination traffic matching criteria
 - Tenant, Segment, IP, TCP/UDP Port, or any
6. Next hop destination for permitted traffic (**optional**)



```
10 deny 10.1.100.0/24 to 10.1.200.0/24
20 deny tenant red to tenant green
30 deny segment-interface web any to any
40 permit any to any
```

BCF LOGICAL TOPOLOGY

ACL (Access Control List) – Example 1

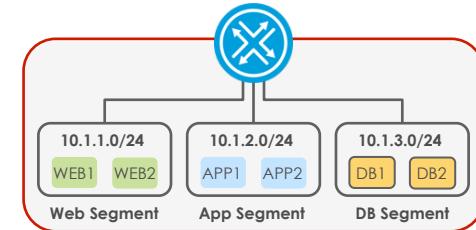
Build a policy to prevent any communication between Web and DB segment

```
tenant Dev
  logical-router
    interface segment Web
      ip address 10.1.1.1/24
    interface segment App
      ip address 10.1.2.1/24
    interface segment DB
      ip address 10.1.3.1/24

  policy-list p1
    10 deny 10.1.1.0/24 to 10.1.3.0/24
    99 permit any to any

  policy-list p2
    10 deny tenant Dev segment Web to tenant Dev segment DB
    99 permit any to any

  apply policy-list p1
```



Build Policy

- Use CIDR blocks or
- Use logical topology elements

Apply Policy

- Applies to router
- Not an interface

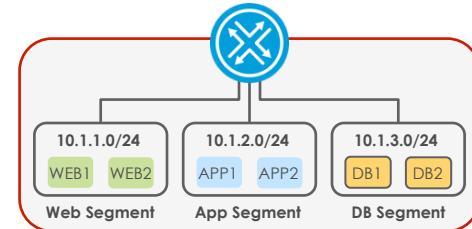
BCF LOGICAL TOPOLOGY

ACL (Access Control List) – Example 2

Build a policy that permits ssh to App segment only

- 1st permit ssh to App segment
- 2nd deny all other ssh traffic
- Then permit all other traffic

```
tenant Dev
  logical-router
    policy-list p1
      10 permit proto tcp any to tenant Dev segment App port 22
      20 deny proto tcp any to any port 22
      99 permit any to any
    apply policy-list p1
```



BCF LOGICAL TOPOLOGY

Service Insertion – Inter-segment Traffic (1)

```
tenant Dev
  logical-router
    interface segment Web
      ip address 10.1.1.1/24
    interface segment App
      ip address 10.1.2.1/24
    interface segment DB
      ip address 10.1.3.1/24
    interface segment FW
      ip address 10.1.4.1/24

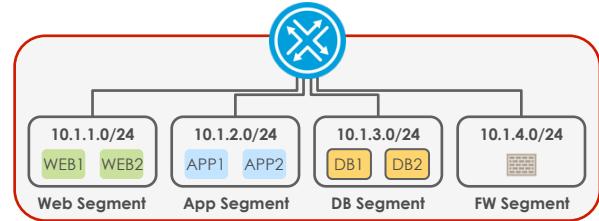
  next-hop-group Dev-FW
    ip 10.1.4.10

  policy-list p1
    10 permit segment-interface Web any to any next-hop Dev-FW
    20 permit segment-interface App any to any next-hop Dev-FW
    30 permit segment-interface DB any to any next-hop Dev-FW
    99 permit any to any

  apply policy-list p1
```

Criteria

- All inter-segment traffic must be directed through a local firewall



Policy P1

- Redirects traffic from specific segments to the firewall
- Allows all other traffic to be routed normally

BCF LOGICAL TOPOLOGY

Service Insertion – Inter-segment Traffic (2)

```
tenant Dev
logical-router
  interface segment Web
    ip address 10.1.1.1/24
  interface segment App
    ip address 10.1.2.1/24
  interface segment DB
    ip address 10.1.3.1/24
  interface segment FW
    ip address 10.1.4.1/24

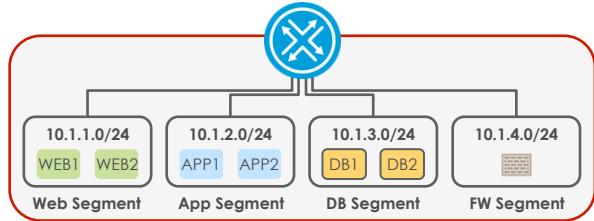
next-hop-group Dev-FW
  ip 10.1.4.10

policy-list p1
  10 permit segment-interface FW any to any
  20 permit any to any next-hop Dev-FW

apply policy-list p1
```

Criteria

- All inter-segment traffic must be directed through a local firewall



Policy P1

- Allows traffic from FW segment to be routed normally
- Redirects all traffic to FW segment

BCF LOGICAL TOPOLOGY

Service Insertion – Inter-tenant Traffic

Criteria

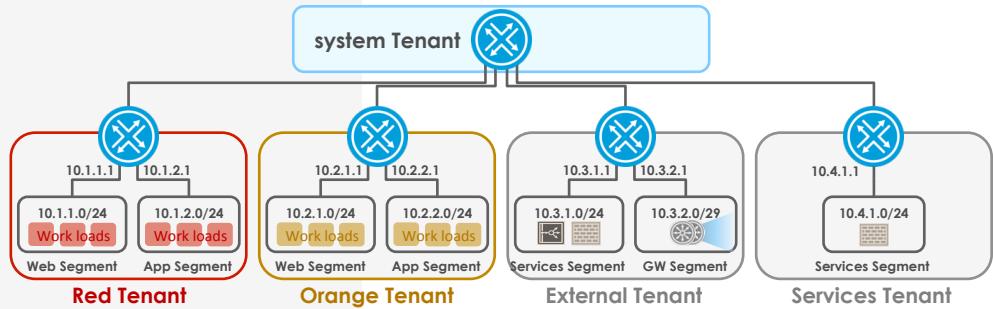
- All inter-tenant traffic uses Data Center Firewall in the Services Tenant
- All external bound traffic uses Perimeter Firewall in the External Tenant

```
tenant system
logical-router
  interface tenant Red
  interface tenant Orange
  interface tenant External
  interface tenant Services

  route 0.0.0.0/0 next-hop tenant External

  policy-list p1
    10 permit tenant-interface Services any to any
    20 permit tenant-interface External any to any
    30 permit tenant Red to tenant Orange next-hop tenant Services
    40 permit tenant Orange to tenant Red next-hop tenant Services
    99 permit any to any

  apply policy-list p1
```



From Services / External Tenant

- Process using normal routing

To Red / Orange Tenant

- Re-direct traffic to Services Tenant

To External Tenant

- Process using normal routing

BCF LOGICAL TOPOLOGY

Service Insertion – Inter-tenant Traffic

Criteria

- All inter-tenant traffic uses Data Center Firewall in the Services Tenant
- All external bound traffic uses Perimeter Firewall in the External Tenant

```
tenant system  
logical-router
```

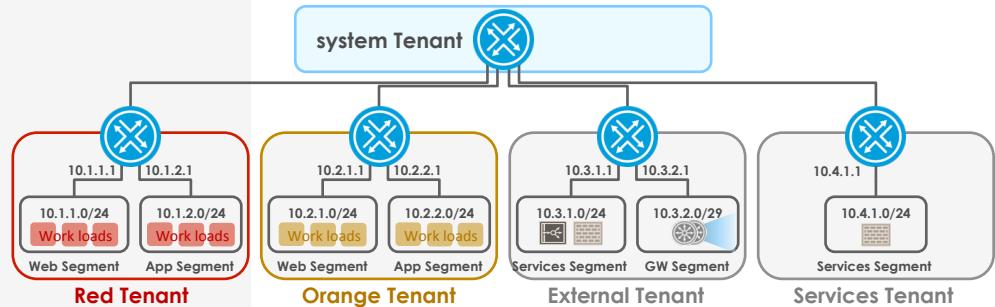
```
    interface tenant Red  
    interface tenant Orange  
    interface tenant External  
    interface tenant Services
```

```
route 0.0.0.0/0 next-hop tenant External
```

```
policy-list p1
```

```
 10 permit tenant-interface Services any to any  
 20 permit tenant-interface External any to any  
 30 permit any to tenant Orange next-hop tenant Services  
 40 permit any to tenant Red next-hop tenant Services  
 99 permit any to any
```

```
apply policy-list p1
```



What about when new tenant is added?

- Statement 30 & 40 reduces the number of new policy statements required as new tenants are added

BCF LOGICAL TOPOLOGY

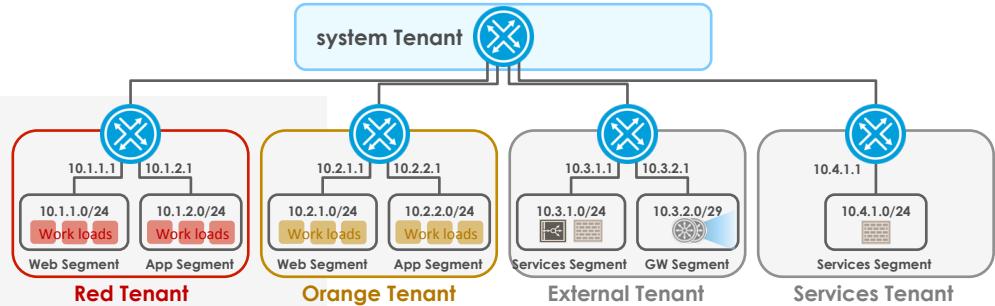
Service Insertion – Inter-tenant Traffic

```
tenant Services
  logical-router
    interface tenant system
    interface segment Services
      ip address 10.4.1.1/24
      route 0.0.0.0/0 next-hop tenant system

      next-hop-group DC-FW
        ip 10.4.1.10

      policy-list p1
        10 permit tenant-interface system any to any next-hop DC-FW
        99 permit any to any

      apply policy-list p1
```



From system Tenant

- Re-direct traffic to Data Center Firewall

All Other Traffic

- Process using normal routing

BCF LOGICAL TOPOLOGY

Service Insertion – Inter-tenant Traffic

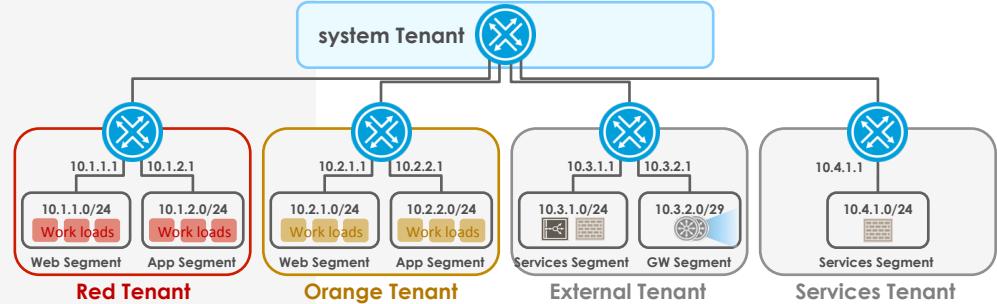
```
tenant External
  logical-router
    interface tenant system
    interface segment Services
      ip address 10.3.1.1/24
    interface segment GW
      ip address 10.3.2.1/29

    route 0.0.0.0/0 next-hop GW-routers
    route 10.0.0.0/8 next-hop tenant system

  next-hop-group Ext-FW
    ip 10.3.1.2
  next-hop-group GW-routers
    ip 10.3.2.3
    ip 10.3.2.4

  policy-list p1
    10 permit tenant-interface system any to any next-hop Ext-FW
    20 permit segment-interface GW any to any next-hop Ext-FW
    30 permit segment-interface Services any to any

  apply policy-list p1
```



From system Tenant

- Re-direct traffic to Perimeter Firewall

From GW Segment

- Re-direct traffic to Perimeter Firewall

From Services Segment

- Process using normal routing

BCF LOGICAL TOPOLOGY

Summary

Tenants

- Define logical networks
- Built using Logical elements
- Span entire fabric

Shared Tenant

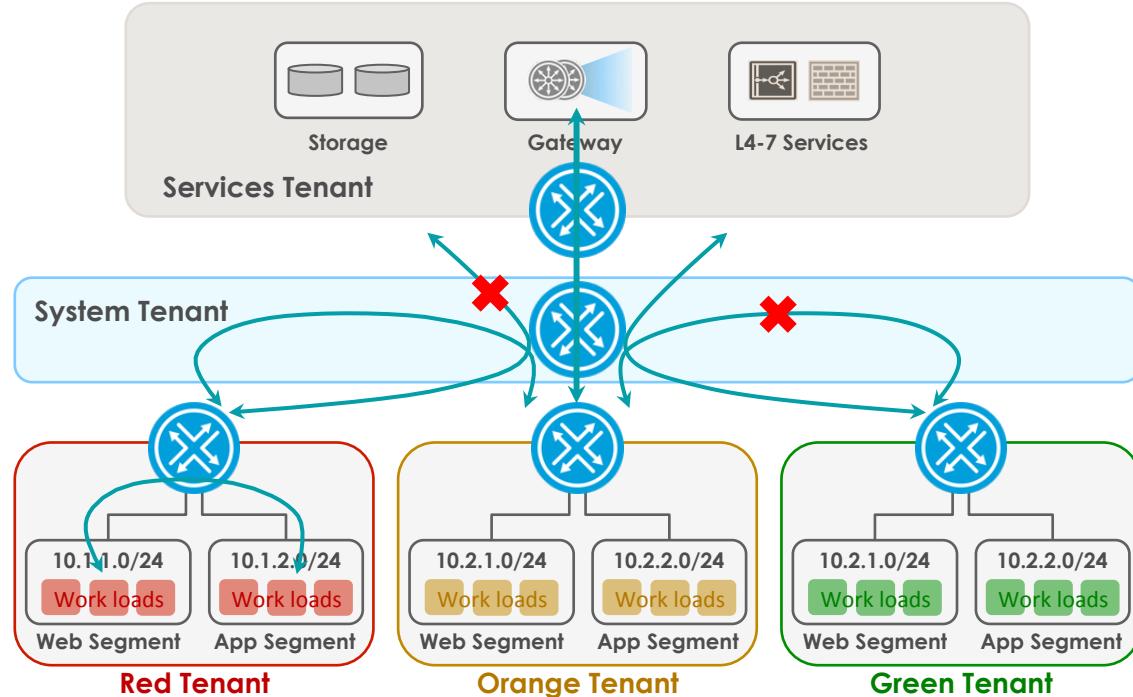
- Same as other tenants
- Provides shared services (storage, Firewall & external gateway)

System Tenant

- Provides inter-tenant connectivity

Granular Controls

- Inter-segment communication
- Inter-tenant communication
- Accessing shared services
- Accessing external network



Software Defined Data Center

Questions?

TRAINING OUTLINE

Day 2

Module 4: BCF Logical Topology (Cont'd)

- **Lab – Configuring BCF Topologies** 120 mins

Module 5: BCF Additional Configurations

- QoS, Multicast
- BGP (Connect to L3 network)
- Inter-POD Connectivity (L3, L2 – VLAN/VxLAN)
- Extending Segments
- BPDU Protection (Connect to xSTP enabled network)
- Storm Control
- sFlow, SPAN

Module 6: BCF Troubleshooting

- Tools & Capabilities
- Flow Chart
- Starting Points
- **Lab – Troubleshooting BCF** 120 mins

BSN LABS

Launch Module

- Sign into <http://labs.bigswitch.com/training>
 - Use userid and password provided by instructor
- Launch BCF Module 1



Lab – Configuring BCF Topologies

LAB OBJECTIVES

- Provision tenants and segments
- Enable intra-tenant routing
- Enable inter-tenant routing
- Connect BCF to external gateways
- Load balance traffic using ECMP Next-Hop Groups
- Implement ACLs using policies

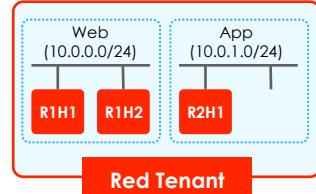
LAB – CONFIGURING BCF TOPOLOGIES – PAGE (1)

1. Load configuration for this lab

- copy snapshot://BCF-Config3 running-config

2. Create a tenant and a segment

- Create a tenant named “Red” (Figure 1)
- Create a segment named “Web” within tenant Red
- Add a segment member: interface group – PG-R1H1, traffic type – untagged
- Add a second member: interface group – PG-R1H2, traffic type – untagged
- Verify configuration: show this and show running tenant
- Check configuration again by typing “exit” then “show this”
- Check configuration again by typing “exit” then “show this”
- Check tenant status: show tenant
- Check segment status: show segment



```
tenant T1
segment S1
  member switch R1L1 interface ethernet20 vlan 37
segment S2
  member interface-group G5 vlan untagged
```

3. Create a second segment named App

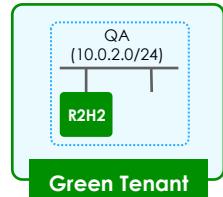
- Assign interface group PG-R2H1 to this segment as untagged

Figure 1 - Sample configuration

LAB – CONFIGURING BCF TOPOLOGIES – PAGE (2)

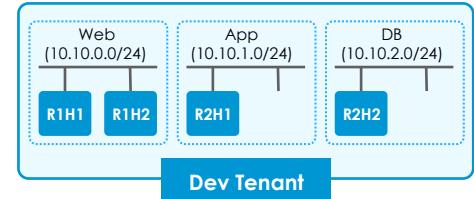
1. Create the Green tenant

1. Create segment QA
2. Assign member: interface group – PG-R2H2, traffic type – untagged



2. Create a new tenant named Dev

1. Create same segments under Dev as in Red (Web, App) and DB
2. Add same interface groups used for Red and Green tenant
3. Instead of traffic-type “untagged”, traffic-type for this tenant will be VLAN 100
4. Segment: Web Interface group: PG-R1H1 VLAN: 100
5. Segment: Web Interface group: PG-R1H2 VLAN: 100
6. Segment: App Interface group: PG-R2H1 VLAN: 100
7. Segment: DB Interface group: PG-R2H2 VLAN: 100
8. Verify configuration
9. Check member rules: show member-rule
10. All “untagged” traffic from the 4 hosts will be mapped to either Red or Green tenant
11. All “VLAN 100” traffic from the same 4 hosts will be mapped to Dev tenant



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (3)

1. Since R1H1 and R1H2 are in the same segment, can they ping each other?
 1. Start a shell on host R1H1 (Figure 1)
 2. R1H1> ping r1h2
2. Check that endpoints (R1H1 and R1H2) have been learned
 1. show endpoint
 2. Note attachment point, VLAN, and IP State
 3. These endpoints belong to tenant Red since ping traffic was untagged
3. Check tenant, segment and fabric status
 1. show tenant ; show segment ; show fabric
 2. Fabric status also lists the tenants, segments, and endpoints
4. Test inter-segment connectivity
 1. Can R1H1 in Web segment ping R2H1 in App segment?
 2. R1H1> ping r2h1
 3. **Why can't R1H1 and R2H1 ping each other?**

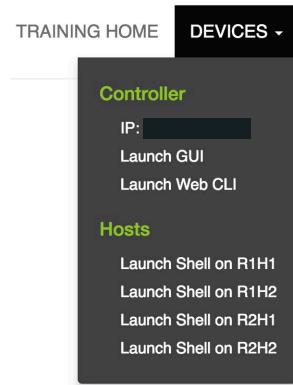
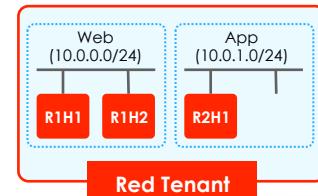


Figure 1



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (4)

1. Create a logical router for the Red tenant

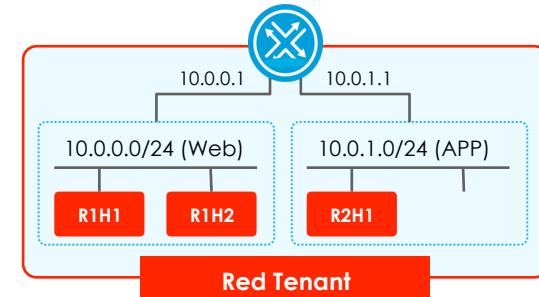
1. Create an interface for Web segment and assign IP address 10.0.0.1/24 (Figure 1)
2. Create an interface for App segment and assign IP address 10.0.1.1/24
3. Verify configuration: “show this”
4. Check configuration by typing “exit” then “show this”
5. Check configuration again by typing “exit” then “show this”
6. Check state of logical router: show logical-router
7. Check logical router interfaces: show logical-router ...
8. Check routes on this logical router

2. Test inter-segment connectivity again

1. R1H1> ping r2h1

3. Check endpoints

1. There should be three endpoints learned in tenant Red
2. Now that tenant is performing L3 routing, endpoint “IP State” is no longer “learned-L2 Only”



```
tenant T1
logical-router
  interface segment S1
    ip address 120.14.3.27/24
  interface tenant system
```

Figure 1 - Sample configuration

LAB – CONFIGURING BCF TOPOLOGIES – PAGE (5)

1. Create a logical router for the Green tenant

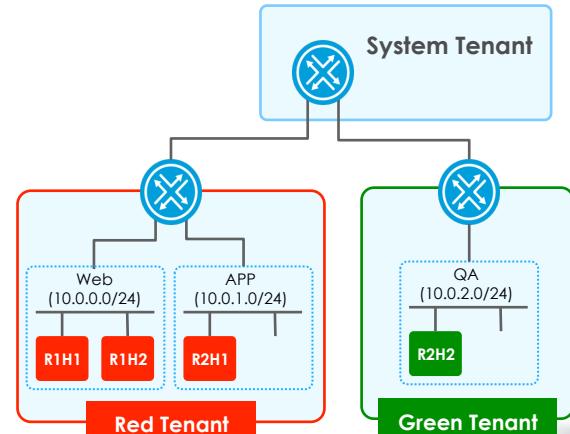
1. Create an interface for QA segment and assign IP address 10.0.2.1/24
2. Since there's no second host in QA segment, ping R2H2's default gateway
3. Check endpoints to verify R2H2 has been learned

2. Create the system tenant

1. “system” is a reserved keyword. Must be all lowercase
2. Create a logical router for the “system” tenant
3. Create interfaces for Red and Green tenants
4. Verify configuration

3. Test inter-tenant connectivity

1. R1H1> ping r2h2
2. R2H2> ping r1h1
3. **Why can't these hosts ping each other yet?**
4. Can R1H1 and R2H2 ping their gateway?



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (6)

1. Check inter-tenant connectivity

1. show logical-router
2. Check “Connected to System Router” column
3. show logical-router interface
4. Check “Tenant Interface State Table”
5. show logical-router incomplete
6. Shows incomplete L3 runtime state

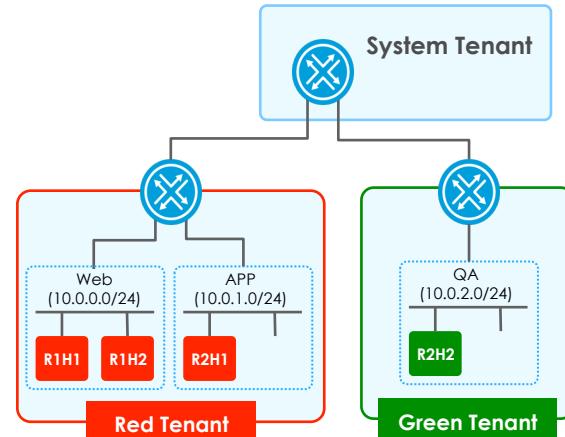
2. Connect Red & Green tenants to “system” tenant

1. Previously an interface was created in “system” tenant for Red and Green tenants
2. Create an interface in Red tenant logical router for “system” tenant
3. Create an interface in Green tenant logical router for “system” tenant

3. Check inter-tenant connectivity again

4. Test inter-tenant connectivity again

1. **Why can't hosts still ping each other?**



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (7)

1. Check routing tables and default routes

1. show logical-router (check “Default route state”)
2. Check the Red tenant routing table
3. show tenant Red logical-router route

2. Add default route to Red and Green tenant routers

1. tenant <> ; logical-router ; route 0.0.0.0/0 ...
2. Set next hop as tenant “system”

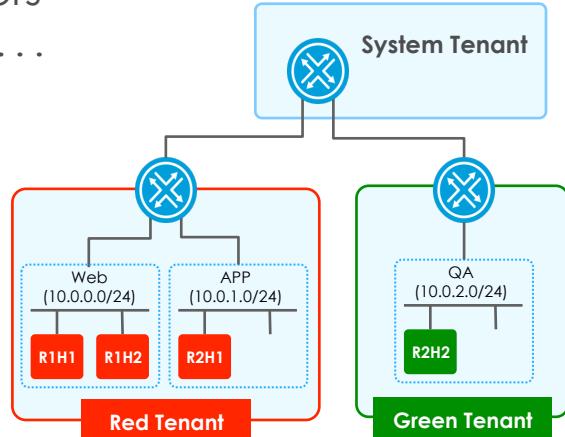
3. Test inter-tenant connectivity

1. All fabric hosts should be able to ping each other

4. Check endpoints

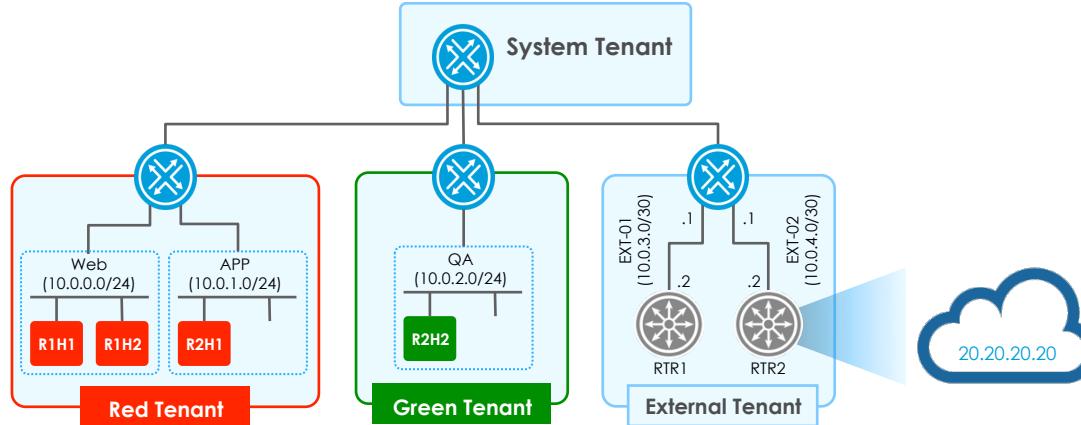
5. Check logical-router routing tables

1. Red Tenant routing table
2. Green tenant routing table
3. system tenant routing table



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (8)

1. Create a new tenant named External (use CLI or GUI for these steps)
 1. Create a segment Ext-01 and assign interface-group PG-RTR1 as untagged
 2. Create a second segment Ext-02 and assign interface-group PG-RTR2 as untagged
 3. Create a logical router
 4. Create an interface for segment Ext-01 and assign IP 10.0.3.1/30
 5. Create an interface for segment Ext-02 and assign IP 10.0.4.1/30
 6. Create an interface for system tenant



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (9)

1. External tenant configuration cont'd (use CLI or GUI for these steps)
 1. Create static routes for 10.0.0.0/24, 10.0.1.0/24, and 10.0.2.0/24 with nexthop as system tenant
 2. Create an ECMP group (next-hop-group) called EG1 to load balance outbound traffic
 3. tenant <> ; logical-router ; next-hop-group ...
 4. Assign IP 10.0.3.2 and 10.0.4.2 to this group
 5. Create a default route for the internet with next-hop as EG1

2. Can Red and Green tenants reach External tenant IP addresses?

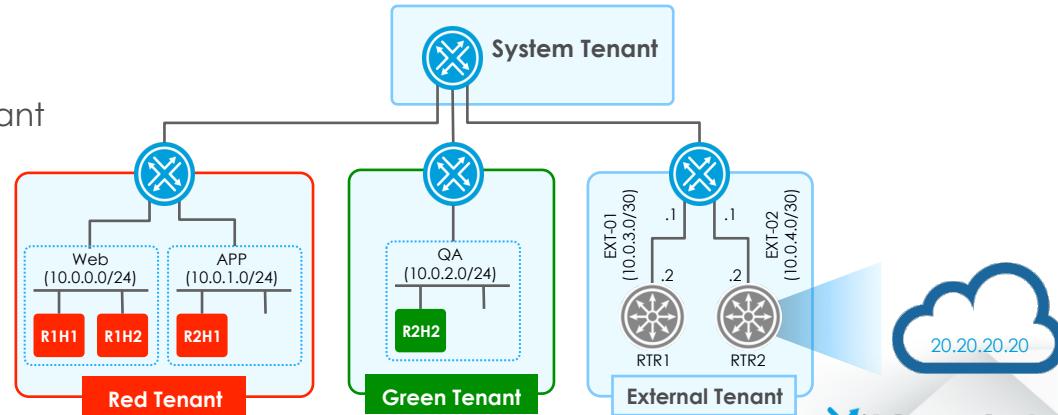
1. R1H1> ping 10.0.3.1 and ping 10.0.3.2

3. Update system tenant router

1. Add interface for external tenant

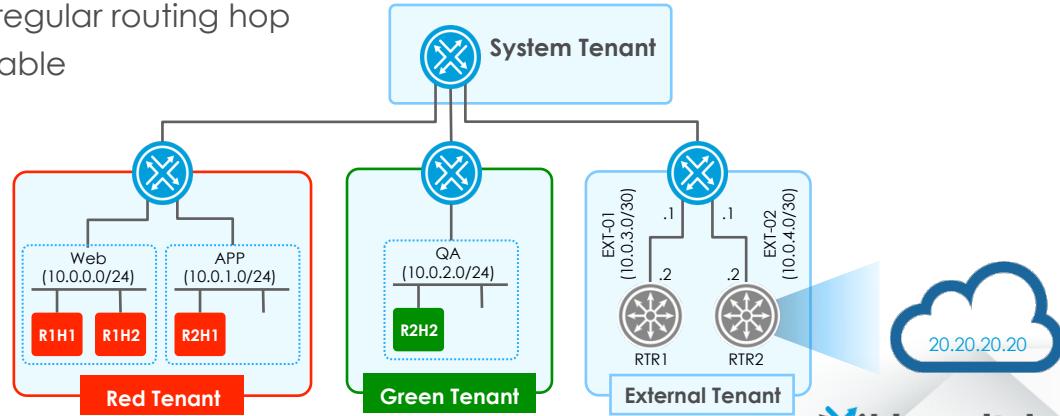
4. Test ping again

1. R1H1> ping 10.0.3.1
 2. R1H1> ping 10.0.3.2
 3. R1H1> ping 10.0.4.1
 4. R1H1> ping 10.0.4.2



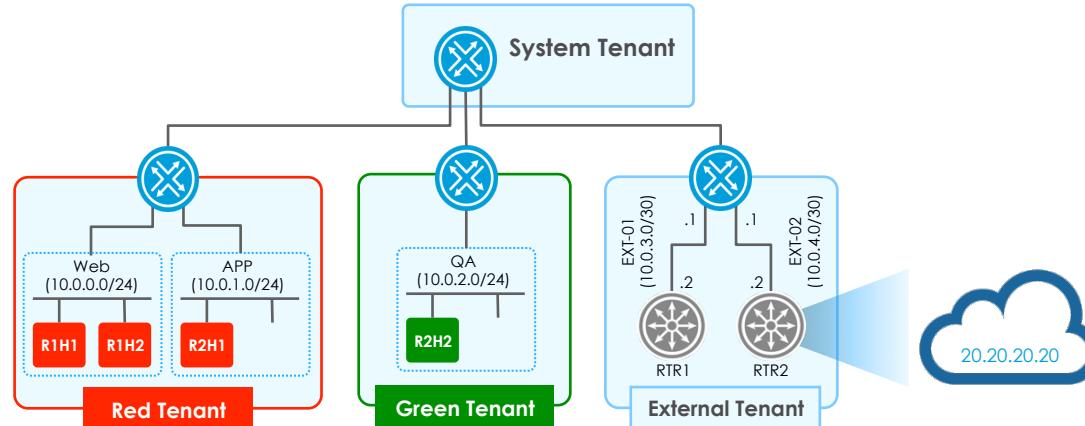
LAB – CONFIGURING BCF TOPOLOGIES – PAGE (10)

1. Check that two new endpoints have been learned
2. Test external connectivity
 1. Can Red and Green tenants reach internet?
 2. R1H1> ping 20.20.20.20
 3. **Why can't fabric hosts reach the internet?**
3. Check the routing path from R1H1 to Internet via each hop
 1. Trace forward and reverse path to see where connectivity is broken
 2. Treat each logical-router as a regular routing hop
 3. Check each tenant's routing table



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (11)

1. Add default route to system tenant router
 1. Add default route with next-hop as External tenant
 2. System tenant doesn't need routes for any of the fabric segments/tenants.
 3. When a tenant connects to system router, it learns all the tenant routes.
2. Test external connectivity again
 1. All fabric hosts should be able to reach Internet



LAB – CONFIGURING BCF TOPOLOGIES – PAGE (12)

1. Implement ACL using policy to block inter-tenant connectivity between Red and Green tenants

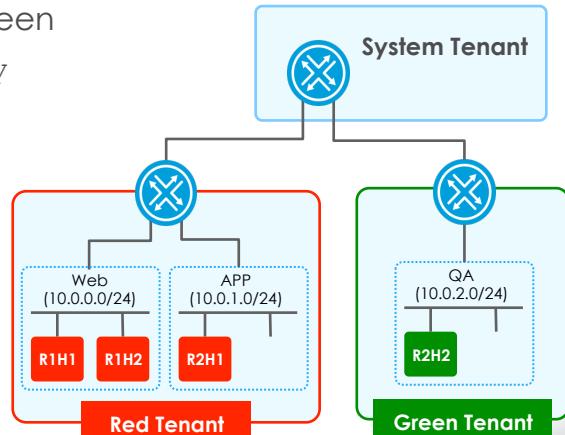
1. Can be implemented in system tenant or the User-defined tenants
2. **Solution 1:** Implement policy on Red Tenant
3. Create policy "p1" on Red tenant (Figure 1)
4. Deny traffic from tenant Red segment Web to tenant Green
5. Deny traffic from tenant Red segment App to tenant Green
6. Do not forget implicit deny all: 99 permit any to any
7. Apply policy for it to take effect

2. Test connectivity

1. R1H1> ping r2h2 (Red → Green traffic blocked)
2. R1H1> ping r2h1 (Red → Red traffic allowed)
3. R1H1> ping world (Red → External traffic allowed)
4. Try the same pings from Green tenant

```
tenant T1
logical-router
policy-list p1
 10 deny tenant T2 to tenant T3
 20 deny any to 12.2.1.0/24
 99 permit any to any
apply policy-list p1
```

Figure 1 - Sample configuration



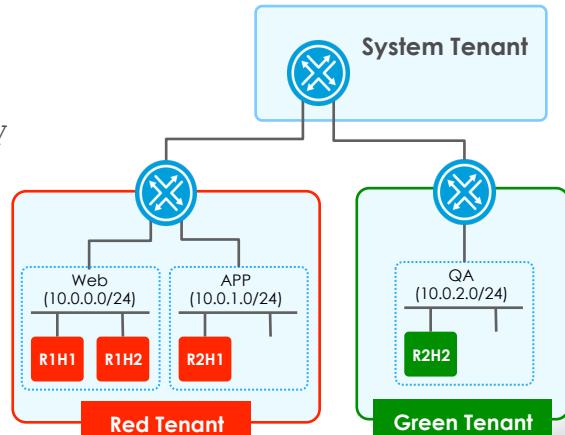
LAB – CONFIGURING BCF TOPOLOGIES – PAGE (13)

1. Implement ACL using policy to block inter-tenant connectivity between Red and Green tenants

1. **Solution 2:** Implement policy on system tenant
2. Remove policy from Red tenant logical router (do not forget to delete apply statement)
3. Verify that Red and Green tenant hosts can ping each other
4. Create policy “p1” on system tenant
5. `tenant system ; logical-router ; policy p1 ; ...`
6. Deny traffic from tenant Red to tenant Green
7. Deny traffic from tenant Green to tenant Red
8. Do not forget implicit deny all: `99 permit any to any`
9. Apply policy for it to take effect

2. Test connectivity

1. `R1H1> ping r2h2` (Red → Green traffic blocked)
2. `R1H1> ping r2h1` (Red → Red traffic allowed)
3. `R1H1> ping world` (Red → External traffic allowed)



Questions?

Please stop here and let your instructor know that you have finished.



Module 5 – BCF Additional Configurations

BCF ADDITIONAL CONFIGURATIONS

Module Outline

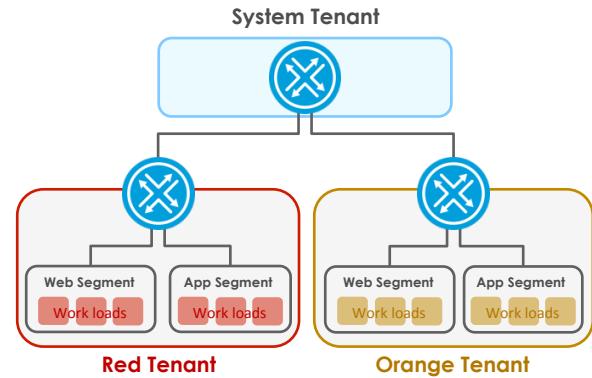
- QoS
- Multicast
- BGP
- Inter-POD Connectivity (L3, L2 – VLAN/VxLAN)
- Extended Segments
- Auto Host Detection
- BPDU Guard & Connecting to STP Enabled Network
- Storm Control
- sFlow
- SPAN (Local / Fabric)

BCF ADDITIONAL CONFIGURATIONS

QoS – Overview

Classification Modes

- Two modes
- Based on source segment
 - Ex: Tenant Red, Segment Web → Traffic Class 2
- Based on DSCP value
 - Ex: AF11 → Traffic Class 3
- Applicable fabric wide



Traffic Classes & Queues

- 4 traffic classes and 4 traffic queues
- Pre-defined mapping
- Weighted Round Robin (WRR) queues
- Traffic defaults to Traffic Class 0

Traffic Class → Queue		
0	→	0
1	→	1
2	→	2
3	→	3

BCF ADDITIONAL CONFIGURATIONS

QoS – Queues

Default Mode (QoS not Enabled)

Queue	Default Mode	% Bandwidth
0	All traffic	95%
5	Fabric SPAN, sFlow	5%
7	Inband management traffic	
8	Control packets (LACP, LLDP, ...)	

Strict Priority Queue

Fabric QoS Enabled

Queue	Traffic Class	Weight
0	0	10
1	1	20
2	2	30
3	3	30
4	PFC	
5	5	5
7	Inband management traffic	
8	Control packets (LACP, LLDP, ...)	

Weight is user configurable.
Shown values are examples only

BCF ADDITIONAL CONFIGURATIONS

QoS – Traffic Classes & Weights

Example 1

- Equal weights
- Equal bandwidth

Queue	Traffic Class	WRR Weight	WRR Weight
0	0	20	37
1	1	20	37
2	2	20	37
3	3	20	37
5	5	20	37

Example 2

- Q0 – 2 x Q5
- Q1 – 4 x Q5
- Q2 – 6 x Q5
- Q3 – 6 x Q5

Queue	Traffic Class	WRR Weight	WRR Weight
0	0	10	2
1	1	20	4
2	2	30	6
3	3	30	6
5	5	5	1

BCF ADDITIONAL CONFIGURATIONS

QoS – Segment Based Marking Configuration

```
fabric
  qos
    active
    queuing-profile Q-Profile1
      traffic-class traffic-class-0 weight 30
      traffic-class traffic-class-1 weight 10
      traffic-class traffic-class-2 weight 10
      traffic-class traffic-class-3 weight 10
      traffic-class traffic-class-span-fabric weight 10
    apply Q-Profile1
  mode segment
```

Web Segment

- All traffic uses default queue 0

App Segment

- All traffic uses the dedicated queue 1
- Q1 capacity – 1/3 of Q0

DB Segment

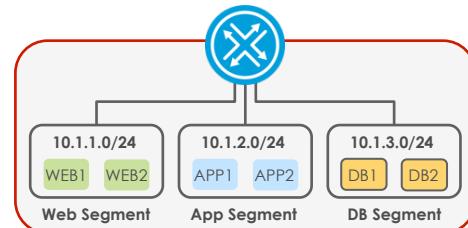
- All traffic uses the dedicated queue 2
- Q2 capacity – 1/3 of Q0

Note: Segment classification uses 802.1p bits

```
tenant Dev
  segment Web
    member switch R2L1 interface ethernet5 vlan 100
    member switch R2L2 interface ethernet5 vlan 100

  segment App
    member switch R2L1 interface ethernet5 vlan 200
    member switch R2L2 interface ethernet5 vlan 200
    qos-traffic-class traffic-class-1

  segment DB
    member switch R3L1 interface ethernet10 vlan 300
    member switch R3L2 interface ethernet10 vlan 300
    qos-traffic-class traffic-class-2
```

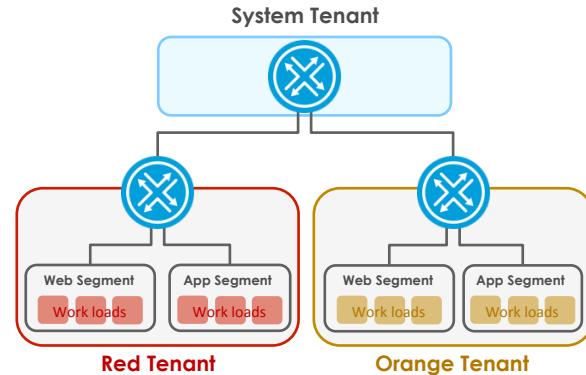


BCF ADDITIONAL CONFIGURATIONS

QoS – DSCP Based Marking Configuration

```
fiberic
  qos
    active
    queuing-profile Q-Profile1
      traffic-class traffic-class-0 weight 30
      traffic-class traffic-class-1 weight 10
      traffic-class traffic-class-2 weight 10
      traffic-class traffic-class-3 weight 10
      traffic-class traffic-class-span-fabric weight 10
    apply Q-Profile1

    classification-profile DSCP-Profile1
      traffic-class traffic-class-1
        dscp af11
        dscp af12
      traffic-class traffic-class-2
        dscp be
    mode dscp classification-profile DSCP-Profile1
```



Traffic Class 1

- Matches “af11”, “af12” traffic regardless of segment or tenant
- Traffic goes out using queue 1

Traffic Class 2

- Matches all “be” traffic regardless of segment or tenant
- Traffic goes out using queue 2

All Other Traffic

- Uses queue 0

BCF ADDITIONAL CONFIGURATIONS

QoS – View Settings & Status

QoS Status

- Check QoS setting (weights and percentages)

```
controller# show fabric qos  
controller# show forwarding qos-queue-weight-table
```

- Check traffic class assignments to segments

```
controller# show segment
```

- View queue counters

```
controller# show switch <> interface <> queue <> counters
```

Questions?

BCF ADDITIONAL CONFIGURATIONS

Multicast – Overview

Multicast

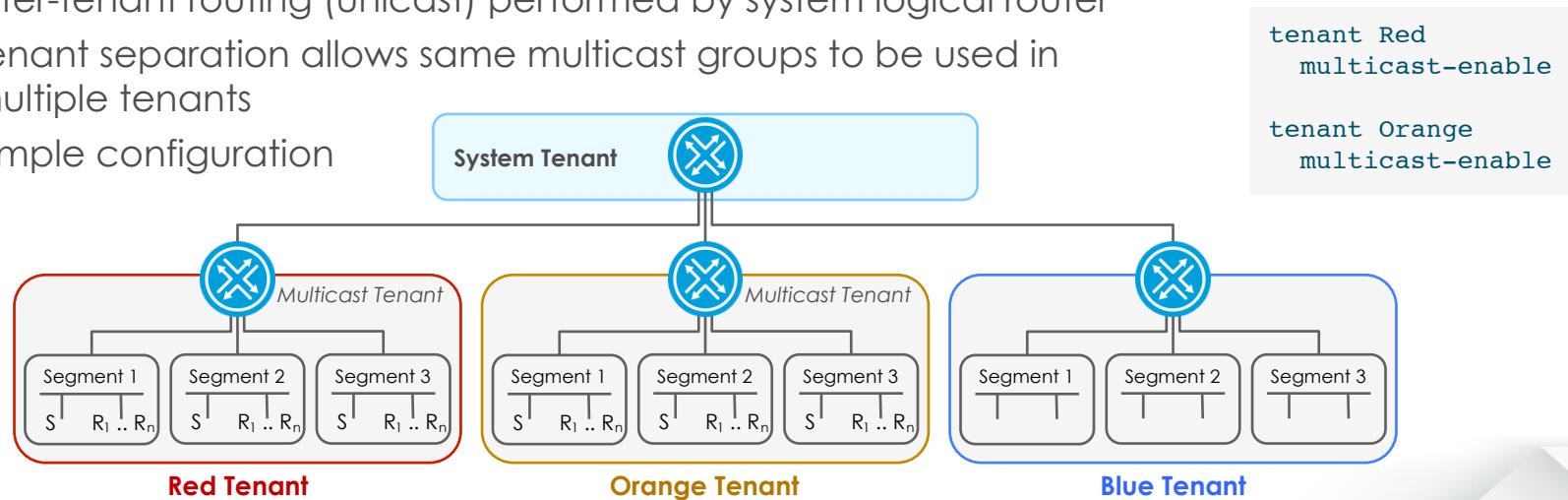
- Disabled by default
 - Enable on a tenant by tenant basis
- Supports IGMP v1 and v2
- Supports IGMP snooping
- Supports L3 multicast forwarding
 - Sources & Receivers in a single tenant
 - Sources & Receivers in multiple tenants
- Supports static multicast groups
- Supported on P-switches and V-switches

BCF ADDITIONAL CONFIGURATIONS

Multicast – Sources & Receivers in a Single Tenant

Intra-tenant Multicast

- All sources & receivers are contained within the same tenant
- Sources & Receivers may span across any number of segments
- Intra-tenant routing (unicast & multicast) performed by tenant logical router
- Inter-tenant routing (unicast) performed by system logical router
- Tenant separation allows same multicast groups to be used in multiple tenants
- Simple configuration

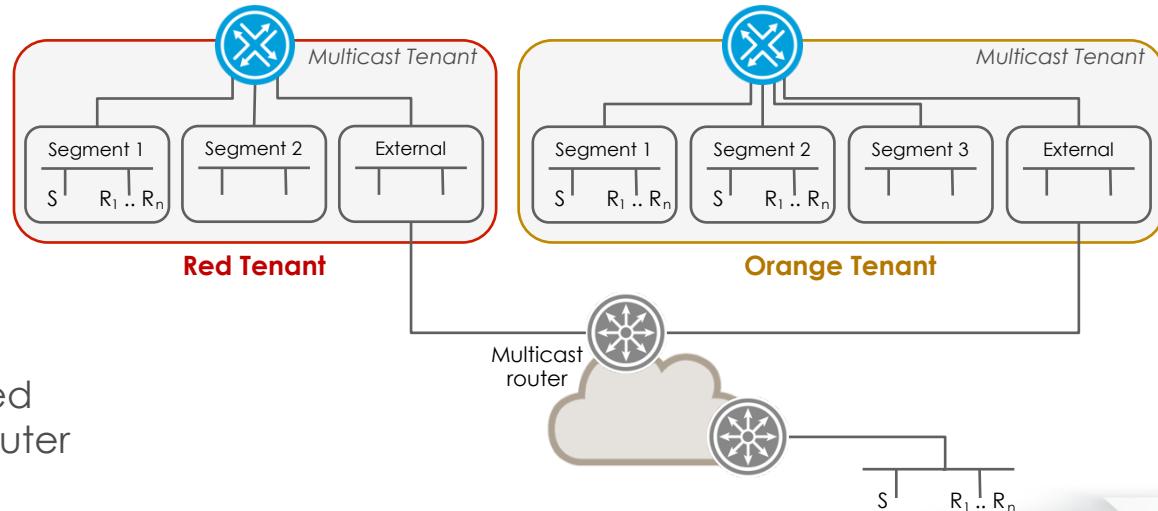


BCF ADDITIONAL CONFIGURATIONS

Multicast – Sources & Receivers in Multiple Tenants

Inter-tenant Multicast

- External gateway required per tenant
 - Handles all inter-tenant traffic (unicast & multicast)
 - No system tenant
- BCF Source External Receivers
 - Static IGMP joins required on BCF for all multicast groups
- External Source BCF Receivers
 - Static IGMP joins required on external gateway router for all multicast groups



BCF ADDITIONAL CONFIGURATIONS

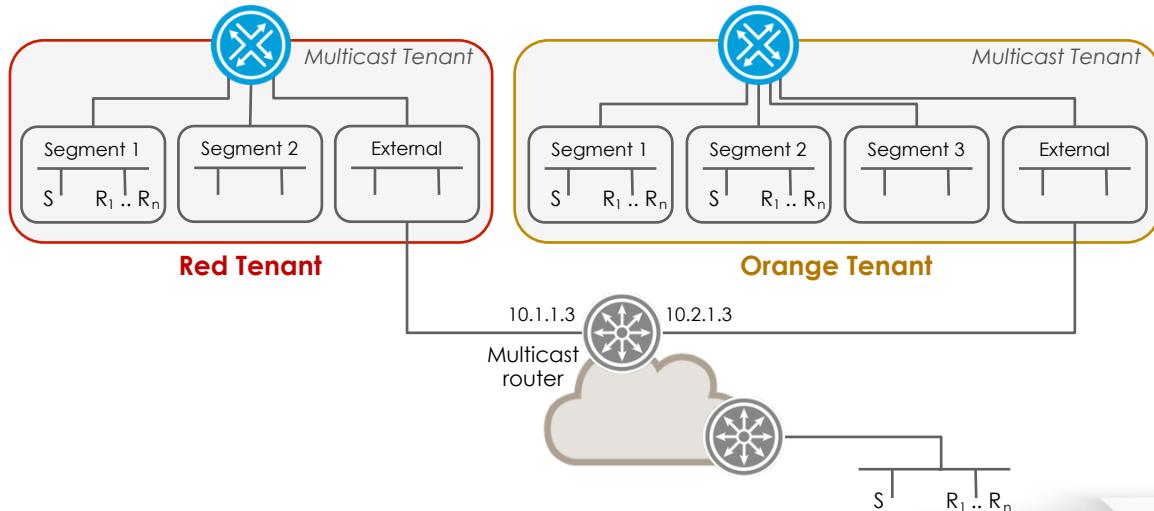
Multicast – Sources & Receivers in Multiple Tenants

Inter-tenant Multicast

- Enable multicast feed from BCF to external receivers or receivers in other BCF tenants
 - Red Tenant: Groups 228.0.0.0 and 228.10.15.5
 - Orange Tenant: Groups 230.0.0.0 and 230.10.15.5

```
tenant Red
multicast-enable
multicast-group 228.0.0.0
  listener ip 10.1.1.3
multicast-group 228.10.15.5
  listener ip 10.1.1.3
```

```
tenant Orange
multicast-enable
multicast-group 230.0.0.0
  listener ip 10.2.1.3
multicast-group 230.10.15.5
  listener ip 10.2.1.3
```



BCF ADDITIONAL CONFIGURATIONS

Multicast – Commands

Multicast Status

- List of L2 & L3 multicast groups

```
controller# show multicast l2groups  
controller# show multicast l3groups
```

- Summary and other multicast commands

```
controller# show multicast mrouter  
controller# show multicast tenant  
controller# show multicast summary
```

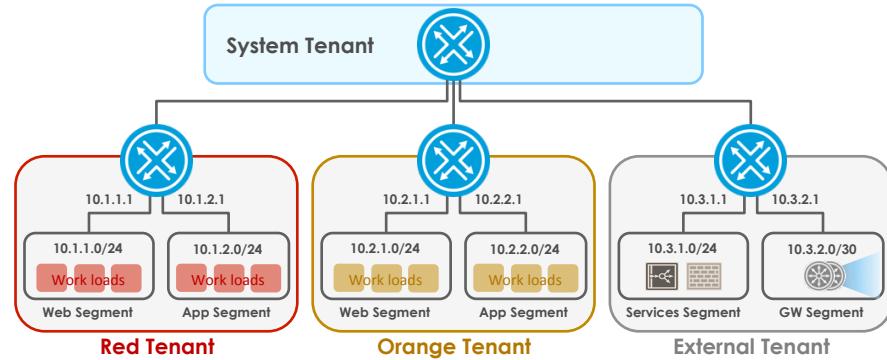
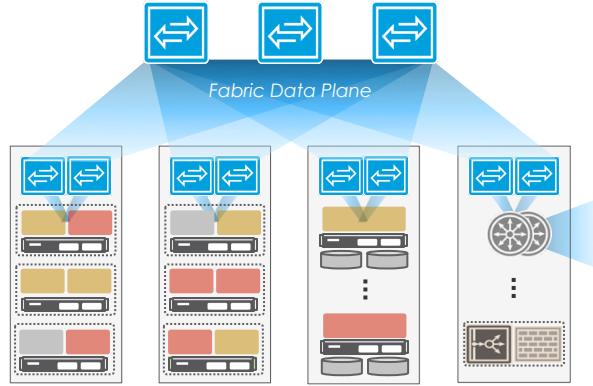
- Debug commands

```
controller# show debug igmp-manager . . .  
controller# show debug mcast-manager . . .
```

Questions?

BCF ADDITIONAL CONFIGURATIONS

BGP – Overview



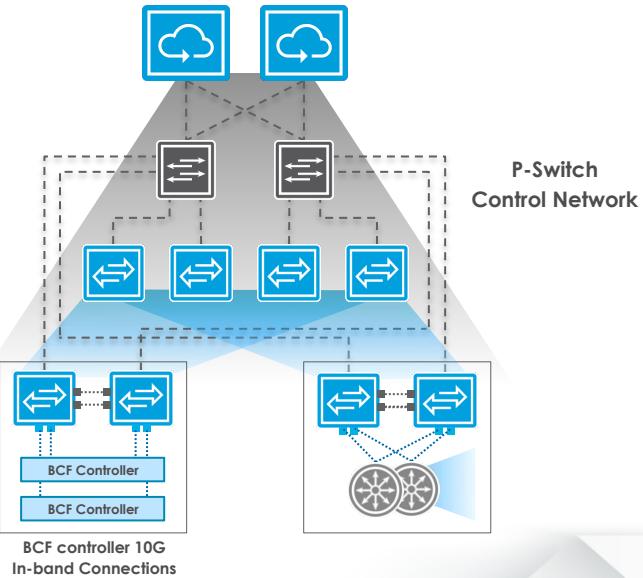
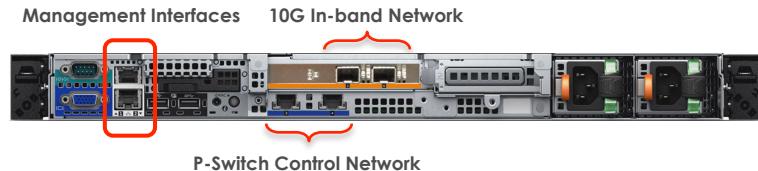
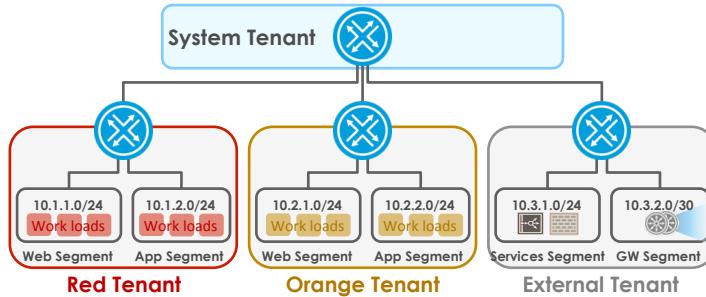
- Peers with tenant logical router
- iBGP / eBGP (multihop)
- Redistribution of connected, static, aggregate routes
- Filter routes using route maps
 - Prefix lists or AS-Path lists
- Graceful restart
- Load balancing using ECMP
- Automatic advertisement of host routes for extended segments
- Multiple BGP peers per logical router

BCF ADDITIONAL CONFIGURATIONS

BGP – BCF In-Band Control Network

BGP & BCF

- BGP process runs on controller
 - Logical router lives on leaf switch
 - BGP packets from controller to the peer traverse the logical router thus requiring multi-hop configuration
 - BGP communicates over the in-band control network
 - In-band interfaces on controller must be connected to leaf switches



BCF ADDITIONAL CONFIGURATIONS

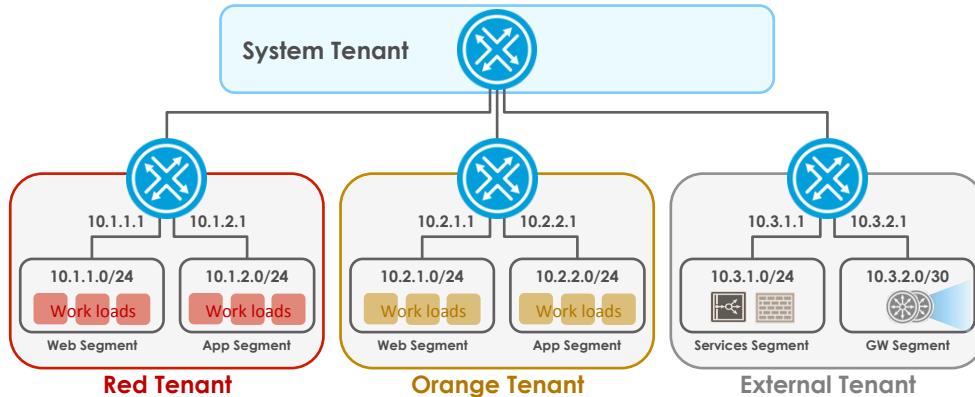
BGP – Configuration Requirements

BGP Configuration

- Local IP address for logical router (protocol IP) to initiate BGP peering
 - /32 address per logical router (tenant)
- Local and remote Autonomous System numbers
- Neighbor's IP address
- Static route and next-hop group (to reach neighbor's IP)
- Multi-hop TTL for eBGP sessions
- BGP gateway shared by multiple tenants
 - Routing configuration within BCF
 - Route distribution
- Route Maps
- Graceful restart

BCF ADDITIONAL CONFIGURATIONS

BGP – Configuration



```
Tenant system
logical-router
  interface tenant Red
  interface tenant Orange
  interface tenant External
    export-route
    route 0.0.0.0/0 next-hop tenant External
```

```
tenant External
logical-router
  interface system
    import-route
  interface GW
    ip address 10.3.2.1

  next-hop-group Router1
    ip 10.3.2.2
  route 10.47.1.0/24 next-hop Router1

bgp
  protocol-ip 10.9.1.1
  local-as 100
  redistribute-connected
  aggregate-address 10.0.0.0/14 summary-only
  graceful-restart

neighbor N1
  description "iBGP to Router 1"
  neighbor-ip 10.47.1.2 remote-as 100
neighbor N2
  description "eBGP to Router 2"
  neighbor-ip 10.3.2.3 remote-as 200
  ebgp-ttl multihop 5
```

BCF ADDITIONAL CONFIGURATIONS

BGP – Route Maps

Route Maps

- Controls the set of routes to be advertised or received
- Modify route attributes
 - Received routes: set local preference
 - Advertised routes: prepend AS-Path
- Applies to each neighbor individually in-bound and/or out-bound
- Route Selection
 - Uses prefix list or AS path list
 - Prefix list selects routes based on CIDR blocks
 - AS path list selects routes based on AS Path using regular expressions

BCF ADDITIONAL CONFIGURATIONS

BGP – Route Maps Configuration

```
tenant External
  logical-router
    prefix-list Martian-Routes
      10 include 10.0.0.0/8
      20 include 172.16.0.0/12
      30 include 192.168.0.0/16
    prefix-list Site-53-Routes
      10 include 204.204.13.0/24
      20 include 173.57.32.0/24
    as-path-list SP1-Routes
      10 exclude "300$"
      20 exclude 4545
      30 include ".*"

    route-map INET-1
      10 permit
        match prefix-list Site-53-Routes
        set local-preference 250
      20 permit
        match as-path-list SP1-Routes
```

```
route-map INET-2
  10 deny
    match prefix-list Martian-Routes
  20 permit

bgp
  protocol-ip 10.9.1.1
  local-as 65500
  redistribute-connected
  aggregate-address 10.0.0.0/14 summary-only

neighbor N2
  description "eBGP to Router 2"
  neighbor-ip 10.3.2.3 remote-as 65511
  ebgp-ttl multihop 5
  apply route-map INET-1 in
  apply route-map INET-2 out
```

BCF ADDITIONAL CONFIGURATIONS

BGP – Status Checks

BGP Status

- Tenants/segments with BGP configuration (BGP segments)

```
controller# show bgp segment
```

- BGP peers

```
controller# show tenant <> bgp summary  
controller# show tenant <> bgp neighbor
```

- BGP routes

```
controller# show tenant <> bgp route  
controller# show tenant <> logical-router route dynamic  
controller# show logical-router route dynamic
```

Questions?

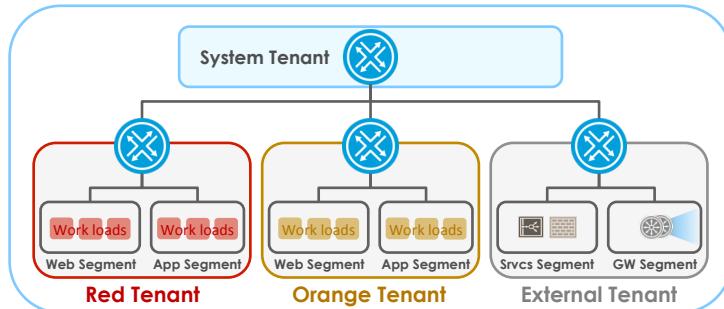
BCF ADDITIONAL CONFIGURATIONS

Inter-POD Connectivity – Outline

- L3 Inter-POD Connectivity
- L2 Inter-POD Connectivity (2 PODs)
- L2 Inter-POD Connectivity (up to 8 PODs)
- L2 Inter-POD Connectivity (VxLAN)
- Extended Segments

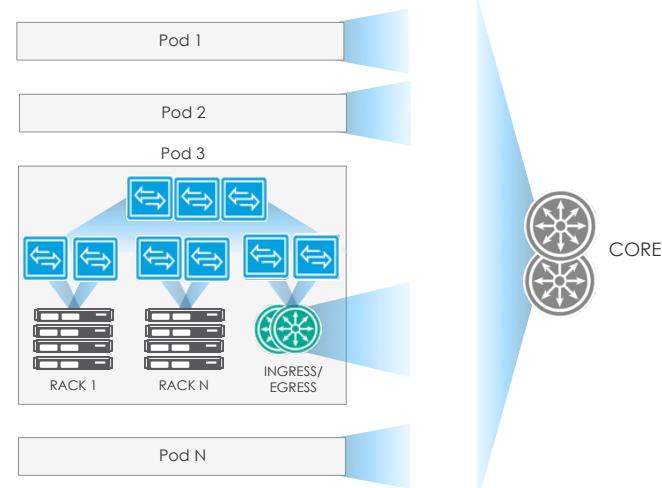
BCF ADDITIONAL CONFIGURATIONS

L3 Inter-POD Connectivity



CORE-AND-POD DESIGN

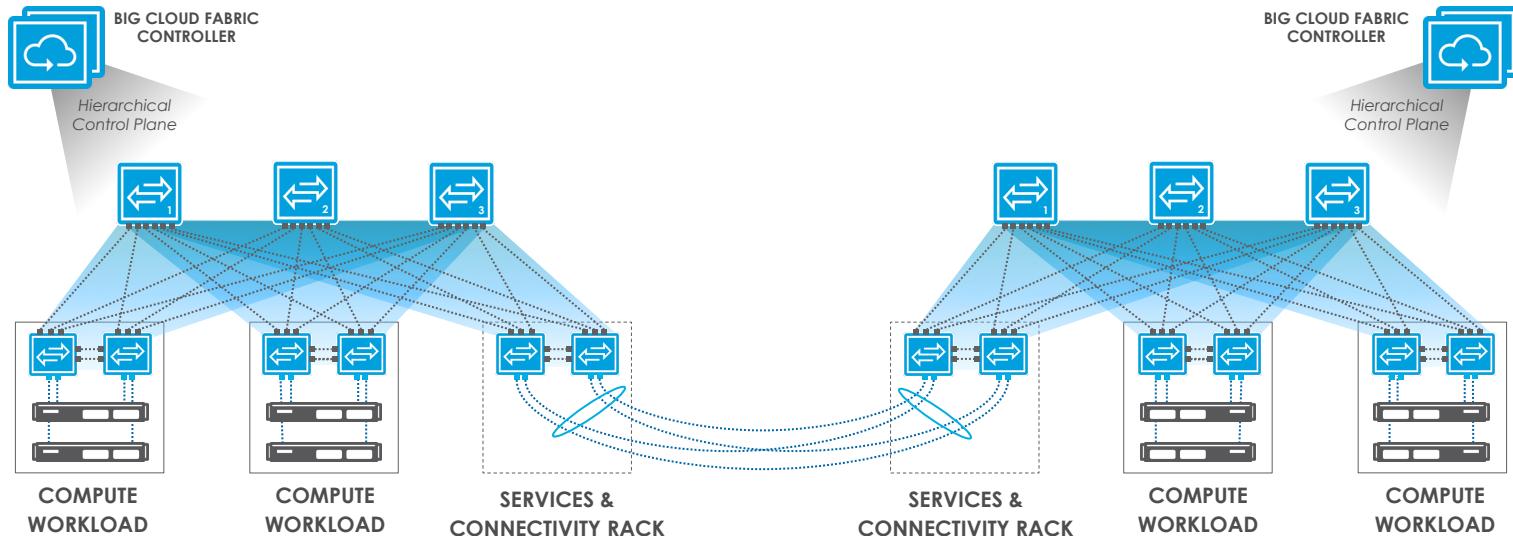
(Hyper-scale Approach)



- ❑ Use one of the external connectivity options
- ❑ GW per BCF; GW per Tenant; GW per Segment

BCF ADDITIONAL CONFIGURATIONS

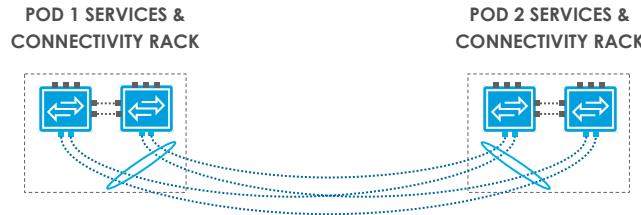
L2 Inter-POD Connectivity



- ❑ Connects two Big Cloud Fabric PODs
- ❑ Uses an interface group; Mode “Inter-pod”

BCF ADDITIONAL CONFIGURATIONS

L2 Inter-POD Connectivity – Configuration



Pod 1 Configuration

```
interface-group pod-2
mode inter-pod
switch R7L1 interface ethernet1
switch R7L1 interface ethernet2
switch R7L2 interface ethernet1
switch R7L2 interface ethernet2

tenant t1
segment s1
member interface-group pod-2 vlan 50
member switch R1L1 interface ethernet10 vlan 50
member switch R1L2 interface ethernet10 vlan 50
```

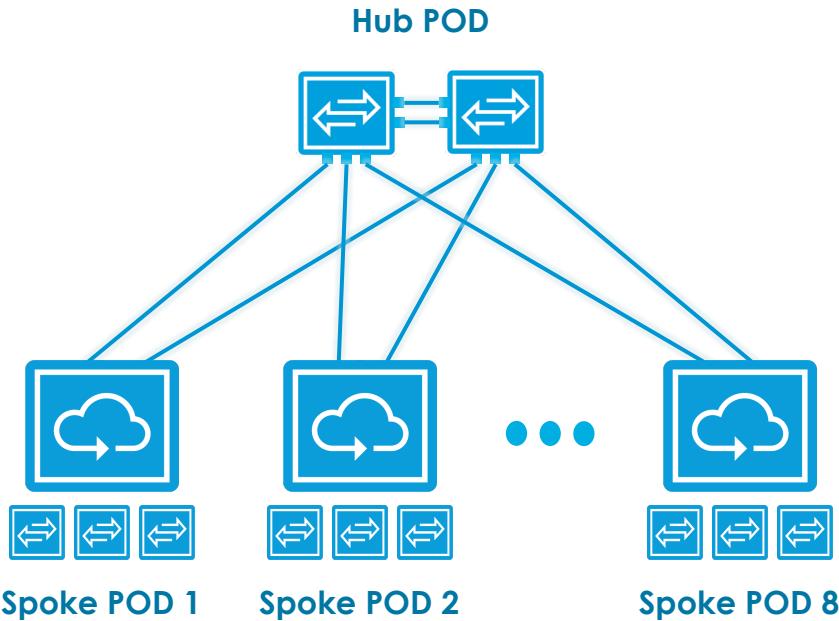
Pod 2 Configuration

```
interface-group pod-1
mode inter-pod
switch R9L1 interface ethernet1
switch R9L1 interface ethernet2
switch R9L2 interface ethernet1
switch R9L2 interface ethernet2

tenant t1
segment s1
member interface-group pod-1 vlan 50
member switch R5L1 interface ethernet23 vlan 50
member switch R5L2 interface ethernet23 vlan 50
```

BCF ADDITIONAL CONFIGURATIONS

L2 Inter-POD Connectivity – Scaling Beyond Two PODs

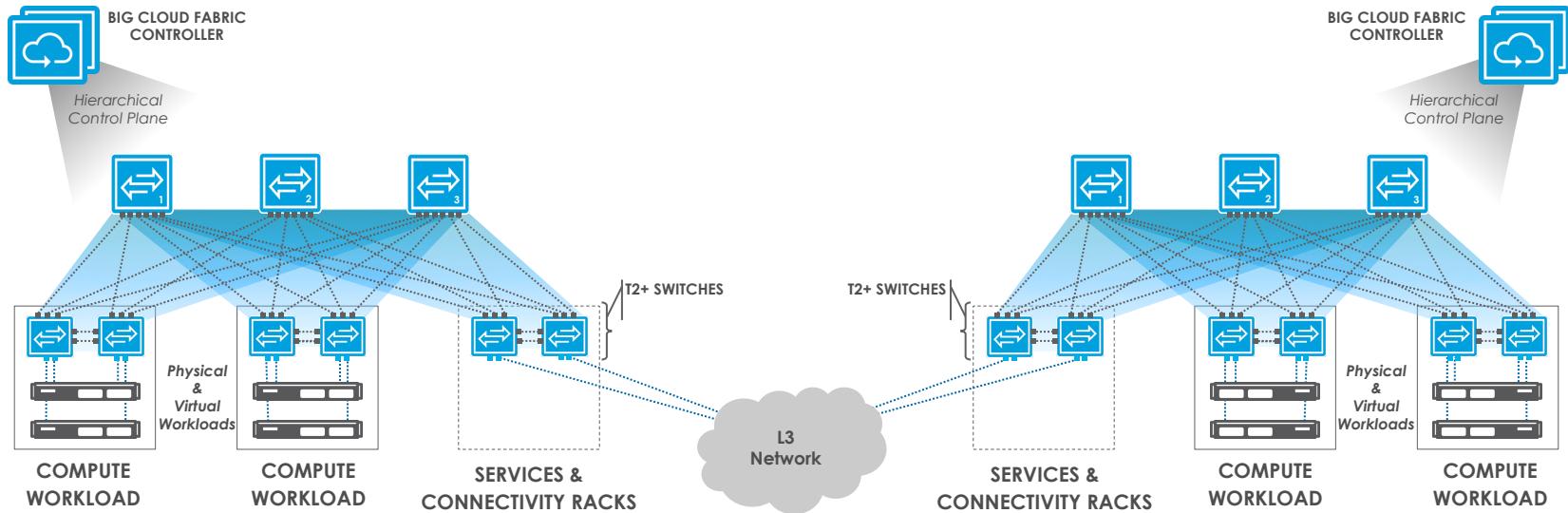


Hub & Spoke BCF PODs

- Extend L2 inter-POD connectivity with up to 8 BCF PODs
- Inter-connects over 500 racks
- HUB POD
 - Two leaf switches
 - One leaf group
 - No Spines
- No L2/L3 protocols required

BCF ADDITIONAL CONFIGURATIONS

L2 Inter-POD Connectivity using VxLAN



- Leaf switches where local and remote VTEPs terminate must be T2+ switches

BCF ADDITIONAL CONFIGURATIONS

L2 Inter-POD Connectivity using VxLAN – Overview

VxLAN

- Extends L2 segments on up to three BCF PODs over a L3 network
- Remote End Points learned using Flood and Learn mechanism
 - No control plane
- BUM traffic handled by local replication

Loop Prevention

- Traffic from remote VTEP only flooded to local ports
- No flooding to remote VTEPs

Supported platforms

- Broadcom Trident II+ family of switches
 - See Module 1 for list of supported switches
- T2+ switches only required where VTEP originates or terminates
 - Spine and other leaf groups not required to be T2+

BCF ADDITIONAL CONFIGURATIONS

L2 Inter-POD Connectivity using VxLAN – Configuration

Configure Remote VTEP

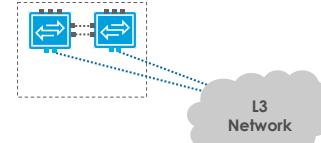
```
Inteface-group To-Rtr
  member switch R3L1 interface ethernet1
  member switch R3L2 interface ethernet2

vxlan-termination
  active
  termination interface-group To-Rtr
  remote-vtep pod2-vtep
  ip 20.20.5.1
```

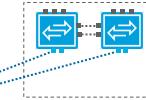
Configure Local VTEP

```
tenant external
  logical-router
    interface segment external-s1
      ip address 10.10.5.1/24
    local-vtep my-vtep
      source interface segment external-s1
      flood vtep pod2-vtep
  segment external-s1
    member interface-group To-Rtr vlan untagged
```

POD 1 SERVICES & CONNECTIVITY RACK



POD 2 SERVICES & CONNECTIVITY RACK



Extend Segment

```
tenant t1
  segment s1
    member vni 1000
    member interface-group ig1 vlan 100
    vxlan-encapsulation
      flood-vtep pod2-vtep
```

BCF ADDITIONAL CONFIGURATIONS

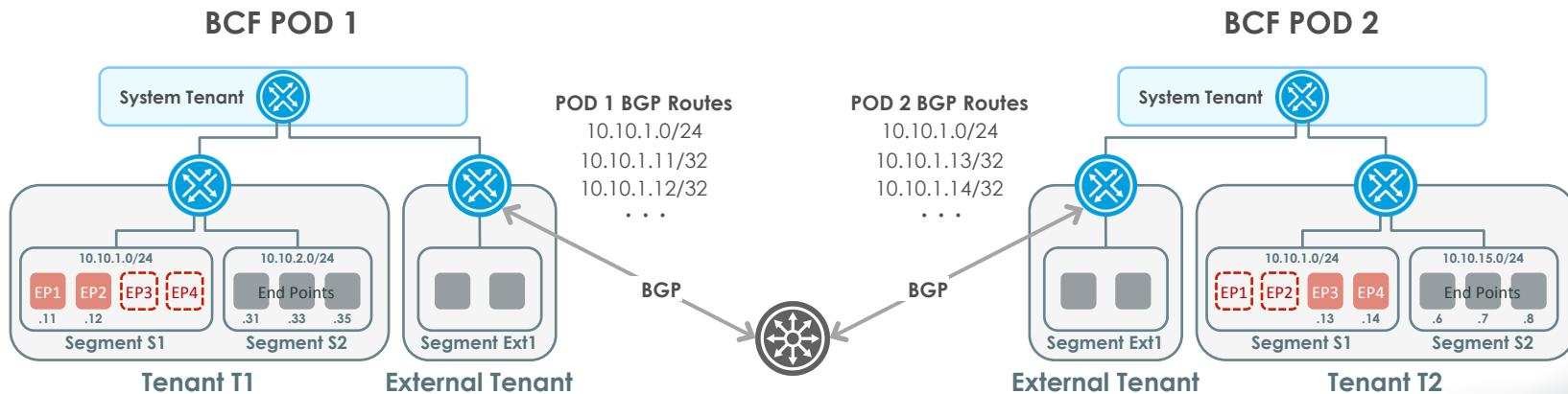
Extended Segments – Traffic Flow

Extended Segment

- Any segment stretched across multiple PODs using inter-pod interface groups or VxLAN

Traffic to segment endpoints (North – South)

- BGP automatically advertises /32 host routes for extended segments
- No extra configuration required



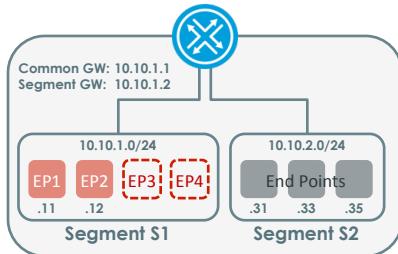
BCF ADDITIONAL CONFIGURATIONS

Extended Segments – Traffic Flow

Traffic to segment gateway (South – North)

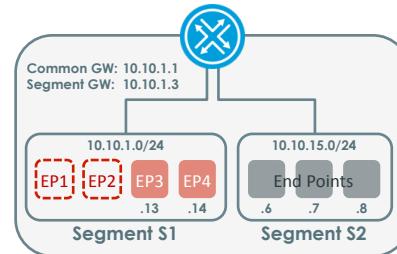
- Configure a local gateway in each POD
- Configure a common gateway using virtual IP on both PODs
- Common gateway uses the same IP and same MAC on both PODs
- No protocols required

BCF POD 1



Tenant T1

BCF POD 2



Tenant T2

POD 1 Configuration

```
tenant T1
logical-router
interface segment S1
ip address 10.1.1.2/24
virtual-ip 10.1.1.1
```

POD 2 Configuration

```
tenant T2
logical-router
interface segment S1
ip address 10.1.1.3/24
virtual-ip 10.1.1.1
```

Questions?

BCF ADDITIONAL CONFIGURATIONS

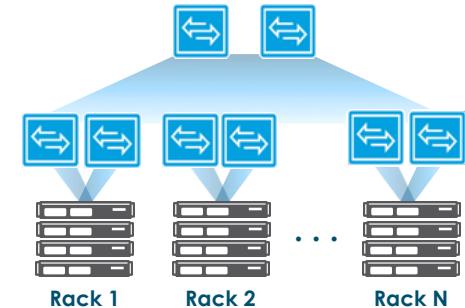
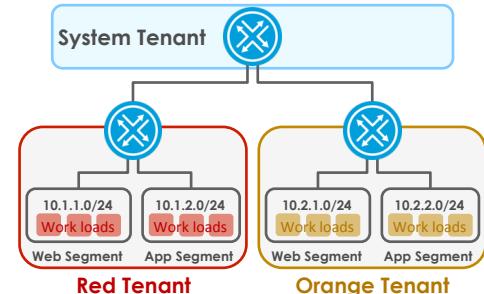
Auto Host Detection

Auto Host Detection

- Connect host to any leaf switch any interface
- Enter interface independent configuration
- Fabric automatically detects host using LLDP
 - Uses “host name” in systemName TLV
 - Uses “interface name” in chassisId or portId TLV
- Creates an interface group of interfaces where LLDP received

```
interface-group CN1
  mode lldp
  member host Compute-Node-1 interface en0
  member host Compute-Node-1 interface en1

tenant Red
  segment App
    member interface-group CN1 vlan untagged
```



BCF ADDITIONAL CONFIGURATIONS

BPDU Guard – Overview

BPDU Guard Default Behavior

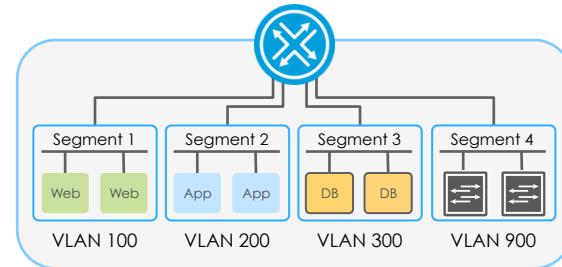
- Enabled by default
- Interface disabled if a BPDU is received
- Requires shut & no shut to clear interface

Disable BPDU Guard

- Applies only to physical leaf switches
- Does not apply to spine switches or virtual switches
- BPDUs are flooded on every interface in the segment

Status

```
controller# show bpdu-guard  
controller# show switch <> interface
```



```
switch leaf1  
  interface ethernet1  
    bpdu-guard-disable
```

```
controller(config)# switch leaf1  
controller(config-switch)# interface ethernet1  
controller(config-switch-int)# shutdown  
controller(config-switch-int)# no shutdown
```

Caveat: Interfaces with BPDU guard disabled are shutdown if there are no peer links on leaf-group switches

BCF ADDITIONAL CONFIGURATIONS

Interoperating with xSTP Enabled Networks

Interoperating with STP, RSTP, MSTP

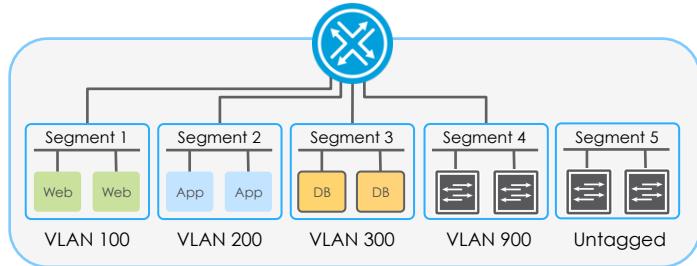
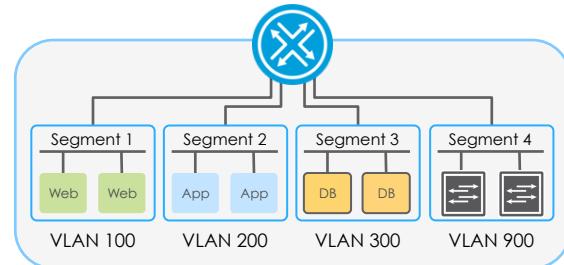
- Disable BPDU guard
- BPDUs received as untagged packets
- Configure a segment to accept untagged traffic if using VLANs

Interoperating with PVSTP, PVSTP+

- BPDUs received on the same VLAN as data traffic
- No additional segments required to flood BPDUs

```
switch leaf-1a
    interface ethernet30
        bpdu-guard-disable

tenant T1
    segment Vlan900
        member interface-group ethernet30 vlan 900
    segment bpdu
        member interface-group ethernet30 vlan untagged
```



BPDU Guard disabled on Segment 4 interfaces and untagged segment configured to allow forwarding of BPDUs

BCF ADDITIONAL CONFIGURATIONS

Storm Control

Storm Control Profiles

- Rate limit storms at ingress on edge ports
- Allows specification of thresholds for 4 traffic types
 - Broadcast, multicast, unknown unicast, and unknown multicast
- Traffic exceeding thresholds is dropped
- Threshold specified as percentage of link speed

Applying Profiles

- Leaf switches
- Leaf switch interfaces
- Leaf switches and Interface

Status

```
controller# show storm-control switch <>
```

```
storm-control-profile scp1
broadcast-rate 10
known-multicast-rate 20
unknown-multicast-rate 20
unknown-unicast-rate 20
```

```
storm-control-profile scp2
broadcast-rate 5
known-multicast-rate 5
unknown-multicast-rate 5
unknown-unicast-rate 5
```

```
switch R1L1
strom-control scp1
interface ethernet10
storm-control scp2
```

BCF ADDITIONAL CONFIGURATIONS

Local SPAN

Local SPAN (Port Mirror)

- Copies traffic from one interface to another interface on same switch
- Maximum of 4 SPAN sessions per switch (local + fabric)

Source

- Spine, leaf, or virtual switch
- Ingress traffic, egress traffic, or ingress and egress traffic
- Filter traffic using match rules (based on switch, interface, traffic type, ...)

Destination

- Any non-fabric interface on same switch
- Multiple interfaces may be configured in a LAG
- Wildcard option allows for easy setup of SPAN on multiple switches
 - any, any-leaf, any-spine

BCF ADDITIONAL CONFIGURATIONS

Fabric SPAN

Fabric SPAN (Port Mirror)

- Copies traffic from any edge interface on any leaf switch
- Forwards traffic to a single fabric destination: an interface group
- Maximum of 3 fabric SPAN sessions per switch
- Maximum of 4 sessions per switch (local + fabric)

Source

- Leaf switch or virtual switch only (no spine)
- Ingress traffic on edge interfaces (no egress traffic; no fabric interfaces)
- Filter traffic using match rules (based on switch, interface, traffic type, ...)

Destination

- A single interface-group configured as “mode span-fabric”
 - May not be individual switch interfaces

BCF ADDITIONAL CONFIGURATIONS

SPAN – Source Filters

SPAN Filters

- Switch
 - Name, any, any-leaf, any-spine
- Interface
 - Name, any, any-edge, any-fabric
- Protocol
 - Src/dst MAC, src/dst IP, src/dst ports (tcp/udp), ...
- Tenants, segments
- Direction
 - Ingress, egress, or both

BCF ADDITIONAL CONFIGURATIONS

SPAN – Configuration

Local SPAN

```
span-local span1
    destination switch any-leaf interface ethernet44
    priority 50
    active
    filter
        10 switch R1L1 interface ethernet5
        20 switch R1L2 dst-ip-cidr 11.0.0.0/8
```

Fabric SPAN

```
interface-group span-int-group1
    mode span-fabric
    member switch R1L1 interface ethernet1
    member switch R1L2 interface ethernet1

span-fabric fspan1
    destination interface-group span-int-group1
    priority 50
    active
    filter
        10 switch any-leaf dst-ip-cidr 99.0.0.0/8
```

BCF ADDITIONAL CONFIGURATIONS

SPAN – Status

SPAN Status

- View configuration

```
controller# show running-config span-local  
controller# show running-config span-fabric
```

- Check local span

```
controller# show span-local  
controller# show span-local <> switch <>
```

- Check fabric span

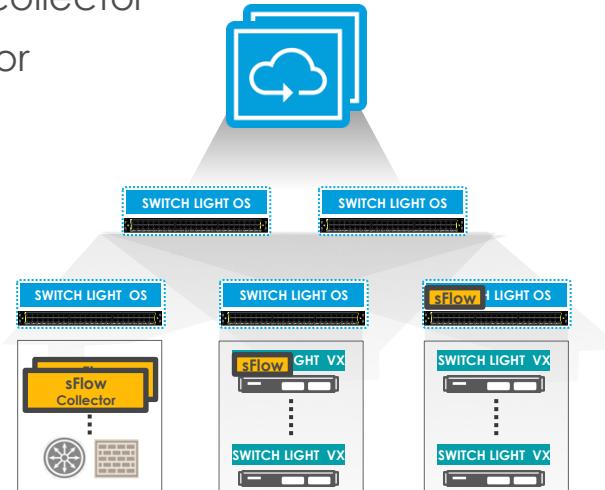
```
controller# show fabric-local  
controller# show fabric-local <> switch <>
```

BCF ADDITIONAL CONFIGURATIONS

sFlow – Overview

sFlow

- Samples packets on leaf switches and forwards to a sflow collector
- Collects interface counters and forwards to a sflow collector
- Works with physical and virtual switches
- Samples only edge interfaces
- Ingress direction only
- Uses BCF data path for forwarding to sflow collector
- Supports up to 4 sFlow collectors
- Shares the same Queue as Fabric SPAN (see QoS)
- Centrally configured but functionally distributed to the switches



BCF ADDITIONAL CONFIGURATIONS

sFlow – Defaults & Configuration

Default Parameters

- UDP port 6343
- Sample packet rate 1 per 10,000
 - Each switch rate limited to 100 pps
- Forwards first 128 bytes of packet
- Counter values sampled every 20 seconds

Configuration

```
sflow
active
collector 10.5.6.5 port 7001 agent-ip tenant t1 interface segment s1
counter-interval 300
```

BCF ADDITIONAL CONFIGURATIONS

sFlow – Status

sFlow Status

- Sflow summary

```
controller# show sflow
```

- Samples taken on switch interfaces

```
controller# show sflow switch <> interfaces
```

- Samples sent to collectors

```
controller# show sflow switch <> collectors
```

- Sampling configuration on switch

```
controller# show forwarding switch <> sflow-sampler-table
```

- Collector info on switch

```
controller# show forwarding switch <> sflow-collector-table
```

Questions?



Module 6 – BCF Troubleshooting

BCF TROUBLESHOOTING

Module Outline

- Troubleshooting Tools & Capabilities
- Troubleshooting Flow Chart
- Starting Points (Checklists)
- **Lab – Troubleshooting BCF**

BCF TROUBLESHOOTING

Troubleshooting Tools & Capabilities

1. Tenant aware ping and traceroute
2. Test path
3. Policy log (packet capture)
4. Counters (policy, data plane, host-bound)
5. Debug commands
6. Device level access & commands
7. Logs
8. Continuous monitor commands
9. SPAN (see “BCF Network Configurations” module)
10. Analytics (see “BCF Analytics” module)

BCF TROUBLESHOOTING

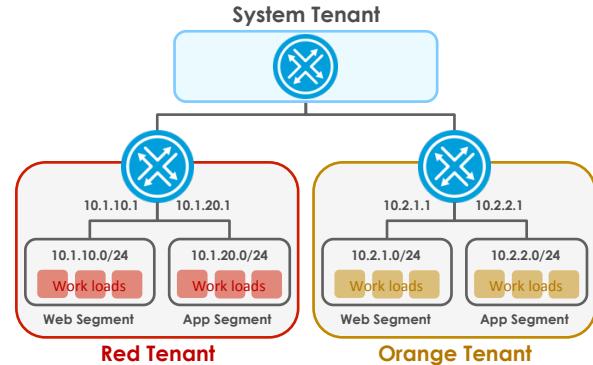
Troubleshooting Tools – Fabric Ping & Traceroute

Ping / Traceroute

- Originates a ping / traceroute from the tenant logical router
- Ping / traceroute a locally attached device from any part of fabric
- Controller originates the packet and places it in data path

```
controller# traceroute 10.1.10.32 src-tenant Red
traceroute to 10.1.10.32, 64 hops max, 64 size packets
1  10.1.10.32  4.484 ms  *  4.471 ms
```

```
controller# traceroute 10.1.10.32 src-tenant Orange
traceroute to 10.1.10.32, 64 hops max, 64 size packets
1  *  *  *
2  10.1.10.1  4.679 ms  *  6.638 ms
3  10.1.10.32  4.265 ms  *  4.830 ms
```



BCF TROUBLESHOOTING

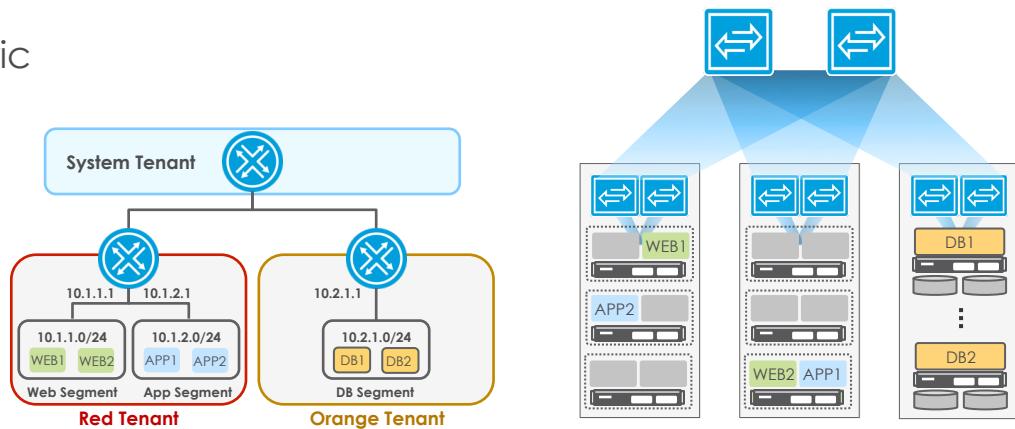
Troubleshooting Tools – Test Path

Tracing path through fabric

- How do you trace the path of a packet from one end-point to another inside the fabric?
- E.G. Web1 (10.1.1.20) to DB2 (10.2.1.35)

Test path

- Identifies the path inside the fabric
- Logical Path
- Physical Path
- Actual Path



BCF TROUBLESHOOTING

Troubleshooting Tools – Test Path

Two Modes

- Controller-view (GUI: Simulate)
 - Determine path from controller point-of-view
 - Can detect configuration and state issues
 - Logical router and leaf-group level granularity
- Fabric-view (GUI: Test)
 - Determine exact path traffic takes through the fabric
 - Can detect physical link issues or switch programming issues
 - Switch and interface level granularity



```
controller# test path src-tenant Red src-ip 10.1.1.20 dst-ip 10.2.1.35 controller-view
controller# test path src-tenant Red src-ip 10.1.1.20 dst-ip 10.2.1.35 fabric-view test-name test-A
```

BCF TROUBLESHOOTING

Troubleshooting Tools – Test Path

Controller-view

- Source endpoint must be learned before using test path

```
controller2# test path src-tenant Red src-ip 10.1.1.99 dst-ip 10.2.1.99 controller-view

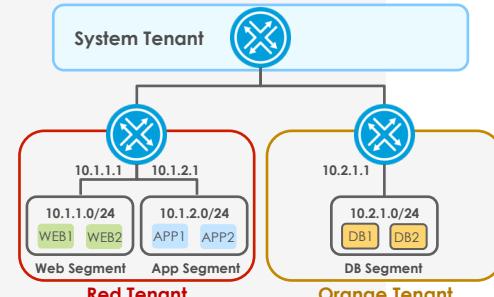
~ Logical path ~
None.

~ Physical path ~
None.

~ Reverse logical path ~
None.

~ Reverse physical path ~
None.

Forward Result      : invalid input
Logical Simulation Error : unknown source endpoint
Reverse Result       : unspecified source
```



Example output for a non-existent source endpoint. Shows sections in which information is provided.

BCF TROUBLESHOOTING

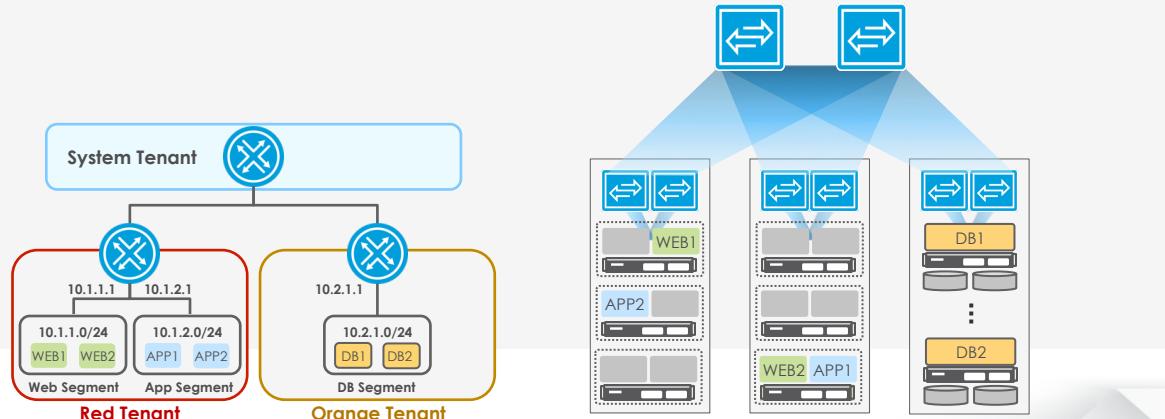
Troubleshooting Tools – Test Path

```
controller(config)# test path src-tenant Red src-ip 10.1.1.20 dst-ip 10.2.1.35 controller-view
```

Logical-paths of Controller-views			
Hop	Index	Hop	Policy
1		10.1.1.20	tenant Red segment Web
2		logical-router Red	99 permit any to any
3		logical-router system	10 permit tenant Red to tenant Orange
4		logical-router Orange	default permit
5		10.2.1.35	tenant Orange segment DB

~ Physical-paths of Controller-views ~

Path	Hop	Index	Hop
1	1		10.1.1.20
1	2		R1
1	3		spine
1	4		R3
1	5		10.2.1.35



BCF TROUBLESHOOTING

Troubleshooting Tools – Test Path

Fabric view

1. Create a test path to program TCAM entries in each switch
2. Send traffic from source to destination
3. Use show command to view results

```
controller# test path src-tenant Red src-ip 10.1.1.20 dst-ip 10.2.1.35 fabric-view test-name test-A

----- Logical-paths of Setup-results -----
Hop          Policy           Route
-----|-----|-----|
10.1.1.20 tenant Red segment Web
logical-router Red      99 permit any to any      static route 0.0.0.0/0
logical-router system   10 permit tenant Red to tenant Orange Tenant Iface Orange
logical-router Orange   default permit        Segment Iface DB
10.2.1.35 tenant Orange segment DB

Forward Result : forwarded
Message       : Setup succeeded. Inject icmp traffic and use "show test path test-A" to observe path.
```

BCF TROUBLESHOOTING

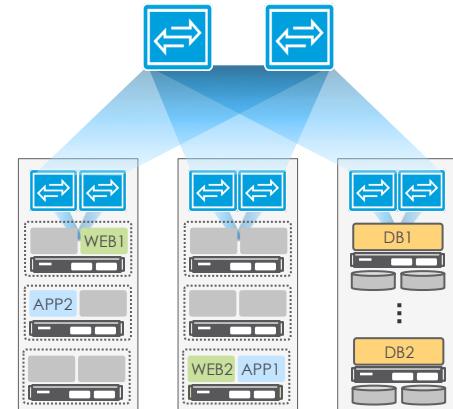
Troubleshooting Tools – Test Path

```
controller# show test path test-A
```

Hop	Index	Hop	Ingress Interface	Egress Interface	Type	TCAM Counter	Test Status
0		R1L1	ethernet7	ethernet50	spine	5	timeout
1		S1	ethernet1	ethernet4	leaf	5	timeout
2		R3L2	ethernet2	IG-DB2	edge	5	timeout

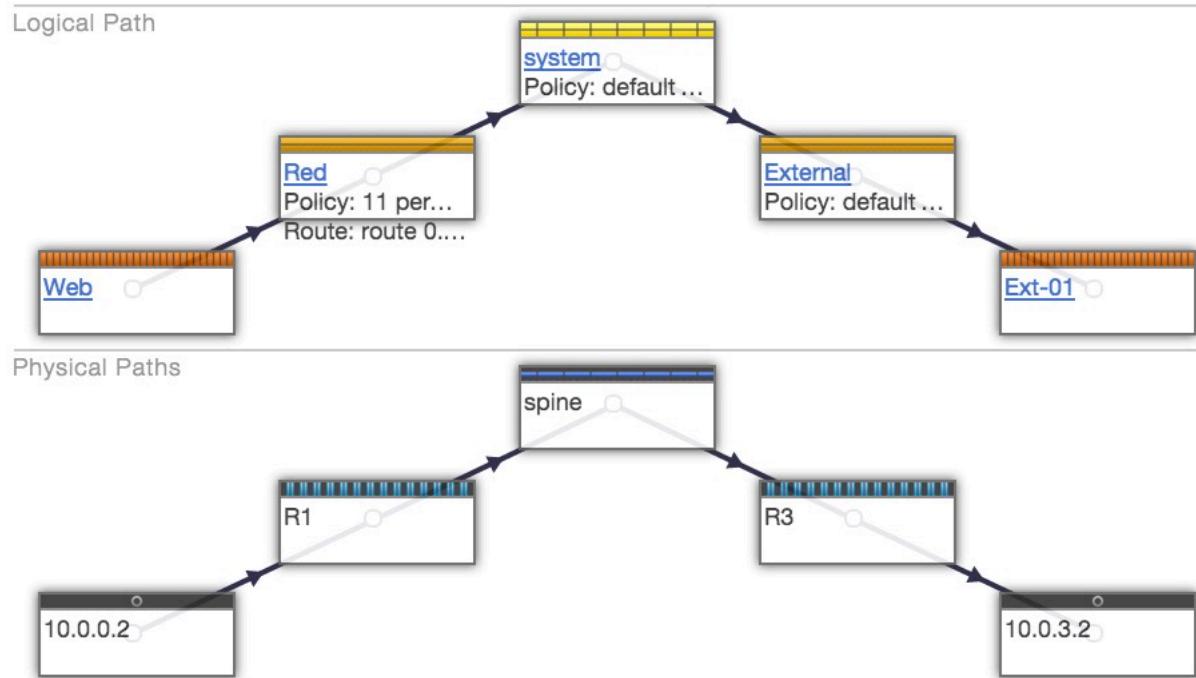
Fabric view

- Exact interfaces used by test traffic
 - Granularity extends beyond “leaf group” and “spine”
- Exact number of hits to the test path TCAM entries
- TCAM entries removed when timeout expires
- Default timeout 30 seconds



BCF TROUBLESHOOTING

Troubleshooting Tools – Test Path – GUI Example

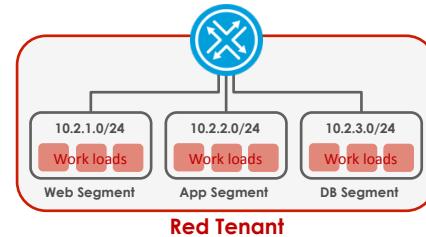


BCF TROUBLESHOOTING

Troubleshooting Tools – Policy Log

Policy Configuration

```
tenant Red
  logical-router
    policy-list cap-host1
      10 permit any to 10.2.24.101/32 log
      99 permit any to any
    apply policy-list cap-host1
```



Policy Log (Packet Capture)

#	Tenant	Name	Interface	Src IP	Dst IP	IP Protocol	Src Port	Dst Port	Byte	Pkt	First Seen	Last Seen
1	Red	tenant	system	10.1.7.5	10.2.24.101	1			204	2	2016-04-09 13:37:56.616000 PDT	2016-04-09 13:38:21.557000 PDT
2	Red	tenant	system	10.1.13.160	10.2.24.101	1			612	6	2016-04-09 13:38:50.466000 PDT	2016-04-09 13:39:15.542000 PDT
3	Red	tenant	system	10.1.5.200	10.2.24.101	17	53	40068	112	1	2016-04-09 14:05:05.997000 PDT	2016-04-09 14:05:05.997000 PDT
4	Red	tenant	system	10.1.5.200	10.2.24.101	17	53	52191	112	1	2016-04-09 14:05:09.120000 PDT	2016-04-09 14:05:09.120000 PDT
... <snip> ...												
172	Red	tenant	system	91.189.88.152	10.2.24.101	6	80	37861	404	3	2016-04-10 07:49:46.095000 PDT	2016-04-10 07:49:56.523000 PDT
173	Red	tenant	system	79.133.49.138	10.2.24.101	6	80	40732	148	2	2016-04-10 07:49:47.153000 PDT	2016-04-10 07:49:56.669000 PDT
174	Red	tenant	system	216.58.219.46	10.2.24.101	6	80	41105	148	2	2016-04-10 07:49:46.956000 PDT	2016-04-10 07:49:56.966000 PDT

BCF TROUBLESHOOTING

Troubleshooting Tools – Policy Log

Policy Log

- Matches traffic in data path based on criteria
- Saves matching packets in a circular buffer (buffer size 5K)
- Configure logical router policy as explained in policy section
 - Add “log” keyword to each policy statement to be logged
 - Limit match to /32 or as small a range as possible to reduce packets captured
 - Policy exits on first match
 - If no existing policy in logical router
 - ❖ add “permit any to any” as last statement before applying policy
- Add policy statement in each logical router where logging required
- Rate limited to 1 packet every 5 seconds per flow per switch
- Can modify existing policy in effect by adding new statements

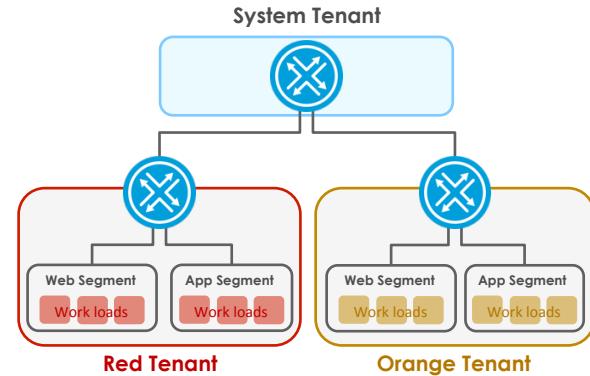
BCF TROUBLESHOOTING

Troubleshooting Tools – Policy Counter

Policy Configuration

```
tenant Red
  logical-router
    policy-list p1
      10 permit segment-interface Web any to tenant Red segment App
      20 permit segment-interface Web any to tenant Orange
      99 permit any to any
    apply policy-list p1
```

Policy Counter



#	Tenant	Name	Rule	Byte	Pkt
1	Red		10 permit segment-interface Web any to tenant Red segment App	490	5
2	Red		20 permit segment-interface Web any to tenant Orange	490	5
3	Red		99 permit any to any	52984	78

BCF TROUBLESHOOTING

Troubleshooting Tools – Counters

Data Plane Counters

- Interface Counters
 - Unicast, Multicast, Broadcast, Errors, Drops, Incorrect VLAN tag

```
controller# show switch <> interface <> counters
controller# show switch <> interface <> counters errors
```

- Tenant and segment counters

```
controller# show tenant <> counters
controller# show segment <> counters
```

Host Bound Counters (switch & controller bound)

- ARP, LACP, IGMP, ICMP, PIMU (host bound)

```
controller# show switch <> agent-counters {arp|lacp|igmp|...}
controller# show switch <> pimu-counters
```

BCF TROUBLESHOOTING

Troubleshooting Capabilities – Debug Commands

Debug Commands

- Set of useful debugging commands grouped together in one area

Some Useful Debug Commands

- List of all attachment points
- List of debug counters
- List of various events
- Various multicast debugging information
- Upgrade status
- HCL compatibility of switch optics
- Endpoint manager incomplete state

```
controller# show debug ...
controller# show debug attachment-points
controller# show debug counters ...
controller# show debug event all
controller# show debug event module <> ...
controller# show debug igmp-manager ...
controller# show debug mcast-manager ...
controller# show debug upgrade status
controller# show debug switch <> inventory
controller# show debug endpoint-manager incomplete
```

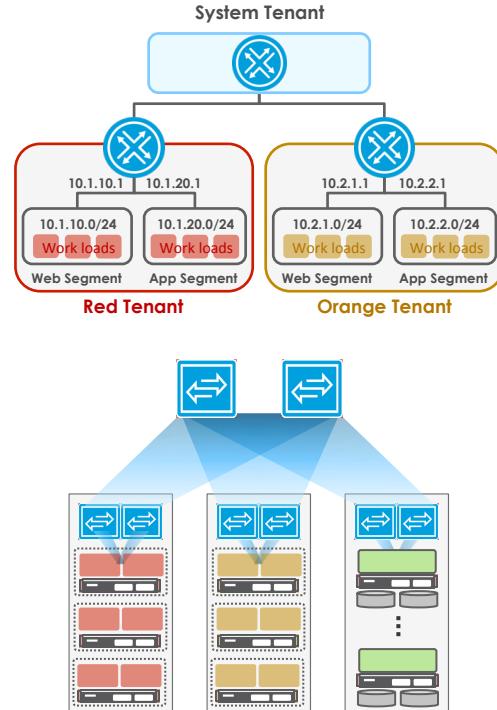
BCF TROUBLESHOOTING

Troubleshooting Capabilities – Debug Commands – Example

Incomplete L2 Configuration / State

- Tenants without segments
- Segments without members
- Configured but inactive endpoints
- **Inactive membership rules**
 - Membership rule configured but interface is down
 - Pre-provisioned or typo?
- **Unknown tracked endpoints**
 - Next hop with unknown IP address
- **Unclassified attachment points**
 - Active ports that are not configured as part of any segment

```
controller# show debug endpoint-manager incomplete
```



BCF TROUBLESHOOTING

Troubleshooting Capabilities – Debug Commands – Examples

Fabric Interface Flap Events

```
controller# show debug event module FabricManager event-name  
          fabric-interface-physical-status-change-event last 20 events
```

Switch Connect/Disconnect Events

```
controller# show debug event module FabricManager event-name  
          fabric-switch-connect-disconnect-event last 20 events
```

BCF TROUBLESHOOTING

Troubleshooting Capabilities – Device Level Access & Commands

Accessing Controller (Beyond CLI)

- Get access to underlying “Bash” shell and Linux utilities
- May use standard scp command to push or pull files
- Very few, if any, operations require Bash access
- Access to controller outside the CLI environment should typically be done in conjunction with BSN Support

```
controller# debug bash
```

Accessing Switches

- Switch level access provided via two means:
 - In-band connection to switch
 - Switch specific CLI commands
- View forwarding state or tables in a given switch

```
controller# connect switch <>
```

```
controller# show forwarding switch <> ...
```

BCF TROUBLESHOOTING

Troubleshooting Tools – Logs

BCF Logs

- Controller & switch logs
- BGP logs
- Upgrade logs
- Audit logs
 - CLI history
- Controller appliance logs
- OpenStack integration logs
- vCenter integration logs

```
controller# show logging controller
controller# show logging switch <>

controller# show logging routing

controller# debug bash
bash$ more /var/lib/floodlight/upgrade/...

controller# show logging audit
controller# history

controller# show logging syslog

controller# show logging networkservices

controller# debug bash
bash$ more /var/log/vsphere-extension.log
```

BCF TROUBLESHOOTING

Troubleshooting Capabilities – Monitoring Commands

CLI Commands

- Watch (runs any CLI command every two seconds)

```
controller# watch show switch R1L1 interface all counters | grep -vP "( +0){3}"  
controller# watch show switch R3L2 agent-counters lacpa
```

Logs

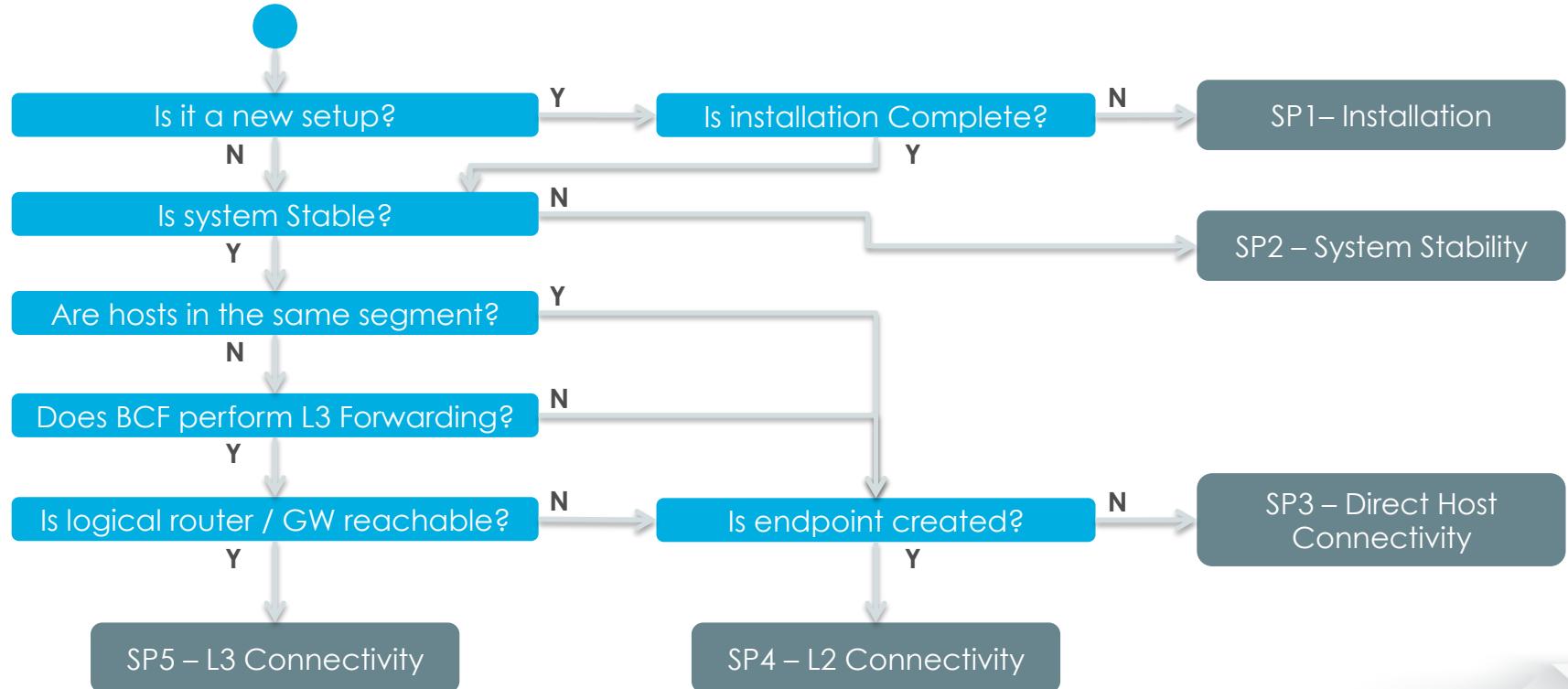
- Display logs continuously when keyword “forever” used
- Same as “tail -f <filename>” for Linux users
- Control-c (^C) to exit

```
controller# show logging controller last 5 minute forever
```

Questions?

BCF TROUBLESHOOTING

Troubleshooting Flow Chart – Identify Starting Point (SP)



BCF TROUBLESHOOTING

SP1 – System Installation Checklist

SP1– Installation

- Check controller status
- Check switch status
- Check fabric status
- Check fabric links
- Check endpoints
- Check routing

BCF TROUBLESHOOTING

SP1 – System Installation (1)

Controller status

- Verify controllers are in expected active/standby state
- If state is unexpected
 - Check out-of-band 1G management connections
 - Check IP addresses
 - Check network connectivity

```
controller# show controller
Cluster Name      : TLC
Cluster Description : Training Lab Cluster
Cluster Virtual IP   : 10.2.24.130
Redundancy Status    : redundant
Last Role Change Time : 2016-04-05 05:50:37.857000 UTC
Failover Reason     : Changed connection state: disconnected from node 32767
Cluster Uptime      : 5 days, 9 hours
# IP          @ State  Uptime
-|-----|-|-----|-----|
1 10.2.24.132 * active  5 days, 10 hours
2 10.2.24.131   standby 3 days, 11 hours
```

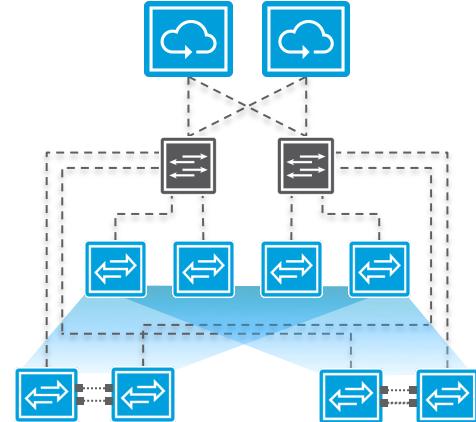
BCF TROUBLESHOOTING

SP1 – System Installation (2)

Switch status

- Verify all switches are in connected state
- If switch is not connected
 - Check connectivity to controller (P-Switch Control Network)
 - Check configured MAC address for typos
 - Check for another OS on switch (remove OS if one exists)
 - Check switch is powering up
 - Check controller logs for discovery messages
- Verify switch SW/FW versions: ONIE, CPLD, SWL
- Verify switches have correct roles
- Verify that all optics connected to switches are supported (see HCL)

```
controller# show switch
controller# show switch all version details
controller# show debug switch <name> inventory
controller# show logging controller | grep <MAC>
```



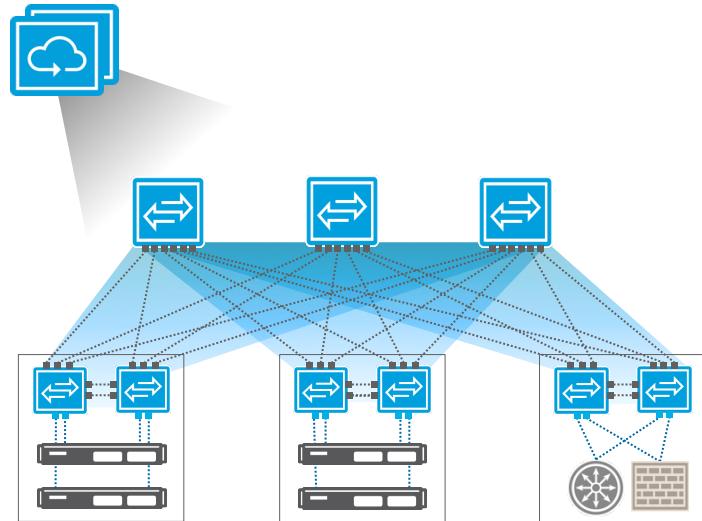
BCF TROUBLESHOOTING

SP1 – System Installation (3)

Fabric status

- Verify that there are no fabric errors
- Verify that all warnings are expected
- If all links are not connected
 - Check interface connections
 - Check optics or cables are supported (see HCL)
 - Perform basic interface troubleshooting

```
controller# show fabric
controller# show fabric errors
controller# show fabric warnings
controller# show debug switch <name> inventory
```



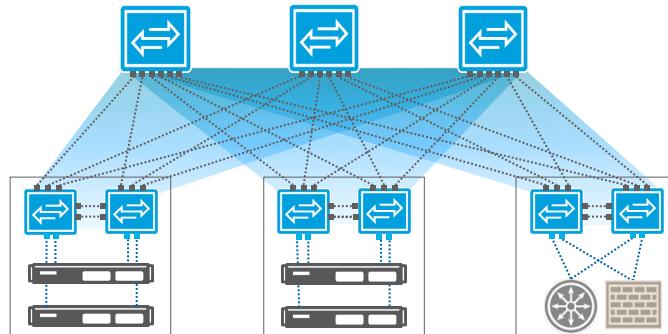
BCF TROUBLESHOOTING

SP1 – System Installation (4)

Fabric links

- Verify all fabric links have been discovered
- If all links are not discovered, check interface connections

```
controller# show link
#  Switch Name IF Name      Switch Name IF Name      Link Type
---|-----|-----|-----|-----|-----|
1  leaf1a     ethernet48  leaf1b     ethernet47  peer
2  leaf2b     ethernet48  leaf2a     ethernet47  peer
3  spine1     ethernet1   leaf1a    ethernet49   leaf-spine
4  spine1     ethernet3   leaf1b    ethernet51   leaf-spine
5  spine1     ethernet5   leaf2a    ethernet49   leaf-spine
6  spine1     ethernet7   leaf2b    ethernet49   leaf-spine
7  spine2     ethernet1   leaf1a    ethernet51   leaf-spine
8  spine2     ethernet3   leaf1b    ethernet49   leaf-spine
9  spine2     ethernet5   leaf2a    ethernet51   leaf-spine
10 spine2    ethernet7   leaf2b    ethernet51   leaf-spine
```



```
controller# show debug event module FabricManager last 20 events
```

BCF TROUBLESHOOTING

SP1 – System Installation (5)

Endpoints

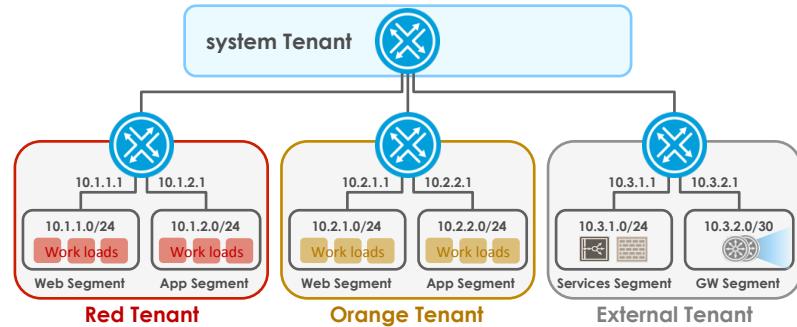
- Verify endpoints have been learned

```
controller# show endpoints
```

Routing

- Verify segments have IP addresses
- Verify static default routes where required
- Verify routing to and from system tenant
- Verify all routes are as expected

```
controller# show logical-router
controller# show logical-router interface
controller# show logical-router route
```



BCF TROUBLESHOOTING

SP1 – System Installation – Command Summary

```
controller# show controller

controller# show switch all details
controller# show switch all version details
controller# show debug switch <name> inventory
controller# show logging controller | grep <MAC>

controller# show fabric
controller# show fabric errors
controller# show fabric warnings
controller# show debug switch <name> inventory

controller# show link
controller# show debug event module FabricManager last 20 events

controller# show endpoint

controller# show logical-router
controller# show logical-router interface
controller# show logical-router route
```

BCF TROUBLESHOOTING

SP2 – System Stability Checklist

SP2 – System Stability

- Check controller/switch/fabric status (See SP1)
- Check controller health (CPU, memory)
- Check switch and fabric link stability
- Check host bound traffic rate
- Check broadcast storm
- Check endpoint stability
- Check routing stability (if applicable)

BCF TROUBLESHOOTING

SP2 – System Stability (1)

SP1 Checklist

- Run all SP1 checklist commands multiple times
- Check for unexpected state changes

CPU & Memory Utilization

```
controller# show controller localhost
```

Flaps

- Check switch connection flaps

```
controller# show debug event module FabricManager event-name  
fabric-switch-connect-disconnect-event last 20 events
```

- Check fabric link flaps

```
controller# show debug event module FabricManager event-name  
fabric-interface-physical-status-change-event last 20 events
```

BCF TROUBLESHOOTING

SP2 – System Stability (2)

Traffic Rates

- Check for excessive host bound traffic

```
controller# clear switch all pimu-counters
controller# show switch all pimu-counters | grep -vP "( +0){5}"
```

- Check for excessive broadcast traffic

```
controller# clear switch all interface all counters
controller# show switch all interface all counters incoming | sort -s 4n -k 7 | awk "$7 != 0"
controller# show switch all interface all counters outgoing | sort -s 4n -k 7 | awk "$7 != 0"
```

Log Activity

- Stable and non-dynamic system has little to no log activity

```
controller# show logging controller last 5 min
controller# show logging syslog last 5 min
```

BCF TROUBLESHOOTING

SP2 – System Stability – Command Summary

***** All commands from "SP1 – Installation" and the following commands *****

```
controller# show controller localhost

controller# show debug event module FabricManager event-name
            fabric-switch-connect-disconnect-event last 20 events
controller# show debug event module FabricManager event-name
            fabric-interface-physical-status-change-event last 20 events

controller# clear switch all pimu-counters
controller# show switch all pimu-counters | grep -vP "( +0){5}"

controller# clear switch all interface all counters
controller# show switch all interface all counters incoming | sort -s 4n -k 7 | awk "$7 != 0"
controller# show switch all interface all counters outgoing | sort -s 4n -k 7 | awk "$7 != 0"

controller# show logging controller last 5mins
controller# show logging syslog last 5mins
```

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity Checklist

SP3 – Direct Host Connectivity

Endpoint not created

- Check interface
- Check interface-group / bond configuration
- Check for VLAN mismatch
- Check segment rule configuration

Endpoint created but IP address not learned

- Check segment interface
- Clear ARP cache

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity – Endpoint Not Created (1)

Interface status

- Check expected interface is in “up” state
- Other possible states
 - Down because its not connected
 - Down because its “admin shutdown”
 - Down because its disabled after receiving a BPDU
 - Down because of unsupported optics or DAC
- Check interface is connected to expected host
 - Remove link, if possible, to determine that interface also goes down

```
controller# show switch <> interface <>
controller# show debug switch <> inventory
controller(config)# switch <> ; interface <> ; shut ; no shut
```

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity – Endpoint Not Created (2)

Misconfigured interface-group / bond

- Check interface-group members are correct (typos)
- Check member settings are correct on host
- Check mode settings are same on both sides (static or LACP)
- Check LACP TX / RX packets on expected switch interfaces
- Check LACP TX / RX packets on remote host using tcpdump or other tools

```
controller# show interface-group all members
# Name      Mode Switch Name Interface Name Leaf Group IF State IF Down Reason
-|-----|----|-----|-----|-----|-----|-----|-----|-----|-----|
1 H5        lacp R2L1      ethernet1    R2       up     None
2 H5        lacp R2L2      ethernet1    R2       up     None
3 compute-1 lacp R1L1      ethernet15   R1       down   LACP Convergence Timeout
4 compute-1 lacp R1L2      ethernet15   R1       down   LACP Convergence Timeout
```

```
controller# show switch <> agent-counters lacpa description
```

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity – Endpoint Not Created (3)

VLAN configuration mismatch

- Check for packets received on un-configured VLAN
- Check host side / remote side VLAN configuration

```
controller# clear switch all interface all counters

controller# show switch all interface all counters errors | grep -vP "( +0){3}"
~~~~~ Interface Error Counters ~~~~~
#  Switch Name IF Name      Rx Bad Vlan Pkt Rx Error Tx Error
--|-----|-----|-----|-----|-----|
2  R1L1      ethernet3  12          0          0
4  R1L1      ethernet17 137030       0          0
6  R1L1      ethernet39 127         0          0
8  R1L1      ethernet43 96          0          0
16 R1L2      ethernet11 7           0          0
47 R3L1      ethernet11 158300       0          0
```

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity – Endpoint Not Created (4)

Misconfigured segment rule

- Check “Inactive Membership Rules”
- Means that there is a misconfiguration or interface is down
- Check switch and interface names are correct (typos)
- Check interface-group names are correct (typos)

```
controller# show debug endpoint-manager incomplete member-rule
-----Inactive Membership Rules -----
# Tenant    Segment   Switch Interface      Interface group VLAN State
-|-----|-----|-----|-----|-----|-----|-----|
 1 t1        s1       P1L1    ethernet1          100  inactive
 2 t1        s3       R1L1    etherenet33        untagged  inactive
 3 t1        s2       R1L1    ethernet27          123  inactive
```

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity – IP Address Not Learned

Segment interface

- Check segment interface has been configured on logical router
- Check segment interface has an IP address

```
controller# show tenant <> logical-router interface
```

ARP cache

- Host
 - Clear ARP on host and ping to a different host
 - Perform tcpdump on host to verify correct egress interface and VLAN
- Switch
 - Check ARP packet counters

```
controller# show switch <> agent-counters arpa description
```

BCF TROUBLESHOOTING

SP3 – Direct Host Connectivity – Command Summary

```
controller# show switch <> interface <>
controller# show debug switch <> inventory
controller(config)# switch <> ; interface <> ; shut ; no shut

controller# show interface-group all members

controller# show switch <> agent-counters lacpa description

controller# clear switch all interface all counters
controller# show switch all interface all counters errors | grep -vP "( +0){3}"

controller# show debug endpoint-manager incomplete member-rule

controller# show tenant <> logical-router interface

controller# show switch <> agent-counters arpa description
```

BCF TROUBLESHOOTING

SP4 – L2 Connectivity Issue

SP4 – L2 Connectivity

Possible Reasons

- Duplicate IP
- Incorrect segment configuration
- Incomplete installation
- Unstable Fabric links or general system instability

Starting Point

```
controller# test path src-tenant <> src-ip <> dst-ip <> controller-view  
controller# test path src-tenant <> src-ip <> dst-ip <> fabric-view test-name <>
```

BCF TROUBLESHOOTING

SP5 – L3 Connectivity Issue

SP5 – L3 Connectivity

Possible Reasons

- Missing interface to and from system tenant
- Default route to system router
- Inter-tenant route distribution
- Misconfigured policy
- External router configuration or connectivity issues

Starting Point

```
controller# test path src-tenant <> src-ip <> dst-ip <> controller-view  
controller# test path src-tenant <> src-ip <> dst-ip <> fabric-view test-name <>
```

Questions?



Lab – Troubleshooting BCF

LAB OBJECTIVES

- Determine how to get fabric status
- Monitor traffic utilization of various fabric elements
- Use policy counters for troubleshooting
- Use packet capture for troubleshooting
- Trace path of packets through the fabric
- Monitor logs

LAB – TROUBLESHOOTING BCF – PAGE (1)

1. Load configuration for this lab

1. copy snapshot://BCF-Config4 running-config

2. Get fabric status

1. Check controllers: show controller
2. Check switches: show switch
3. Check fabric: show fabric
4. Check errors/warnings if any: show fabric error ; show fabric warning
5. Check links: show link
6. Check for switch flaps: show debug event module FabricManager event-name fabric-switch-connect-disconnect-event last 20 events
7. Check for fabric interface flaps: show debug event module FabricManager event-name fabric-interface-physical-status-change-event last 20 events
8. If there are no interface change events, shutdown switch S1 and check again
9. If S1 was shutdown, perform no shut and check again.
10. Verify that there are no fabric errors before proceeding

LAB – TROUBLESHOOTING BCF – PAGE (2)

1. Send traffic & create endpoints

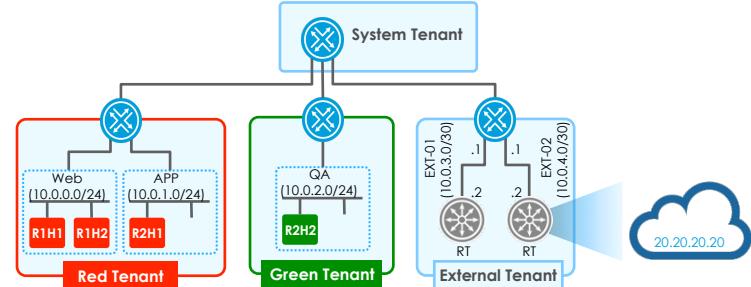
1. Open a CLI session on host R1H1
2. ping r2h1. ping r2h2. ping world

2. Check endpoints

1. show endpoint
2. show tenant Red endpoint; show tenant Green endpoint
3. R1H1, R2H1, and R2H2 endpoints should be learned in their respective tenants
4. Determine how long the endpoints have been active: show endpoint details

3. Prepare to send large volume of traffic from R1H1

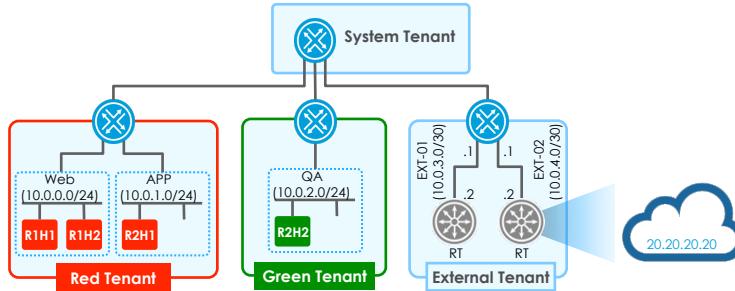
1. First clear counters
2. clear switch all interface all counters
3. clear tenant all counters ; clear segment all counters
4. R1H1> bin/send-traffic
5. 100 packets of size 1000 bytes will be sent to R2H1, R2H2, and World
6. Ping will take about 20 seconds



LAB – TROUBLESHOOTING BCF – PAGE (3)

1. Check utilization

1. On switches: show switch all interface all counters outgoing
2. On all R2 Leaf Group switches: show switch all interface all ... | grep R2L
3. On tenants: show tenant all counters
4. On segments: show segment all counters
5. Tenant and Segment packet counts: Red tenant tx/rx 300 (Web) + 100 (App). Green tx/rx 100. External tx/rx 100
6. Continuously monitor interface utilization: watch ...



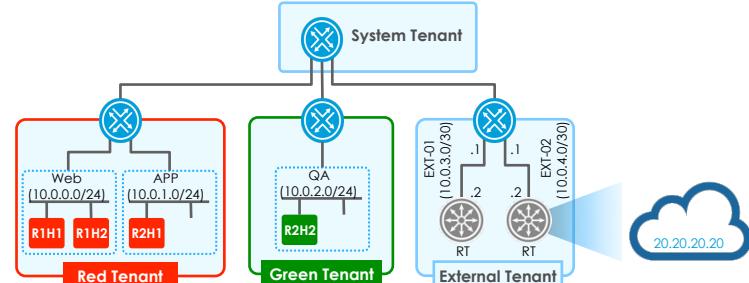
LAB – TROUBLESHOOTING BCF – PAGE (4)

1. Add policy to count packets in Red Tenant

1. Create a new policy “p1” as in previous lab but with only permit statements
2. Count packets from Web to App segments: 10 permit segment-interface Web any ...
3. Count packets from Web to Green Tenant: 20 permit segment-interface Web any ...
4. Add a final statement that permits all traffic and **apply policy**: 99 permit any to any
5. Send traffic. R1H1> ping -c 5 r2h1 ; ping -c 5 r2h2 ; ping -c 5 10.0.3.2 ; ping -c 5 world
6. Check counters: show tenant Red logical-router policy-counter all
7. Clear counters using clear tenant Red ...

2. Add similar policy in system Tenant

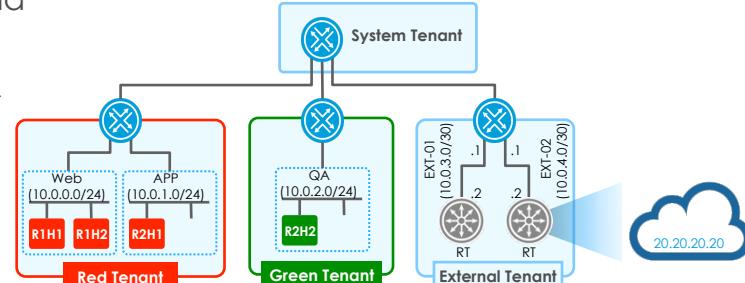
1. Count Red tenant to Green tenant traffic
2. 10 permit tenant Red to tenant Green
3. Count all traffic to External tenant
4. 20 permit any to tenant External
5. Add a final statement that permits all traffic and **apply policy**
6. Send traffic. R1H1> ping -c 5 r2h1 ; ping -c 5 r2h2 ; ping -c 5 10.0.3.2 ; ping -c 5 world
7. Check counters: show tenant all logical-router policy-counter all



LAB – TROUBLESHOOTING BCF – PAGE (5)

1. Add policy to capture packets from Web to App segment

1. Build a new policy “p2” in tenant Red logical router
2. Log packets from Web to App segments
3. 10 permit segment-interface Web any to tenant Red segment App log
4. Add a final statement that permits all traffic: 99 permit any to any
5. Apply policy
6. Since only one policy is active at a time, policy “p1” will no longer be in effect
7. Send traffic. R1H1> ping -c 5 r2h1 ; ping -c 5 world
8. Check packet capture buffer
9. show tenant Red logical-router policy-log
10. Ping packets to r2h1 are captured
11. Ping packets to world are not captured
12. Check packet payload:
show tenant Red logical-router policy-log detail
13. Clear log using clear tenant Red ...



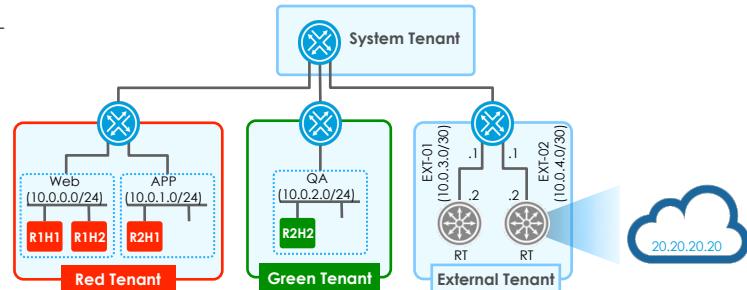
LAB – TROUBLESHOOTING BCF – PAGE (6)

1. Trace path through fabric between two endpoints

1. Perform fabric trace between R1H1 and R2H2 using controller-view
2. test path src-tenant Red src-ip 10.0.0.2 dst-ip 10.0.2.2 ...
3. Identify the logical path
4. Identify the physical path
5. Perform the same trace using fabric view. Setup a test path with name “test1”
6. test path src-tenant <> src-ip <> dst-ip <> fabric-view test-name test1
7. Send traffic from R1H1 within 30 seconds: R1H1> ping -c 5 r2h2
8. Check test path results: show test path test1
9. Identify exact interfaces used by ping packets
10. Delete test1 entry: clear test path ...
11. Perform fabric trace from 10.0.0.2 to 10.0.3.2
12. Perform fabric trace from 10.0.0.2 to 20.20.20.20
13. Perform the same traces from GUI:
(Visibility → Test Path)
14. Simulate = Controller View; Test = Fabric View (see Figure 1)



Figure 1



LAB – TROUBLESHOOTING BCF – PAGE (7)

1. Check controller logs

1. Check logs for the last 30 minutes
2. show logging controller last 30 min
3. Check controller logs for "Error" messages
4. show logging controller complete | grep Error
5. Check controller logs related to newly discovered endpoints
6. show logging controller complete | grep "Endpoint "
7. Check controller logs for leaf switch "R2L1"
8. Check controller logs for interface "R3L1-eth5"

2. View audit logs

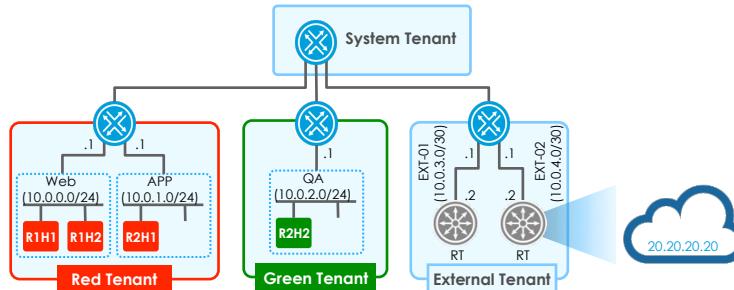
1. show logging audit
2. Check the audit logs of the last 2 hours
3. Filter out only the CLI commands: show logging audit complete | grep cli
4. Pipe the above output into this expression: sed "s/floodlight.*User=//"

LAB – TROUBLESHOOTING BCF – PAGE (8)

Troubleshooting Exercise 1: Inter-segment troubleshooting

Objective: Determine why R1H1 cannot ping R2H1. Fix configuration so that R1H1 and R2H2 can communicate with each other.

1. Load configuration for this exercise
 1. copy snapshot://BCF-Config5 running-config
2. R1H1> ping r2h1
 1. Use troubleshooting flowchart to identify your starting point

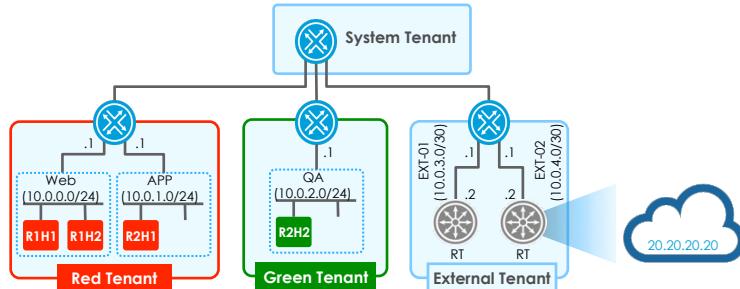


LAB – TROUBLESHOOTING BCF – PAGE (9)

Troubleshooting Exercise 2: Inter-tenant troubleshooting

Objective: Determine why R1H1 cannot ping R2H2. Fix configuration so that R1H1 and R2H2 can communicate with each other.

1. Load configuration for this exercise
 1. copy snapshot://BCF-Config6 running-config
2. R1H1> ping r2h2

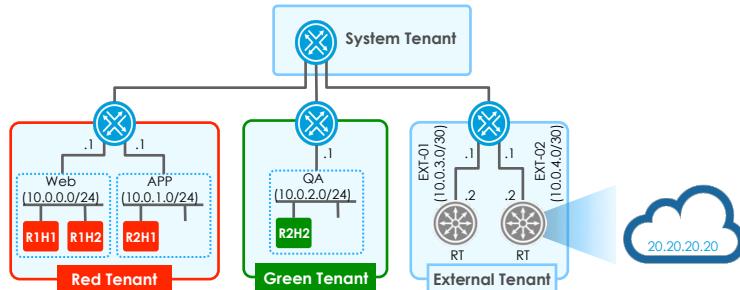


LAB – TROUBLESHOOTING BCF – PAGE (10)

Troubleshooting Exercise 3: External connectivity troubleshooting

Objective: Determine why R1H1 and R2H2 cannot ping 20.20.20.20 (world). Fix configuration so that connectivity is established.

1. Load configuration for this exercise
 1. copy snapshot://BCF-Config7 running-config
2. R1H1> ping world
3. R2H2> ping world



Questions?

Please stop here and let your instructor know that you have finished.

TRAINING OUTLINE

Day 3

Module 7: BCF Operations	40 mins
▪ Upgrade BCF, High Availability ▪ RMA Procedures, Change Management	
Module 8: BCF Analytics	20 mins
▪ Analytics dashboards, Navigation, ▪ Custom & Advanced Queries ▪ Lab – Using Analytics	60 mins
Module 9: BCF vCenter Integration	60 mins
▪ Automation, Visibility, Troubleshooting	
Module 10: BCF In-Depth	60 mins
▪ Fabric Components, Fabric LAGs ▪ Discovering Endpoints, Packet Walks	
Module 11: Contacting Support	10 mins

BSN LABS

Launch Module

- Sign into <http://labs.bigswitch.com/training>
 - Use userid and password provided by instructor
- Launch BCF Module 1



Module 7 – BCF Operations

BCF OPERATIONS

Module Outline

- Upgrade Fabric
- High Availability
- RMA Procedure
- Change Management
- Other Operations
- BCF Status

BCF OPERATIONS

Upgrade Fabric – Overview

Two Partitions

- Divides BCF into two partitions

Phase 1

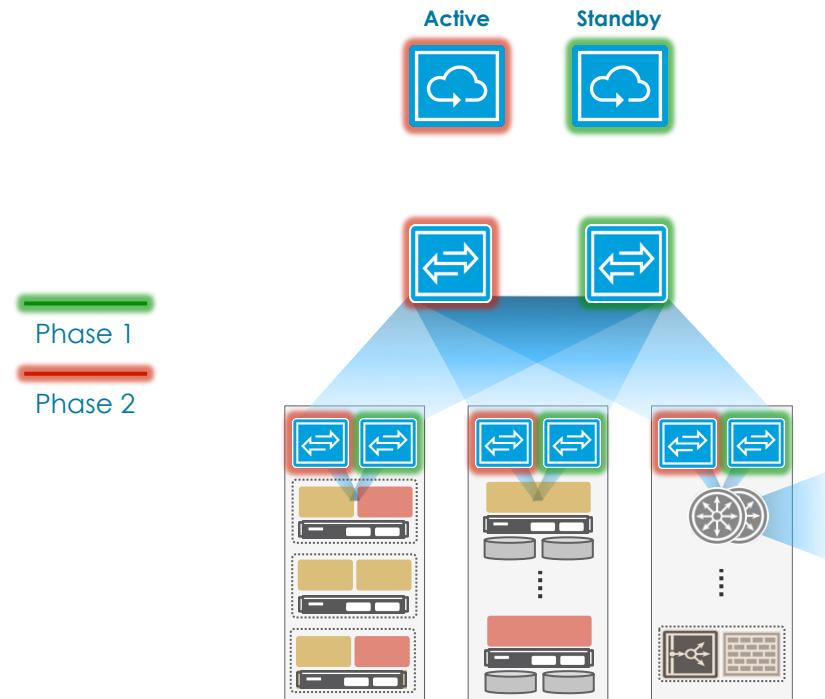
- Standby controller
- Half of the spine switches
- One leaf switch from each leaf group

Phase 2

- Active controller
- Remainder of the switches

Upgrade

- Transitions traffic to phase 2 switches
- Upgrades phase 1 controller & switches
- Transitions traffic to phase 1 switches
- Upgrades phase 2 controller & switches



BCF OPERATIONS

Upgrade Fabric Procedure

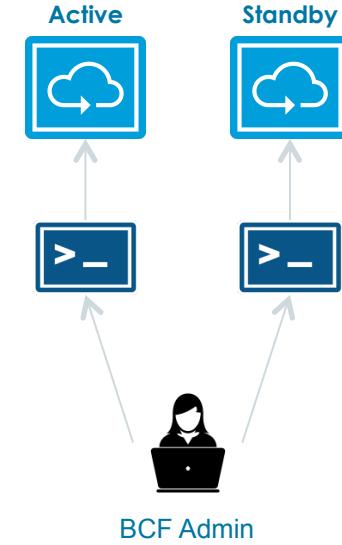
Procedure

- Log in to both controllers as user “admin”
- Check fabric status
 - Upgrade not allowed if fabric errors
- Copy upgrade package to both controllers
- Stage the upgrade on both controllers
- Launch the upgrade on both controllers
 - Upgrade will only proceed after issued on both controllers

```
controller-1# copy scp://<upgrade-package> image://  
controller-2# copy scp://<upgrade-package> image://
```

```
controller-1# upgrade stage  
controller-2# upgrade stage
```

```
controller-1# upgrade launch [pause]  
controller-2# upgrade launch [pause]
```



BCF OPERATIONS

Upgrade Fabric Procedure

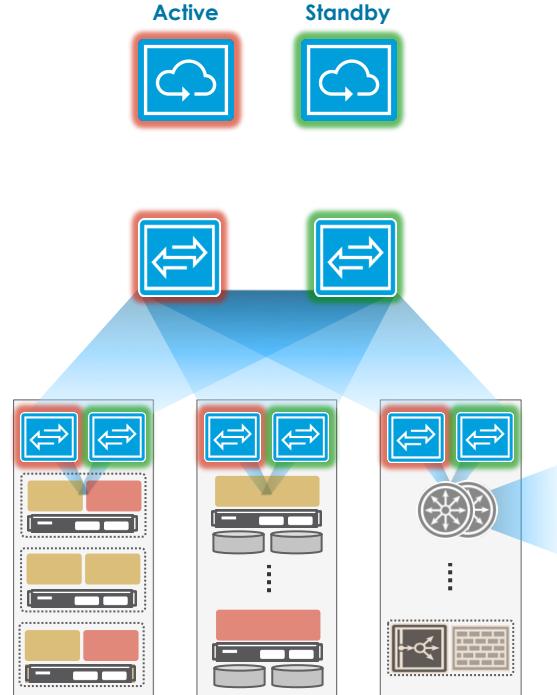
Pause Option

- Causes upgrade to pause after phase 1 is complete and before starting phase 2
- Pause duration specified when upgrade command issued

During the Pause Window

- Perform status checks
- Extend the pause window if more time is required
- Allow phase 2 upgrade to proceed
- Revert upgrade to previous version

```
new-active-controller# upgrade pause extend  
new-active-controller# upgrade pause exit  
new-active-controller# upgrade pause revert
```



BCF OPERATIONS

Upgrade Fabric Status & Firmware Upgrades

Upgrade Progress & Status Checks

- Upgrade progress can be monitored from the new controller
- Verify controller software version on both controllers
- Verify switch software versions
 - Determine current firmware version and new firmware version needed

```
controller# show upgrade progress  
controller# show version  
controller# show switch all version
```

Firmware Upgrades

- ONIE and CPLD must be upgraded manually
- Update from the controller CLI

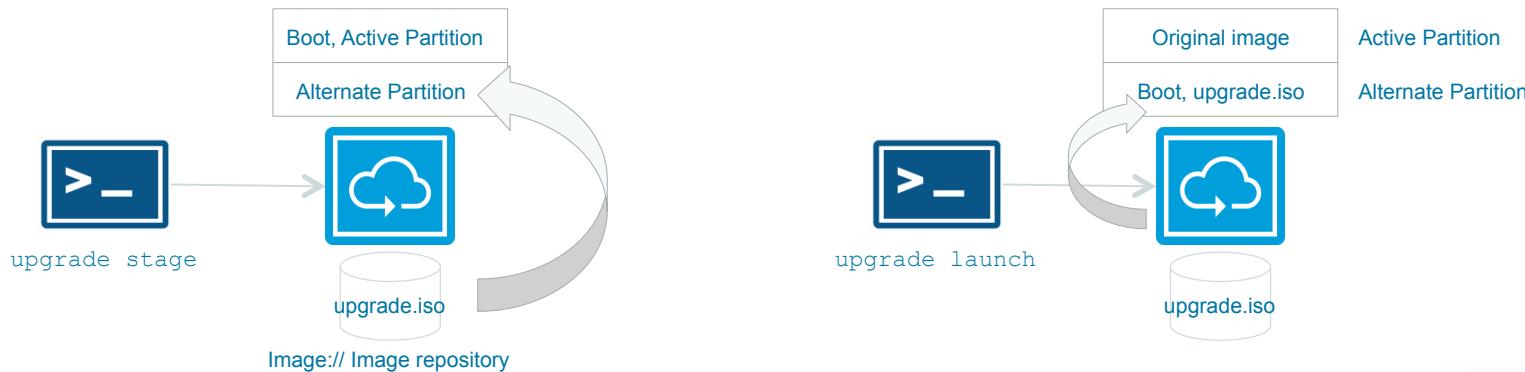
```
controller# system install switch <> onie reboot  
controller# system install switch <> cpld reboot
```

BCF OPERATIONS

Controller Boot Partitions

Boot Partitions

- Two partitions
- During staging, new image loaded into alternate partition
- During launch, system reboots into alternate partition
 - Alternate partition becomes active.
 - Other partition becomes alternate



BCF OPERATIONS

Upgrade Rollback

Rollback Upgrade

- Upgrade can be rolled back (hitless) before phase 2 is completed
- Upgrade can be rolled back after upgrade is complete (not hitless)
- Check boot partitions to see previous version installed on controller

Rollback Procedure

- Move fabric to headless mode
 - Disconnect controllers from P-Switch Control Network
- Save configuration to a remote server
- Boot both controllers to alternate partition
- Verify configuration on controller
- Restore P-Switch Controller connectivity
- Reboot switches

```
controller# show boot partition
```

```
controller# boot partition alternate
```

Questions?

BCF OPERATIONS

High Availability – Failure Scenarios

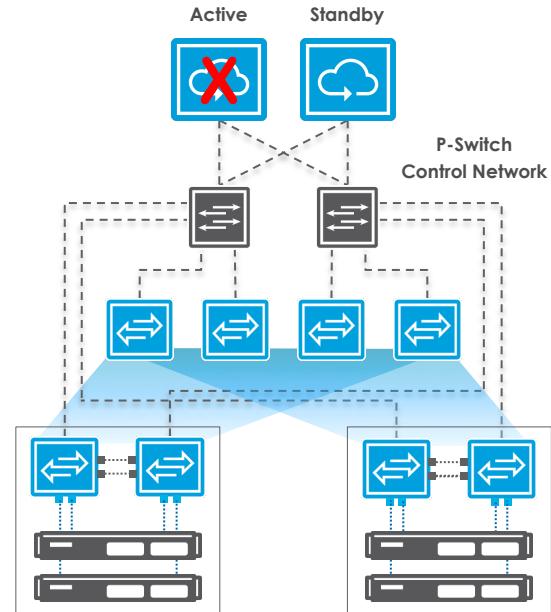
- Active controller failure
- Standby controller failure
- Switch failure (spine, leaf)
- Headless mode failure scenarios
 - P-switch control network failure
- Peer link failure
- Fabric link failure

BCF OPERATIONS

High Availability – Active Controller Failure

Active Controller Failure

- Standby controller detects active controller failure
- Polls all switches for connection to active controller
 - If even one positive acknowledgement, standby controller remains in standby state
 - Standby becomes active controller only if no switch can see the active controller
- Sends a role change notification to each fabric switch
- Confirms configurations installed in each fabric switch using hash signatures
 - If signatures do not match, determines the subset of configuration that is different
 - Pushes only the subset of configuration changes

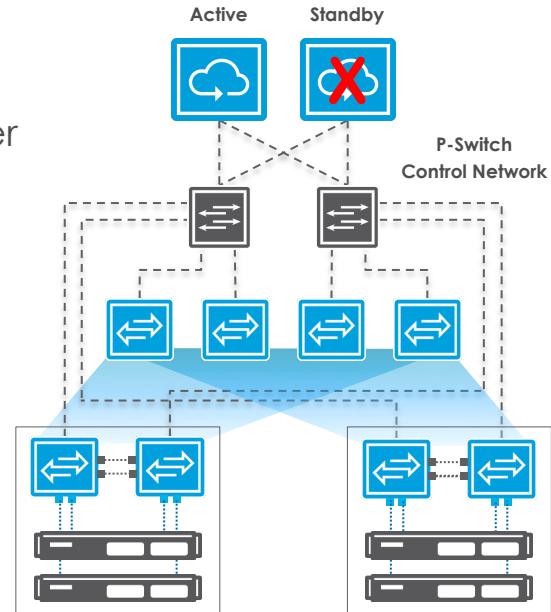


BCF OPERATIONS

High Availability – Standby Controller Failure

Standby Controller Failure

- Standby controller goes down or disconnects
- Active controller and switches detect loss of standby controller
- Active controller is still running
- No impact on traffic
- No changes required in fabric



BCF OPERATIONS

High Availability – Spine Switch Failure

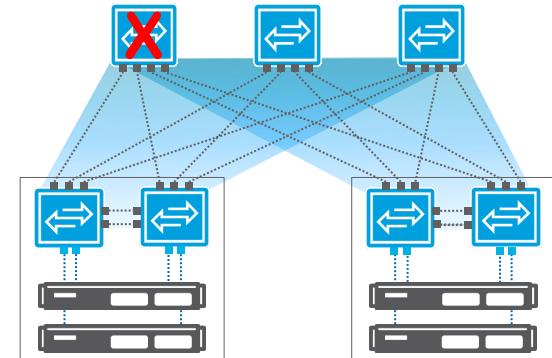
Leaf switches

- Leaf switches detect the loss of link(s) to spine
- Leaf switches remove interfaces from all LAGs that contain links to failed spine switch



Controller

- Controller detects the loss of spine switch
- Controller removes all links from fabric that are connected to failed spine switch
- No changes to forwarding tables required unless failed spine switch isolates a leaf switch or a leaf-group
- LAG tables will be updated to remove links connected to failed switch



BCF OPERATIONS

High Availability – Leaf Switch Failure

Spine

- Spine switches detect the loss of link(s) to leaf group
- Spine switches remove interfaces from all LAGs that contain links to failed switch

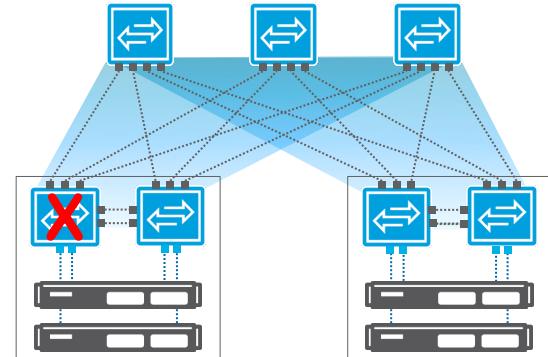


Peer Switch

- Removes the interfaces from the LAG to peer switch

Controller

- Controller detects the loss of leaf switch
- No changes to forwarding table required if every spine switch can still reach the leaf-group
- If a spine switch is unable to reach a leaf-group, the spine will be removed from forwarding tables for any destinations present on the isolated leaf-group
- LAG tables will be updated to remove links to failed switch



BCF OPERATIONS

High Availability – Leaf Switch Failure (Single Switch in Leaf-Group)

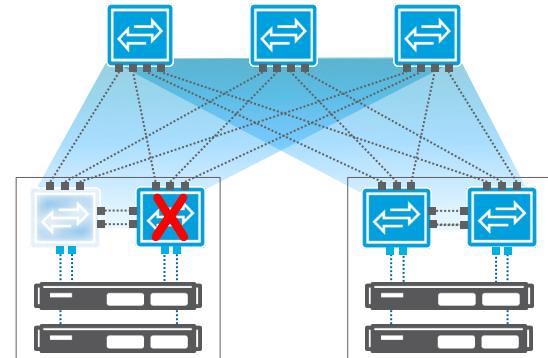
Spine

- Spine switches detect the loss of link(s) to leaf group
- Spine switches remove interfaces from all LAGs that contain links to failed switch



Controller

- Controller detects the loss of leaf switch
- Identifies a leaf-group as unreachable
- Programs all leaf switches and spine switches accordingly
- Inter-segment traffic (intra-tenant) will be dropped by logical router on ingress leaf switch
- Inter-tenant traffic will be dropped by system logical router on spine switches



BCF OPERATIONS

High Availability – Headless Mode

Definition

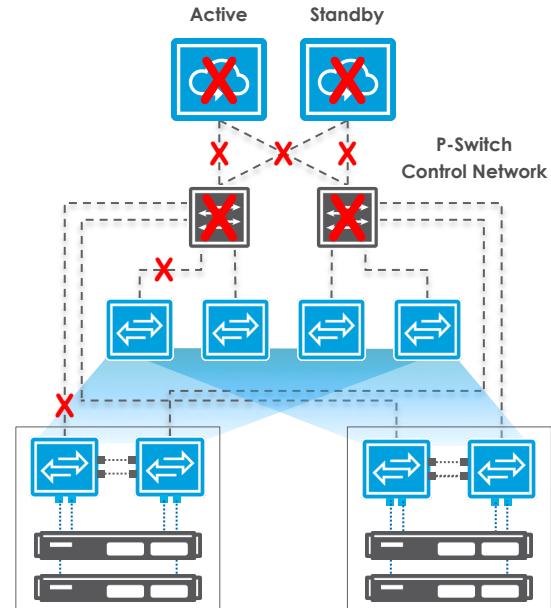
- Any switch or switches no longer able to communicate with either of the controllers

Scenarios

- Dual controller failure
- Failure of all links between controllers and P-Switch control network
- Complete P-Switch control network failure
- Partial P-Switch control network failure
- Switch management link failure

Impact

- Entire fabric operating in headless mode
- Portion of the fabric operating in headless mode

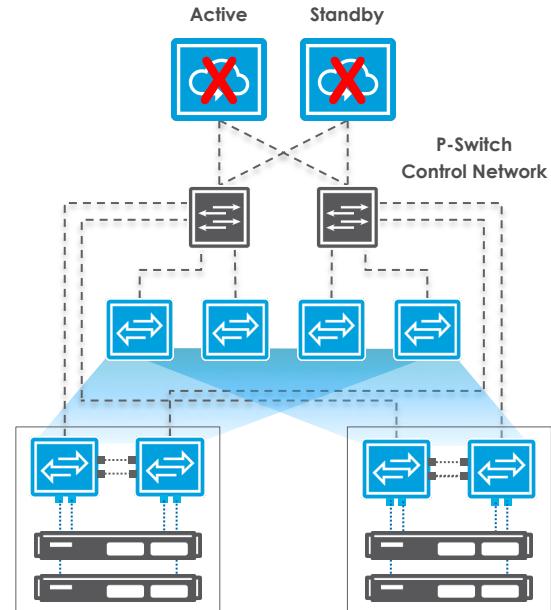


BCF OPERATIONS

High Availability – Headless Mode (Dual Controller Failure)

Dual Controller Failure

- Switches retain all programming
- Packet forwarding continues as normal
- Existing policies continue to be enforced
- No new MAC and IP addresses are learned
- No new provisioning can be performed
- Switches will perform local repair of LAG
 - Remove any member that goes down
- Switches will not add any members to LAG
- MAC and IP addresses will not age-out

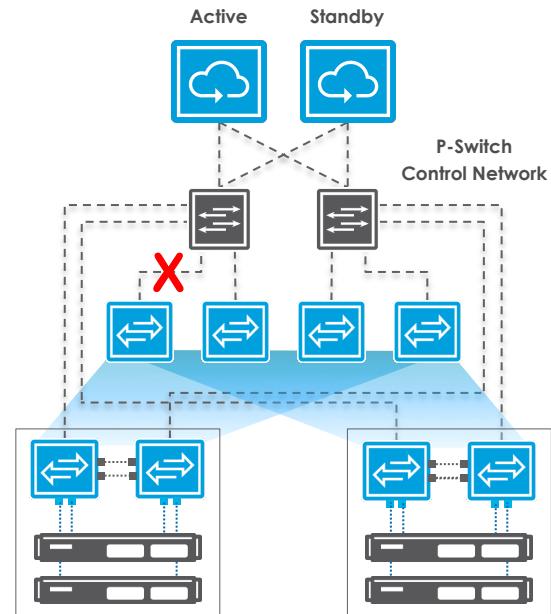


BCF OPERATIONS

High Availability – Headless Spine

Headless Spine

- From the controller perspective, result is the same as if Spine switch has gone down
- Leaf switches still have all the links
- Leaf switches continue forwarding traffic to spine switch until controller reprograms all switches to route traffic around this switch
- No traffic is sent to the headless spine switch but it retains its programming and continues to operate in headless mode
- No impact on functionality other than reduction of overall capacity of fabric bandwidth
- Once the spine re-establishes connectivity, the Active controller updates the affected switch and reprograms the fabric

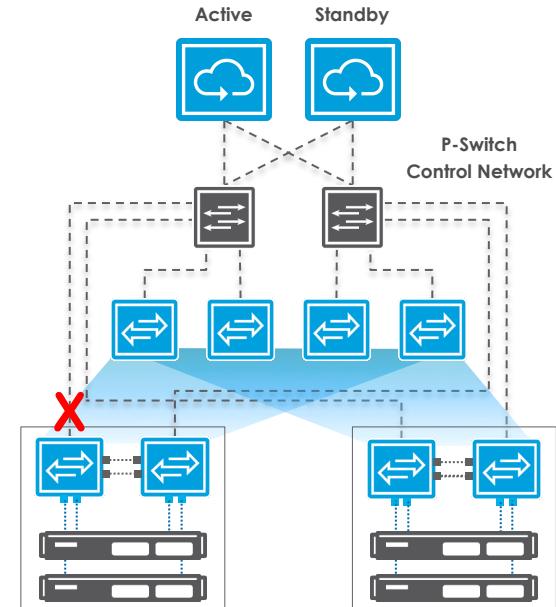


BCF OPERATIONS

High Availability – Headless Switch in a Leaf-Group

Headless Switch (within leaf-group)

- From the controller's perspective, it's the same as if a leaf switch has gone down
- Controller tells the peer switch to send a message to the headless switch to bring down all its links
 - Uplinks to spines
 - Outward facing links to hosts
- End points will no longer forward any traffic to this switch
- Spine and peer switch will stop sending traffic to this switch
- Controller will remove the links from spine and peer switch LAGs
- No programming needed in other spine or leaf switches
- No loss of functionality but throughput capacity for hosts attached to this leaf-group is reduced



BCF OPERATIONS

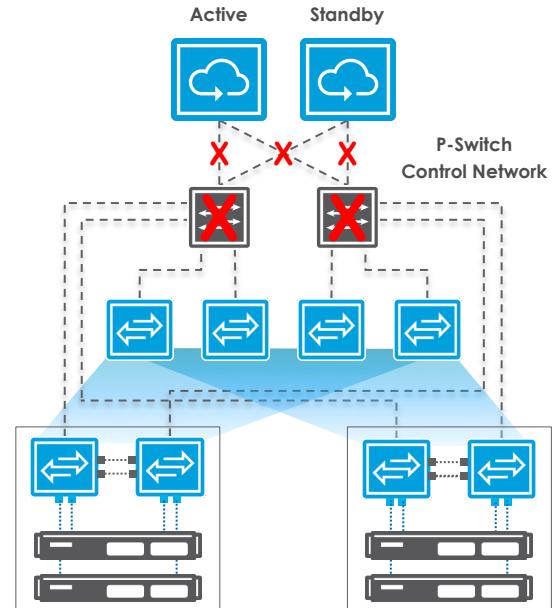
High Availability – Headless Mode (P-Switch Control Network Failure)

Complete P-Switch control network failure

- Both controllers assume mastership (both are up but cannot reach each other)
- Fabric switches go into headless mode
- When control network is restored, an election will determine the new active controller
- Active controller will attempt to configure switches based on its current configuration

Partial P-Switch control network failure

- Some switches lose connectivity to controllers
- Some fabric switches go into headless mode
- Impact will be determined whether it's a single switch in a leaf group or two switches
- Result will be same as per headless spine, headless single leaf switch in a leaf-group, or a headless switch with two switches in a leaf-group

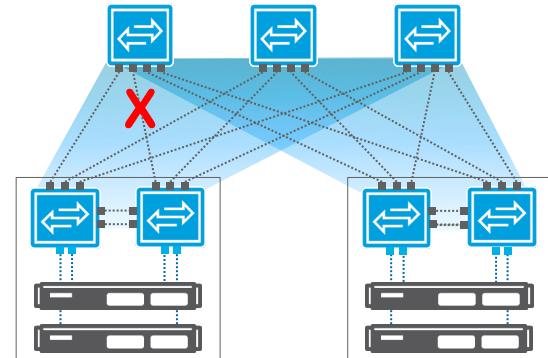


BCF OPERATIONS

High Availability – Spine/Leaf Link Failure

Link Failure

- Link is removed from the LAGs by Spine and Leaf switches
- Controller is notified of the down interface
- No forwarding table reprogramming is required unless the leaf-group is no longer reachable by the same spine switch
- LAG tables are updated to remove the link

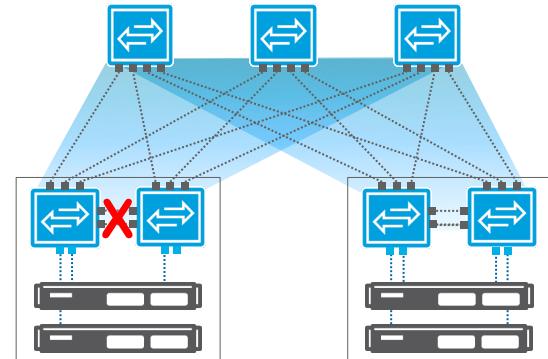


BCF OPERATIONS

High Availability – Peer Link Failure

Peer Link Failure

- No forwarding table reprogramming is required
- LAG tables will be updated to remove the link
- Fabric continues to forward traffic as before
- All dual connected servers to a leaf group will not experience any connectivity issues
- Single connected servers will experience intermittent connectivity for RX traffic based on which leaf switch in the leaf-group receives traffic from the spine switches
- Any interface with BPDU Guard Disable will be shut down to avoid loops



Questions?

BCF OPERATIONS

RMA – Controller Switchover; Remove/Add New Controller

Controller Failover

- Initiate manual controller failover
- Issue command on active or standby controller

```
controller# system failover
```

Remove Controller

- Removes controller from cluster
- Cluster operating with only active controller

```
controller# system remove-node 10.2.24.142
```

Add Controller

- New controller must be same version as active controller
- First boot required to join cluster
- Install new OS using USB to initiate first boot
- Or perform factory default to initiate first boot

```
controller# boot factory-default
```

BCF OPERATIONS

RMA – Switch Replacement Workflow

Replacing Switch

- In configuration mode, “shutdown” the old switch
- Update the configuration to replace MAC address of old switch with that of new switch
- Swap all wires from old switch to new switch including serial console network, P-Switch control network, In-band network if applicable, and any host connections
- Power up the new switch
- Perform “no shutdown” on the switch

```
controller(config)# switch R1L1 ; no shutdown
```

```
switch R1L1
  fabric-role leaf
  leaf-group R1
  mac 00:00:00:02:00:01
  shutdown
```

```
switch R1L2
  fabric-role leaf
  leaf-group R1
  mac 00:00:00:02:00:02
```

```
switch S1
  fabric-role spine
  mac 00:00:01:02:01:02
```

BCF OPERATIONS

Recovering from Dual Controller Failure

Bring Up Controllers

- Bring up first controller using steps given in “BCF Deployment” module
- Bring up second controller using same procedure
 - This step can be performed here or after loading config

Load Configuration

- Load configuration on active controller
 - **Do not load or enter switch configuration only**
 - **This will wipe the programming from all switches as soon as they connect**
- Load the complete configuration
- If configuration is in components
 - Load all other components of configuration first (interface-groups, tenants, segments, ...) before loading switch configuration

BCF OPERATIONS

Change Management – Snapshot Configurations

Configuration Rollback

- Save configuration in a snapshot before making changes

```
controller# copy running-config snapshot://config-pre-workorder-10137
```

- Easy to rollback changes
- Copy snapshot to running-config (replaces running-config)
- Snapshots available locally on the controller
- No remote access or server required

```
controller# copy snapshot://config-pre-workorder-10137 running-config
```

BCF OPERATIONS

Change Management – Update Policies

Updating Policies

- Configure new policy instead of modifying existing policy
- Apply new policy
- Easy to revert back

```
tenant Production
logical-router

policy-list p1
    10 permit segment-interface Services any to any
    20 permit tenant-interface System any to any next-hop ext-fw
    30 permit segment-interface GW any to any next-hop ext-fw

policy-list p2
    10 deny segment-interface GW 10.1.1.0/28 to any
    20 permit segment-interface Services any to any
    30 permit tenant-interface System any to any next-hop ext-fw
    40 permit segment-interface GW any to any next-hop ext-fw
    50 permit any to any

apply policy-list p2
```

BCF OPERATIONS

Maintenance

Locate Switch

- Physically identify switch in a datacenter rack or row
- Helpful for operational personnel
- Lights all switch LEDs for a given duration of time
- Defaults to 30 seconds

```
controller# system beacon switch S1 timeout 60
```

BCF OPERATIONS

Useful Grep Expressions

Grep Expressions

- Only show interfaces with errors (or non zero lines)

```
controller# show switch all interface all counters errors | grep -vP "( +0){3}"
```

- Only show interfaces with traffic (continuously)

```
controller# watch show switch R1L1 interface all counters | grep -vP "( +0){3}"
```

- Only show log messages that do not match switch1

```
controller# show logging controller | grep -v switch1
```

- Only show switches that have uptime greater than 0

```
controller# show switch all connections | grep " 0 "
```

- Only show switches with hostbound traffic

```
controller# show switch all pimu-counters | grep -vP "( +0){5}"
```

BCF OPERATIONS

BCF Status

BCF Status Commands

- Check status of controllers
- Check status of switches
- Check overall fabric status
- Check fabric errors if reported
- Check inter-switch fabric links
- Check level of logging
- Check existence of endpoints

```
controller# show controller
controller# show switch
controller# show fabric
controller# show fabric error
controller# show link
controller# show logging controller last 5mins
controller# show endpoint
```

Questions?



Module 8 – BCF Analytics

BCF ANALYTICS

Module Outline

- Analytics Dashboards
- Navigation
- Custom & Advanced Queries
- Saving & Sharing Dashboards
- **Lab – Using Analytics**

BCF ANALYTICS

Analytics – Overview

Built-in Analytics (Elasticsearch)

- No software installation required
- Immediately accessible from the GUI

Visualize Data

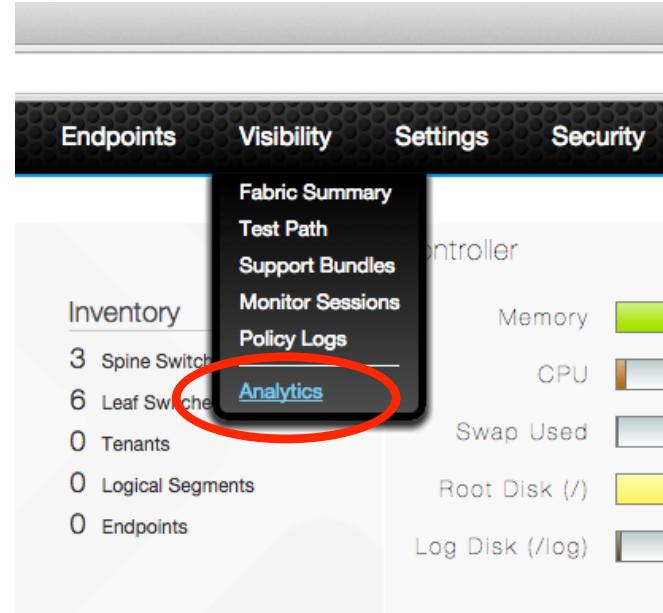
- Switch / Interface / Tenant utilization
- Perform trend analysis

Data Store

- 200 GB separate disk partition
- Up to 1 year of data*
- No summarization/roll-up over time (no loss in granularity)
- Persistent across software upgrades

Data Content

- Controller logs (floodlight, syslog, audit)
- Switch logs
- Stats (collected every 3 minutes)



BCF ANALYTICS

Analytics – Dashboards

Over 30 pre-defined dashboards

Home Page

- Major events during the last 24 hours

Physical

- Parse all logs (controller and/or switches)
- CPU/Mem or Interface utilization stats
- Configuration changes / command history

Logical

- Events, errors, state changes, . . .
- Troubleshoot endpoint discovery / routing
- vCenter related events/messages
- NSX / VM networking
- OpenStack orchestration

Custom

- New dashboards saved on controller (available to all)

Physical

- **Fabric**
 - General
 - Major Events 24hrs
 - All Logs By Device
 - CPU/Memory
 - Switch Report
 - Errors
 - Errors and Warnings
 - Errors by Process
 - Controller/Switch
 - Correlate Logs
 - Switch Interface
 - Rates Statistics
 - Error/Drop Statistics
- **Controller**
 - Configuration Changes
 - CLI Change 24hrs
 - CLI Commands
 - All Config Changes
 - REST API calls
 - Login/Logout

Logical

- **Tenant**
 - Events
 - Statistics
- **Segment**
 - Events
 - Statistics
- **Endpoint (On active only)**
 - Errors/Warnings
 - All Changes
 - Endpoints with same IP
 - Endpoint by MAC Address
 - Endpoint by IP Address
 - Endpoint by Name
- **VMware vCenter**
 - All Events
 - vMotion
 - Virtual Machines
 - VM Networking
 - NSX Networking
- **Orchestration**
 - Network Events
- **Protocols**
 - IGMP
 - BGP

BCF ANALYTICS

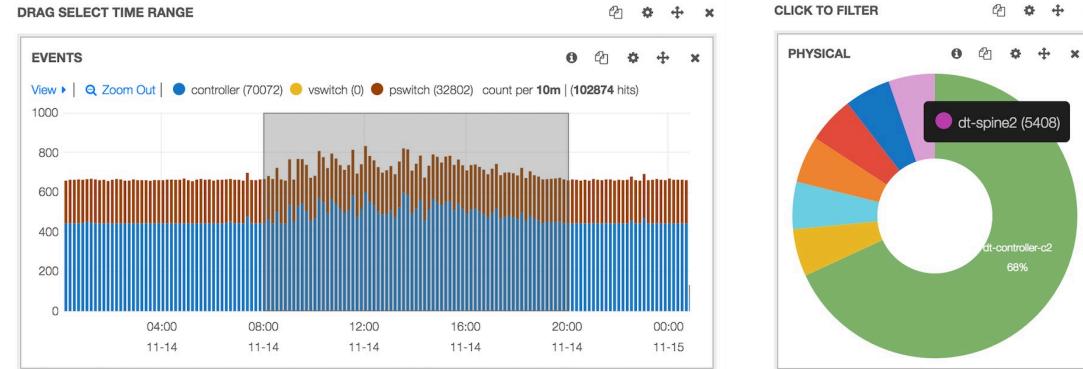
Analytics – Point & Click Navigation

1. Time Range Selector

- Selectable bar graph
- Drill down to specific time period

2. Device Selector

- Selectable pie chart
- Drill down to a specific device



BCF ANALYTICS

Analytics – Point & Click Navigation

1. Time Range Selector

- Selectable bar graph
- Drill down to specific time period

2. Device Selector

- Selectable pie chart
- Drill down to a specific device

3. Field Selector

- Tabular format
- Records of all visualized data entries
- Expandable
- Selectable fields
- Drill down to specific record types

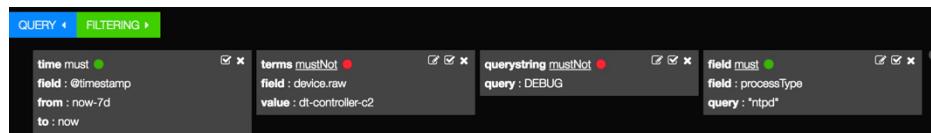
4. Update Filter

- Deactivate criteria
- Delete criteria

@timestamp	device	message
2016-11-15T12:59:55.332-08:00	dt-leaf1a	dt-leaf1a sshd: INFO Received disconnect from fe80::5054:ff%eth0 port 55332 (enci...
2016-11-15T12:59:55.332-08:00	dt-leaf1a	dt-leaf1a sshd: INFO pam_unix(sshd:session): session closed ...

View: [Table](#) / [JSON](#) / [Raw](#)

Field	Action	Value
@timestamp	<input type="text"/> <input type="button"/> <input type="button"/>	2016-11-15T20:59:55.332Z
@version	<input type="text"/> <input type="button"/> <input type="button"/>	1
_id	<input type="text"/> <input type="button"/> <input type="button"/>	AVhpYgOsA5BaFqv7OvE
_index	<input type="text"/> <input type="button"/> <input type="button"/>	logstash-2016.11.15
_type	<input type="text"/> <input type="button"/> <input type="button"/>	logs
action	<input type="text"/> <input type="button"/> <input type="button"/>	pam_unix(sshd:session): session closed for user admin
device	<input type="text"/> <input type="button"/> <input type="button"/>	dt-leaf1a
host	<input type="text"/> <input type="button"/> <input type="button"/>	dt-controller-c2
message	<input type="text"/> <input type="button"/> <input type="button"/>	dt-leaf1a sshd: INFO pam_unix(sshd:session): session closed for user admin
path	<input type="text"/> <input type="button"/> <input type="button"/>	/var/log/switch/dt-leaf1a.log
processType	<input type="text"/> <input type="button"/> Add filter to NOT match this value <input type="button"/>	
severity	<input type="text"/> <input type="button"/> <input type="button"/>	INFO
tags	<input type="text"/> <input type="button"/> <input type="button"/>	
type	<input type="text"/> <input type="button"/> <input type="button"/>	logs



BCF ANALYTICS

Analytics – Point & Click Navigation (More Options)

Field List

- Additional fields for tabular view

Micro Analysis

- Localized analysis of data in table
- Selectable fields
- Updates dashboard filters

Modify Queries

- Edit individual filter boxes
- Modify test conditions
- Activate / deactivate filters

TABLE

Fields

All (37) / Current (20)

Type to filter...

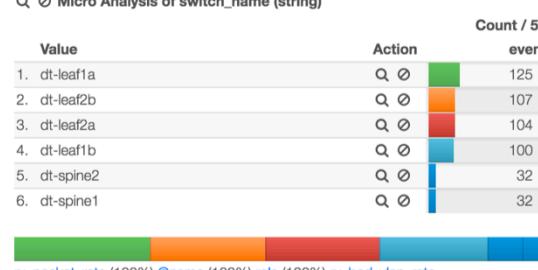
@name
 @timestamp
 _id
 _index
 _type
 role
 rx_bad_vlan_rate
 rx_bcst_packet_rate
 rx_byte_rate
 rx_drop_rate
 rx_error_rate
 rx_mcst_packet_rate
 rx_packet_rate
 switch_name
 tx_bcst_packet_rate
 tx_byte_rate
 tx_drop_rate
 tx_error_rate
 tx_mcst_packet_rate
 tx_packet_rate

0 to 50 of 500 available for

@name	role	tx_packet_rate	rx_packet_rate	tx_bcst_packet_rate	rx_bcst_packet_rate
ethernet49	spine	57	59	0	0
in dt-leaf1b					
ethernet29	edge	0	0	0	0
in dt-leaf1b					
ethernet27	edge	3	3	0	0
in dt-leaf1b					
ethernet23	edge	58	22	0	0
in dt-leaf1b					
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Q Ø Micro Analysis of switch_name (string)

Value	Action	Count / 500 events
1. dt-leaf1a	Q Ø	125
2. dt-leaf2b	Q Ø	107
3. dt-leaf2a	Q Ø	104
4. dt-leaf1b	Q Ø	100
5. dt-spine2	Q Ø	32
6. dt-spine1	Q Ø	32



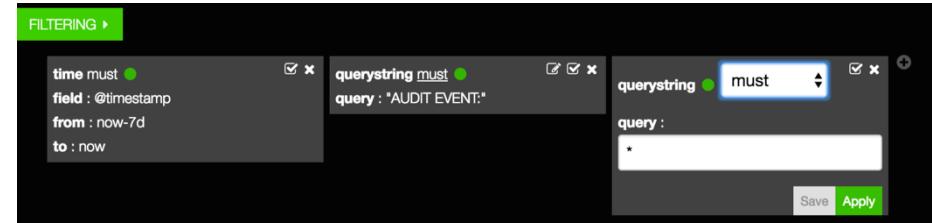
rx_packet_rate (100%), @name (100%), role (100%), rx_bad_vlan_rate (100%), rx_bcst_packet_rate (100%), rx_byte_rate (100%), rx_drop_rate (100%), rx_error_rate (100%), rx_mcst_packet_rate (100%), @timestamp (100%), More ▾

BCF ANALYTICS

Analytics – Custom Queries

Filtering

- Add any number of additional filter statements
- Edit existing filter statements
- Match conditions: must / must not / either
- Matches whole words
- Wildcard "?" matches one character
- Wildcard "*" matches zero or more characters



Sample Queries

- spine
 - spine*
 - spine1 ethernet30
 - "spine1 ethernet30"
 - sw?-ethernet*
 - switch:spine1
 - severity:WARN severity:ERR*
- Matches the word "spine" but not "spine1"
Matches the words "spine1", "spine2", "spineA", "spine-r1", ...
Matches the word "spine1" or the word "ethernet30"
Matches the words "spine1 ethernet30"
Matches expressions like "sw2-ethernet30" but not "sw22-ethernet30"
Matches "spine1" in the field named "switch"
Matches ERROR or WARN messages

BCF ANALYTICS

Analytics – Advanced Queries

Two Levels for Building Data Sets

- Query
- Filtering

Query

- Uses fuzzy logic to select data
- Activate / Deactivate / Pin each query

The screenshot shows the 'QUERY' tab of the Big Switch Analytics interface. It displays three main search conditions:

- time must**: field : @timestamp from : now-7d to : now
- querystring must**: query : "AUDIT EVENT:"
- terms mustNot**: field : method.raw value : GET

Each condition has an 'Activate' (green dot) and 'Deactivate' (red dot) checkbox. A '+' button is located at the bottom right of the query section.

Filtering

- Uses boolean data search conditions

Advanced Operators

- Fuzzy operator (search similar terms)
- Proximity searches

Online Help

- Lucene query syntax
- Regex query syntax

The screenshot shows the 'QUERY' tab with two help pop-ups open:

- Lucene**: A tooltip for the 'lucene' field containing the text "About the lucene query".
- Regex**: A tooltip for the 'regex' field containing the text "About the regex query".

Both pop-ups include a 'Close' button at the bottom right.

BCF ANALYTICS

Analytics – Custom Dashboards

Build new or customize existing dashboard

- Start with an existing dashboard
- Modify as needed
- Add new panels, queries, graphs, charts
- Click  to save dashboard
- Saved dashboards can be accessed from 



Save dashboard to Analytics Home

- Configure “Available Dashboards” panel
- Add saved dashboard
- Revert to the original Home Page using  icon

Share dashboards

- Customize existing dashboard with additional filters
- Email dashboard allowing other to see same report
- Other user must be first authenticated to view shared dashboard
- Click  for shareable URL. Email URL.

Questions?



Lab – Using Analytics

LAB OBJECTIVES

- Use analytics to triage logs
- Learn to navigate the various analytics features
- Perform CLI parsing
- Track BCF activities
- Search for unauthorized logins
- Save, restore, share dashboards

LAB – USING ANALYTICS – PAGE (1)

1. Launch analytics from GUI (Visibility → Analytics)

1. You are at Analytics home with all the pre-configured dashboards
2. The default dashboard: “Major Events 24hrs”
3. Clicking on Big Switch Logo returns you to Big Cloud Fabric GUI (Figure 1)
4. Clicking on “Fabric Analytics” returns you to this page: Analytics Home (Figure 1)

2. Parsing switch logs

1. Select dashboard “All Logs by Device”; the second dashboard under “Fabric / General”
2. Click on date and select custom (Figure 2)
3. Enter from: May 1, 2017 to: “Right Now” or Sep 1, 2017
4. Now, select June – July 2017 in the Time Range window by dragging the mouse across the given time period (Figure 3)



Figure 1

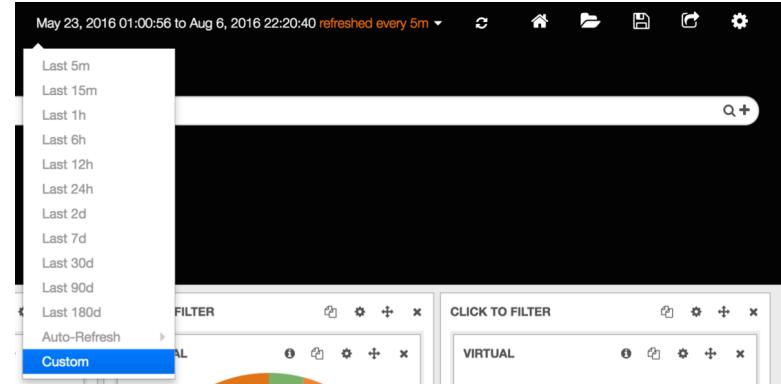


Figure 2

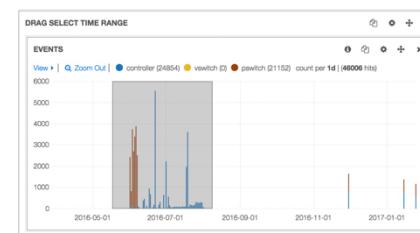


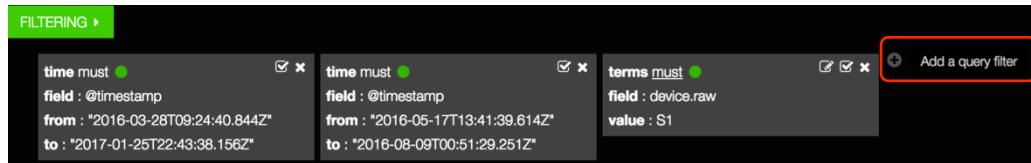
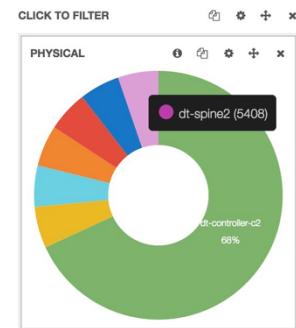
Figure 3



LAB – USING ANALYTICS – PAGE (2)

1. Parse switch logs (Cont'd)

1. Select “S1” to filter out all messages for switch S1 (Figure 1)
2. Add a new query filter box (Figure 2)
3. Type “**eth1**” in query box to filter all S1 eth1 messages (Figure 3)
4. Filter all eth1 messages for all switches other than S1 (Figure 4)
(Change terms from “must” to “must not” in the filter box)
5. Filter all eth1 and eth3 messages for all switches other than S1
(Type “**eth1 eth3**” without any quotes)
6. Remove all filter boxes except timestamp in the “Filtering” pane
7. Add a new filter box and search for “Unqualified module”
(Type “**Unqualified module**” with quotes)
8. Modify previous filter and search for “**SFP**”



LAB – USING ANALYTICS – PAGE (3)

1. Parse switch logs (Cont'd)

1. Perform Micro Analysis on field "Vendor" (Figure 1)
2. Filter "Amphenol" from Micro Analysis window (Figure 1)
(Click on magnifying glass in the "Action" column)
3. Search for all non "Amphenol" optics (Figure 2)
(Modify filter box criteria from "must" to "must not")
4. Search for copper optics "SFP-T" by non Amphenol vendors.
5. Remove all filters except for timestamps (Figure 3)
6. Select all Switch logs. (First select controller (Figure 4) and then change criteria from "must" to "must not" in filter box)
7. **Can you determine if any of the switches have rebooted?**



Figure 2

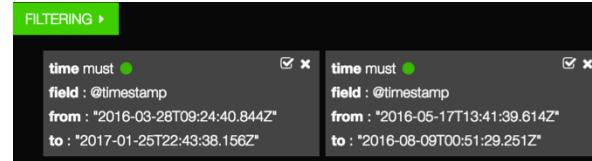


Figure 3

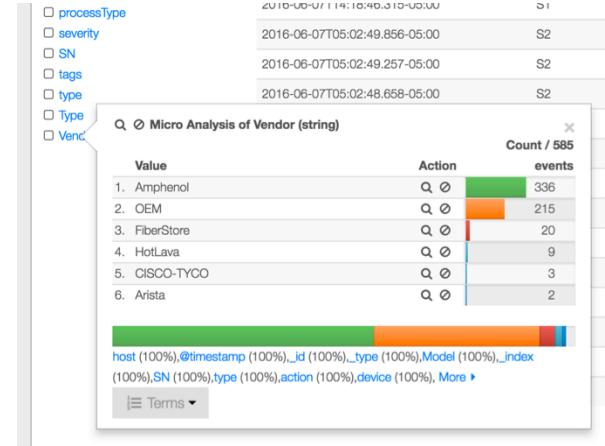


Figure 1

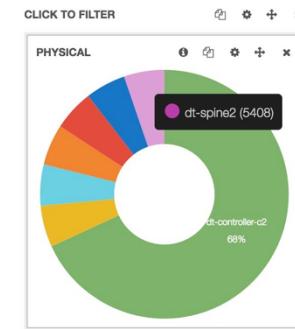


Figure 4

LAB – USING ANALYTICS – PAGE (4)

1. Go to Analytics Home and select dashboard “CLI Commands”

1. Use procedure given on first page of this lab to select time frame May 31 to June 30, 2017
2. Open “Filtering” window (Figure 1)
3. In the “Config Events” pane, click timestamp to sort records starting from June 1st (Figure 2)
4. Configure table size so that 100 records are displayed at a time
 - In the “Config Events” pane, click configure icon (Figure 3)
 - In the “Paging” tab, enter 100 in “Per Page” text box (Figure 4)
5. Exclude logins (Figure 5)
 - Expand record. Click icon “filter to not match” in row “cmd_args”

CONFIG EVENTS

@timestamp	User	cmd_args
2016-06-01T02:00:28.985-05:00	admin	[login]
2016-06-01T02:00:29.006-05:00	admin	[logout]

Figure 2

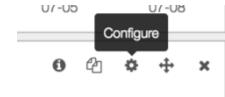


Figure 3

General Panel Paging Queries

Show Controls	Overflow	Per Page	Page limit	Pageable
<input checked="" type="checkbox"/>	expand	100	x 20	= 2000

Figure 4

Figure 1

@timestamp	User	cmd_args
2016-06-01T02:00:28.985-05:00	admin	[login]
2016-06-01T02:00:29.006-05:00	admin	[logout]
2016-06-01T02:01:29.081-05:00	admin	[login]
View: Table / JSON / Raw		
Field	Action	Value
@timestamp	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	2016-06-01T02:01:29.081Z
@version	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	1
User	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	admin
_id	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	dAv2RirYSH-I-OnowuVrA
_index	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	logstash-2016.06.01
_type	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	logs
action	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	Add filter to NOT match this value JD1001: AUDIT EVENT: Session
cmd_args	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	[login]
device	<input type="checkbox"/> <input type="radio"/> <input type="radio"/>	controller

Figure 5

LAB – USING ANALYTICS – PAGE (5)

1. Dashboard “CLI Commands” (Cont’d)

1. Exclude logouts
2. Exclude blank entries (user just hitting enter key at the CLI)
 - Expand record. Click icon “filter to match” in row cmd_args (Figure 1)
 - Edit filter. Add “*” to the end of expression in “query” field. (Figure 2) The final expression should look as follows: “\\“\\”*”
3. You should be able to page through all the commands
4. Search for an “upgrade” that was performed
5. Search for any “no” commands issued by user.
Meaning configuration deletes
6. Remove all filters except for first two filters (Figure 3)

_type	Q Ø # logs
action	Add filter to match this value
cmd_args	Q Ø # "
device	Q Ø # controller

Figure 1

The screenshot shows a search interface with a dark theme. At the top, there is a dropdown menu set to "must". Below it, there is a section labeled "field:" with a dropdown menu set to "cmd_args". Underneath, there is a section labeled "query:" containing the text "\\\"\\\"*". At the bottom right, there are "Save" and "Apply" buttons.

Figure 2



Figure 3

LAB – USING ANALYTICS – PAGE (6)

1. Dashboard “CLI Commands” (Cont'd)

1. Open “Fields” list (Figure 1) and perform Micro Analysis on “User” (Figure 2)
2. Filter all commands issued by “ben”
3. Again exclude blank entries to see complete list of commands and login sessions
4. Remove previous filter
5. Search for any “policy” related commands entered on system
6. Search for any specific command that you would like to search. E.G. password, tenant



Figure 3

Show field list EVENTS		
@timestamp	User	cmd_args
2016-06-01T02:00:28.985-05:00	admin	[login]
2016-06-01T02:00:29.006-05:00	admin	[logout]

Figure 1

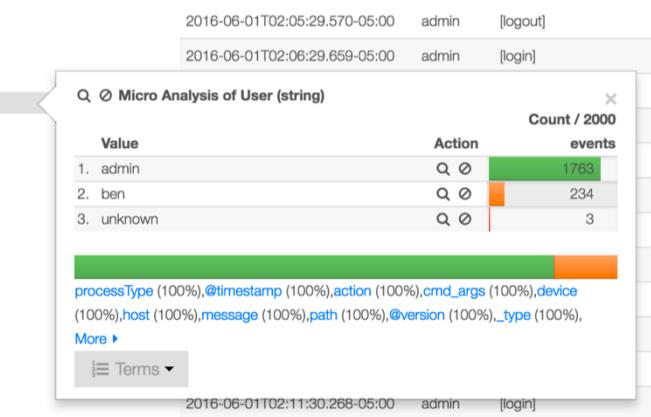


Figure 2

LAB – USING ANALYTICS – PAGE (7)

1. Use analytics to determine unauthorized logins during June 2017

1. From Analytics Home, select dashboard “All Logs By Device”
2. Select June 2017 time frame
3. Try following search terms one at a time and see which produces the best results:
login, failed, auth*, or authentication
4. Now try “authentication failed”
5. How many records are there for failed authentication?
6. How many attempt were made for user “root”
7. How many failed attempts were made using a user other than “root”

2. Save and share dashboard

1. Save this dashboard under the title “Failed Login Attempts” (Figure 1)
2. Goto Analytics Home
3. Reload the “Failed Login Attempts” dashboard (Figure 2)
4. How can this report be shared with another person?



Figure 1



Figure 2

Questions?

Please stop here and let your instructor know that you have finished.



Module 9 – BCF vCenter Integration

BCF INTEGRATIONS

Module Outline

- Virtual Networking
- vCenter Integration
- Automation
- Troubleshooting
- vCenter Plug-in
- NSX Integration
- Analytics

BCF INTEGRATIONS

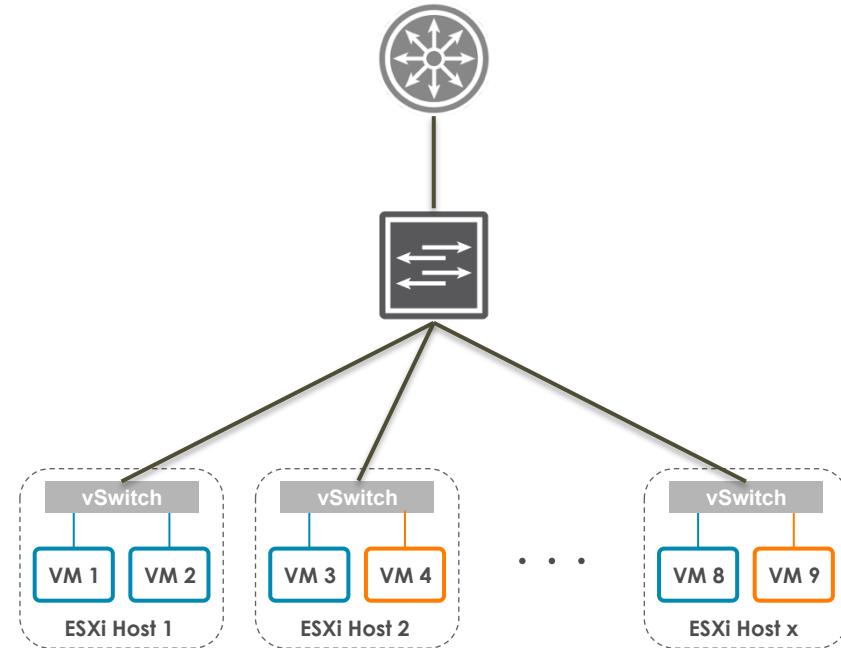
Virtual Networking

Virtual Infrastructure

- VMs
- Virtual switch
- Uplinks to switch between hosts
- Routing for multiple VLANs & External connectivity

Physical Infrastructure

- Configure switch interfaces or LAGs for ESXi hosts
- Configuration VLANs
- Configure routing
- Configure policy
- Migrate policies during vMotion or DRS



Replicate

- Across 10s, 100s, or 1000s of hosts
- Redundant, Error-prone

BCF INTEGRATIONS

Virtual Networking – Closing the Overlay / Underlay Gap

Configuration

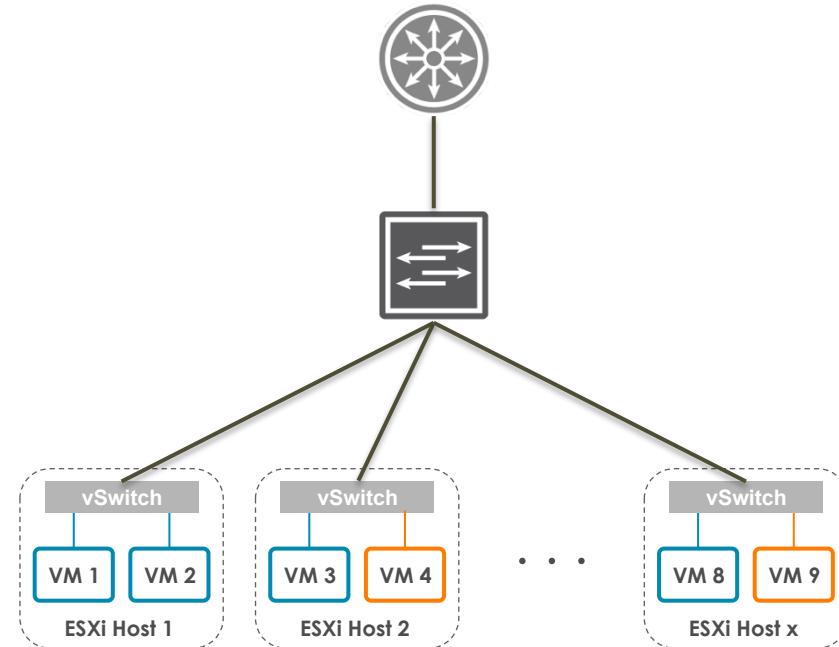
- Avoid having to configure overlay and underlay separately and possibly repeating configuration
- Policies should automatically move for VMs moving from one host to another

Visibility

- See into the physical underlay network from VM console
- See into the virtual overlay network from network console

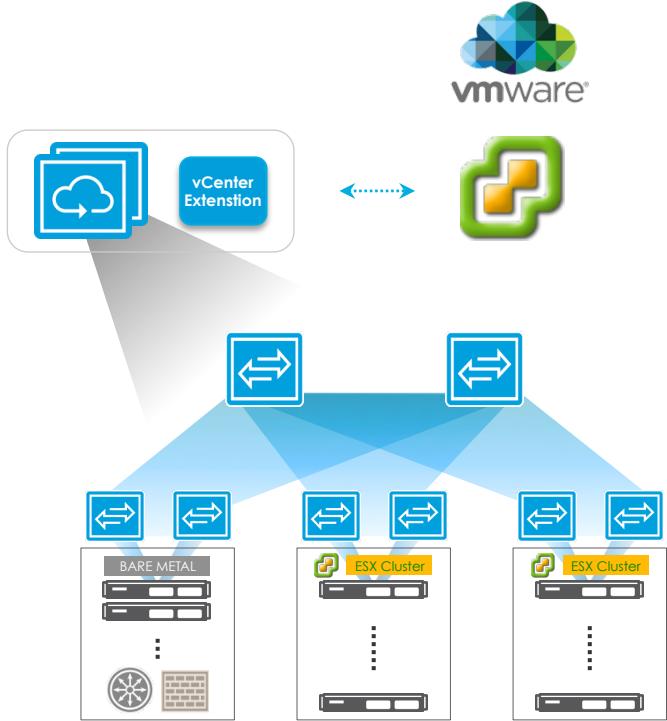
Troubleshooting

- Troubleshoot from either VM console or network console with similar capabilities and visibility



BCF VCENTER INTEGRATION

vCenter Integration



Single source for virtual/physical network configurations

Automation

- Auto learn virtual network (vSwitches, vCenter port groups, and VMs)
- Auto detect ESXi hosts
- Auto configure host uplinks, tenant, L2 segments & VMs
- Auto policy migrations for vMotion/DRS

Visibility

- Virtual network for BCF admin
- Physical network for vCenter admin

Troubleshooting

- VM to VM level traceability for both admins
- Test path (fabric trace) utility available to both admins
- VCenter events captured in analytics

BCF VCENTER INTEGRATION

VMWare Solution Use-cases

1



vCenter



P-Fabric

- Automation
ESX host detection, L2 network creation, vMotion
- L3 configuration using vSphere GUI Plug-in
- VM-level visibility and troubleshooting

2



NSX Controller



P-Fabric

- Automation
Host detection, Transport network creation for VTEP, vMotion, and Storage port groups
- NSX Analytics for network admin
- VTEP-VTEP, VM-VM, VM-host fabric trace

3



Virtual SAN Datastore



P-Fabric

- Automation
Host detection, Auto-configuration for vSAN networking
- One-click multicast deployment
- Node-level visibility and Troubleshooting

4



NSX



openstack



P-Fabric

- VMware private/public clouds with OpenStack Orchestration
- Fully automated, zero-touch fabric
- vSphere & NSX visibility, fabric-wide troubleshooting

BCF VCENTER INTEGRATION

Integration Procedure

Enable Integration

- Reads configuration from vCenter
- **Automation:** Creates an interface-group for each ESXi host uplinks
- **Automation:** Creates a new tenant for vCenter Layer 2 configurations
- **Automation:** Creates a segment for each vCenter port group
- **Automation:** Creates a static endpoint for each VM
 - No Mac learning or ARPs required

```
vcenter vcenter45
  host-name 10.8.23.45
  user-name administrator@vsphere.local
  hashed-password 78201a00251408425a5001045f565d
  description 'New vcenter'
  tenant-name Vcenter45
  automation-level full
```

BCF VCENTER INTEGRATION

Configuration – Segments

```
tenant Vcenter45
  segment Vcenter45-0
    description ' 2 vSphere portgroups:  PXE, NET-Untagged-10'

  segment Vcenter45-25
    description ' 1 vSphere portgroups:  NET-25'

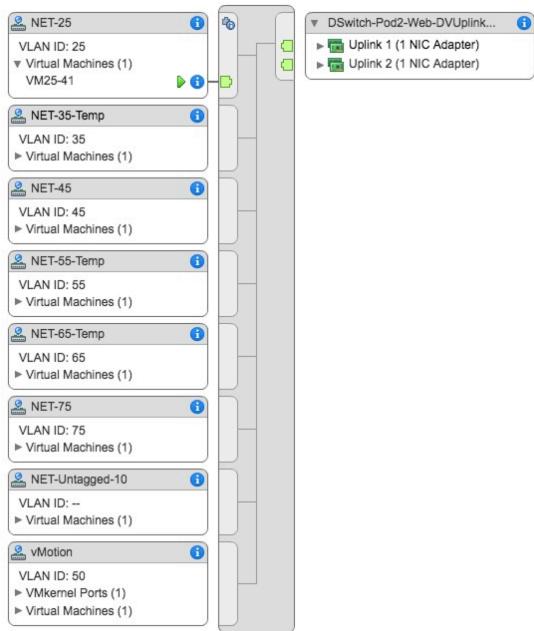
  segment Vcenter45-35
    description ' 3 vSphere portgroups:  DB, NET-35, NET-35-Temp'

  segment Vcenter45-45
    description ' 1 vSphere portgroups:  NET-45'

  segment Vcenter45-55
    description ' 2 vSphere portgroups:  NET-55, NET-55-Temp'

  segment Vcenter45-65
    description ' 2 vSphere portgroups:  NET-65, NET-65-Temp'

  segment Vcenter45-75
    description ' 1 vSphere portgroups:  NET-75'
```



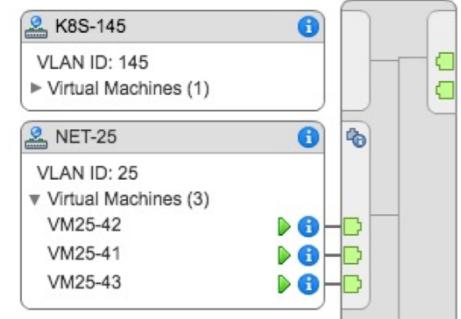
BCF VCENTER INTEGRATION

Configuration – Members & End Points

```
tenant Vcenter45
segment Vcenter45-25
  description ' 1 vSphere portgroups: NET-25'

  member interface-group ESXi-41-DSwitch-Pod2-Traffic vlan 25
  member interface-group ESXi-42-DSwitch-Pod2-Traffic vlan 25
  member interface-group ESXi-43-DSwitch-Pod2-Traffic vlan 25

  endpoint vm-167-00-50-56-bd-d0-82
    description 'VM25-41 (NIC 1) on ESXi Host 10.8.23.41 [25.25.25.141]'
    mac 00:50:56:bd:d0:82
  endpoint vm-189-00-50-56-9f-6b-1c
    description 'VM25-42 (NIC 1) on ESXi Host 10.8.23.42 [25.25.25.142]'
    mac 00:50:56:9f:6b:1c
  endpoint vm-191-00-50-56-9f-31-75
    description 'VM25-43 (NIC 1) on ESXi Host 10.8.23.43 [25.25.25.143]'
    mac 00:50:56:9f:31:7
```



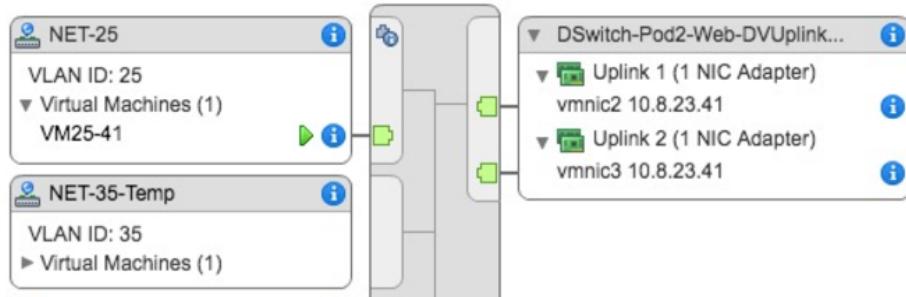
BCF VCENTER INTEGRATION

Configuration – Interface Groups

```
interface-group ESXi-41.qa.bigswitch.com-DSwitch-Pod2-Traffic
  description 'Interface group for virtual switch DSwitch-Pod2-Traffic host ESXi-41.qa.bigswitch.com'
  mode lldp
  member host ESXi-41.qa.bigswitch.com interface vmnic2
  member host ESXi-41.qa.bigswitch.com interface vmnic3
```

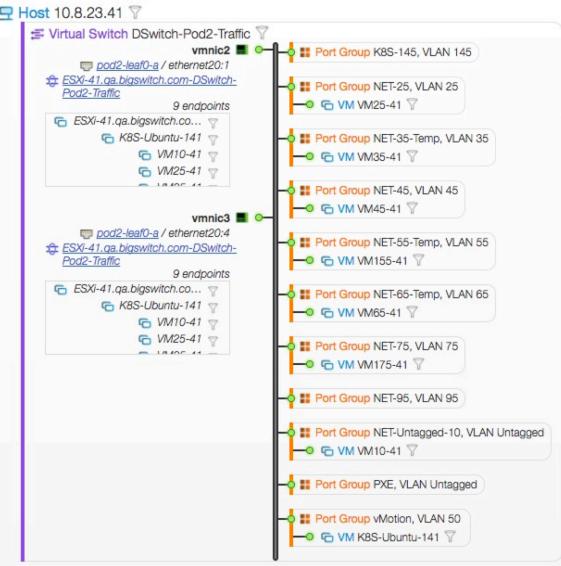
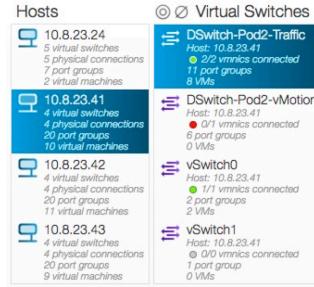
Uplink Automation

- Based on hash (NIC Teaming) setting of vCenter port group
- One interface-group consisting of all uplinks if using IP-based hashing
- One interface-group per uplink for all other types of hashing

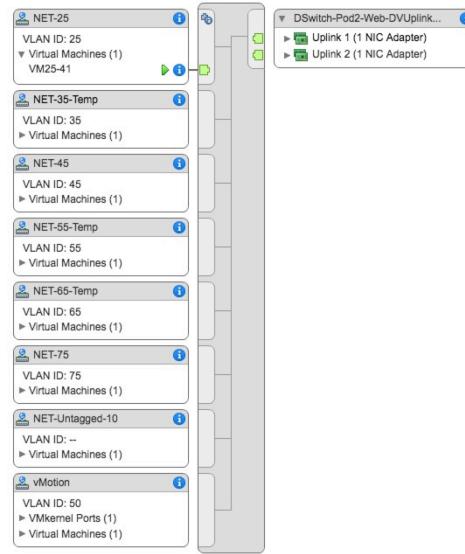


BCF VCENTER INTEGRATION

Visibility of Virtual Networks



BCF View



vCenter View

- Automatically discovered hosts, virtual switches, vCenter port groups, and VMs (Endpoints)
- Independent of automation level

BCF VCENTER INTEGRATION

Visibility of VMs

Endpoints

vm25													
vSphere Endpoint Name	Power State	Active in BCF	Segment	Tenant	VLAN	Host DNS Name	Physical Connection	Virtual Switch	Endpoint Type	MAC Address	IP Addresses	Default Gateway	Gateways
Filtered Results													
VM25-41	✓ powered-on	✓ Yes	Vcenter45-25	Vcenter45	25	ESXi-41.qa.bigswitch.com	[vmnic3 - pod2-leaf0-a / ethernet20:4 [ESXi-41.qa.bigswitch.com-DSwitch-Pod2-Traffic]] [vmnic2 - pod2-leaf0-a / ethernet20:1 [ESXi-41.qa.bigswitch.com-DSwitch-Pod2-Traffic]]	DSwitch-Pod2-Traffic	virtual-machine	00:50:56:bd:d0:82	25.25.25.141/24, 2025:2025:25:141/64, fe80::250:56ff:fed0:0082/64	25.25.25.200	Network 0.0.0.0/0; Address 25.25.25.200, Network ::/0; Address 2025:2025:200
VM25-42	✓ powered-on	✓ Yes	Vcenter45-25	Vcenter45	25	ESXi-42.qa.bigswitch.com	[vmnic2 - pod2-leaf0-a / ethernet20:2 [ESXi-42.qa.bigswitch.com-DSwitch-Pod2-Traffic]]	DSwitch-Pod2-Traffic	virtual-machine	00:50:56:9f:6b:1c	25.25.25.142/24, fe80::250:56ff:fe9f:6b1c/64, 2025:2025:25:142/64	25.25.25.200	Network 0.0.0.0/0; Address 25.25.25.200, Network ::/0; Address 2025:2025:200
VM25-43	✓ powered-on	✓ Yes	Vcenter45-25	Vcenter45	25	ESXi-43.qa.bigswitch.com	[vmnic3 - pod2-leaf1-a / ethernet37 [ESXi-43.qa.bigswitch.com-DSwitch-Pod2-Traffic]] [vmnic2 - pod2-leaf1-b / ethernet37 [ESXi-43.qa.bigswitch.com-DSwitch-Pod2-Traffic]]	DSwitch-Pod2-Traffic	virtual-machine	00:50:56:9f:31:75	25.25.25.143/24, fe80::250:56ff:fe9f:3175/64	25.25.25.200	Network 0.0.0.0/0; Address 25.25.25.200

Jun 21, 2017, 22:50:57 GMT

Endpoint Information

- Name of VM
- State of endpoint (VMs)
- Virtual switch to which its connected
- ESXi host of VM and its uplinks
- Connectivity of ESXi host to physical network
- Tenant and segment mapping

BCF VCENTER INTEGRATION

vCenter GUI Plugin

Visibility for VM Admin

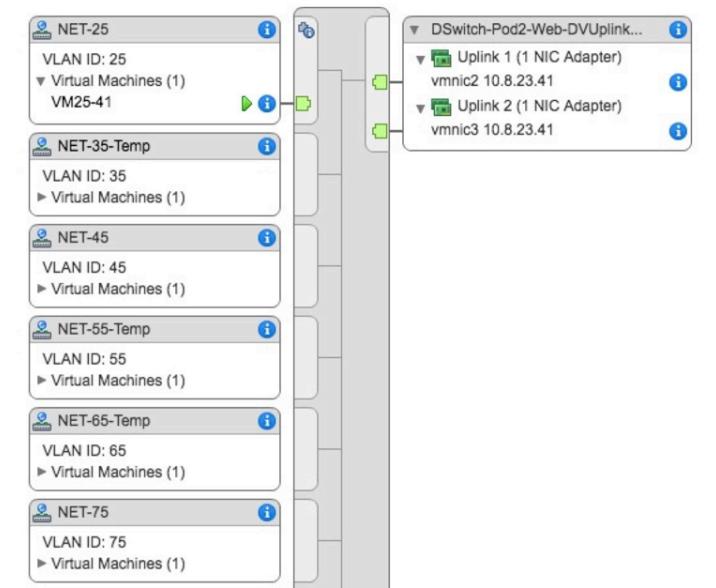
- Requires vCenter GUI Plugin (deploys with a single BCF command)
- Physical network connectivity
- BCF configuration (tenants, segments, ...)

```
controller# deploy vcenter gui-plugin 10.0.0.18 root
```

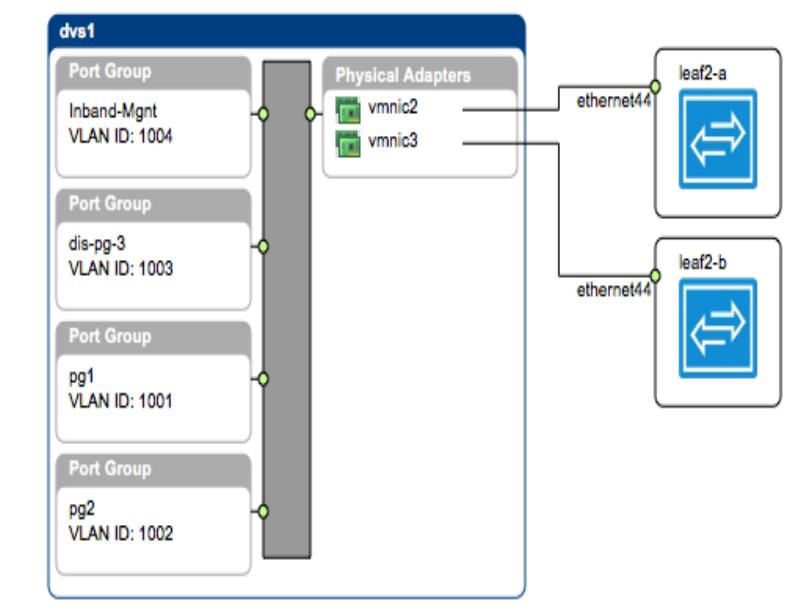


BCF VCENTER INTEGRATION

VM Admin – Visibility of Virtual/Physical Network



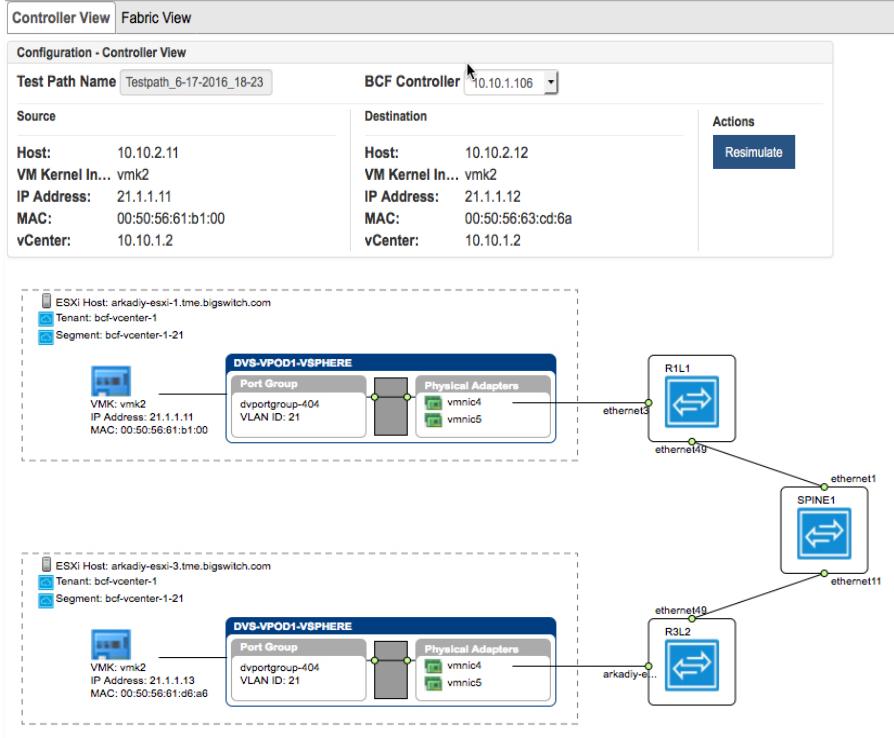
vCenter Network Topology



BCF Plug-in Network Topology

BCF VCENTER INTEGRATION

VM Admin – Troubleshooting Virtual/Physical Network



- VM admin troubleshooting using test path
- Virtual + physical path view between VMs or VMkernels
- Test repository for repeat / periodic execution

BCF VCENTER INTEGRATION

vCenter Extension and GUI Plug-In

vSphere Deployment Workflow	Traditional NW	vCenter Extension	vCenter GUI Plugin
Configure vSwitch / DVS	vCenter Admin	vCenter Admin	vCenter Admin
Configure Port-group	vCenter Admin	vCenter Admin	vCenter Admin
Create VM and associate port-group	vCenter Admin	vCenter Admin	vCenter Admin
Configure L2 VLAN	Network Admin	Auto Creation	Auto Creation
Configure L3 gateway	Network Admin	Network Admin	vCenter Admin OR Network Admin
Configure Routes	Network Admin	Network Admin	vCenter Admin OR Network Admin
Configure Policy	Network Admin	Network Admin	Network Admin

BCF VCENTER INTEGRATION

Troubleshooting

Troubleshooting with vCenter Integration

- Same BCF tools as discussed in Troubleshooting Module
- Additional visibility provided for vCenter networks via CLI commands & GUI console

```
controller# show vcenter <>
controller# show vcenter <> endpoint
controller# show vcenter <> esxi-host
controller# show vcenter <> virtual-network
```

- Test path from vCenter console
 - Provides complete path from VM through virtual switches and physical switches
- Logs – BCF Controller: /var/log/vsphere-extension.log
- Analytics
 - Captures vCenter events & state and correlates with BCF

BCF VCENTER INTEGRATION

Troubleshooting - Checks

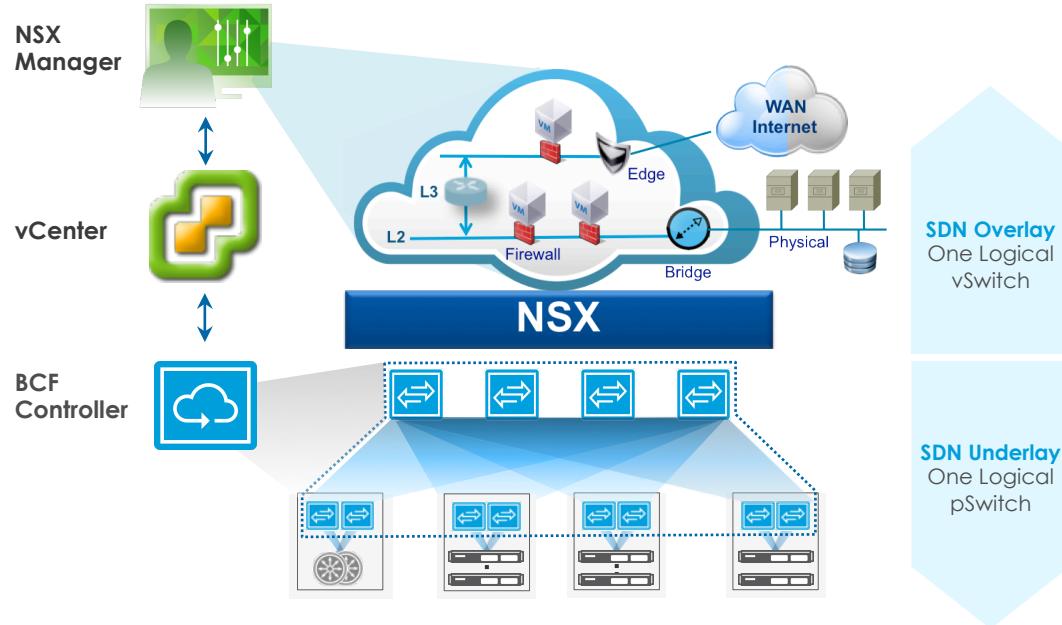
Checks

- CDP/LLDP enabled on vSwitch or distributed virtual switch
- Hashing on vCenter port group
 - “Route based on IP hash” recommended
- No special characters in ESXi and FQDN
 - Allowed characters: alphanumeric, dash (-), asterisk (*), or period (.)
- Hostname+FQDN+Switchname should be under 58 characters
- Domain name cannot be empty
- No LACP for NIC teaming
- All uplinks on ESXi host connected to same leaf-group

Questions?

BCF VCENTER INTEGRATION

NSX – SDN Architecture across Overlay and Underlay



Automation

- Auto host detection
- Auto host uplink LAG formation
- Auto transport network creation for VTEP, vMotion, and Storage

Visibility

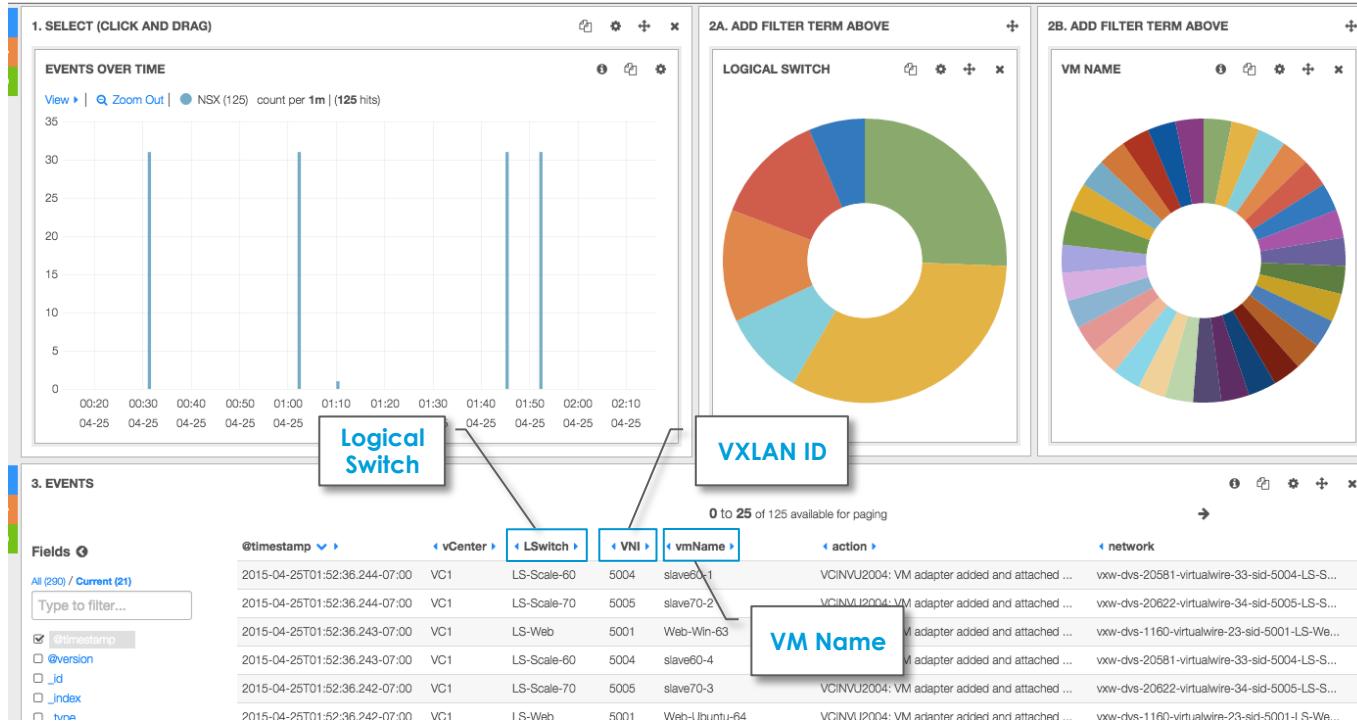
- Up to VTEP on each ESXi host

Troubleshooting

- Same set of tools (test path)
- VTEP to VTEP level fabric trace
- NSX analytics for network admin

BCF VCENTER INTEGRATION

NSX – Analytics

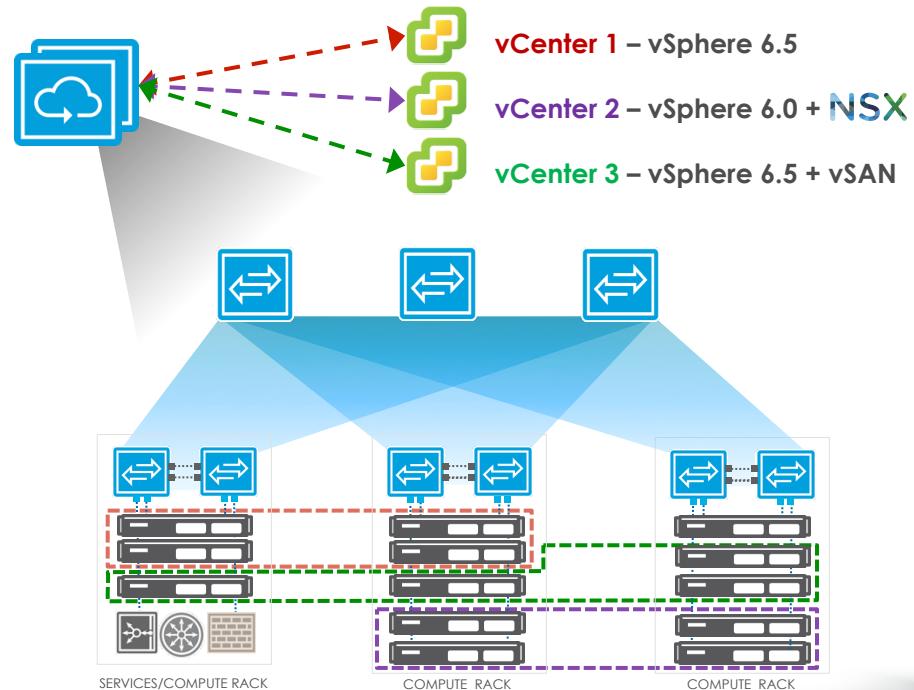


BCF VCENTER INTEGRATION

Logical VPODs

Logical vPOD

- A single instance of vCenter domain integrated with Big Cloud Fabric
- Supports integration of multiple vCenter domains into a single BCF
- vPOD can integrate using same or different versions of vCenter
- Each vCenter domain or vPOD lives within its own tenant
- Supports overlapping IPs within and across vPODs
- Supports overlapping VLAN tags across vPODs



Questions?



Module 10 – BCF In-Depth

BCF IN-DEPTH

Module Outline

- Fabric Components
- Fabric View
- Fabric LAGs
- Endpoints
- Segmentation and Routing
- Forwarding packets

BCF IN-DEPTH

Fabric Components

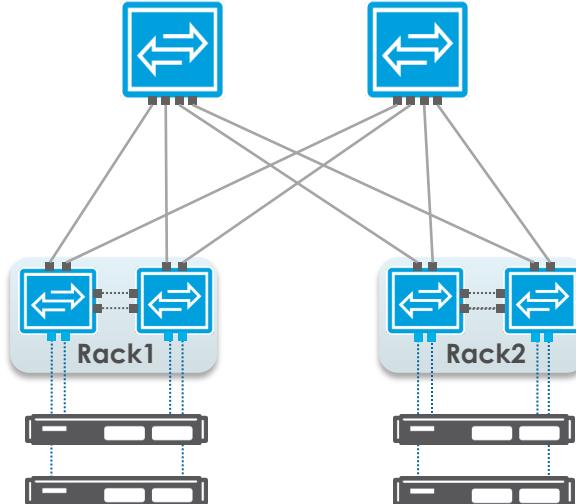
Physical Components

- Nodes
- Interfaces (switch ports)
- Links (connectivity between two switches)

Logical Components

- Nodes
 - Spine
 - Rack (Leaf Group)
 - Leaf
- Fabric LAGs (Inter-switch Links)
 - Spine LAG
 - Rack LAG
 - Peer LAG
 - Inter Tenant LAG

• • •



BCF IN-DEPTH

Discovering Physical Components

Discovering Nodes

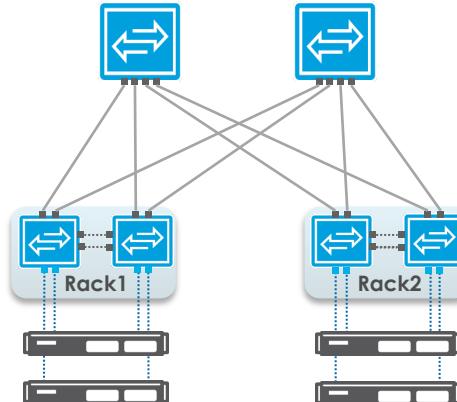
- Static configuration
- Zero Touch Fabric (ZTF)
 - Automatically loads new Switch Light OS and base configuration

Discovering Interfaces

- Dynamically learned as Part of switch discovery

Discovering Links

- Dynamically discovered using LLDP
 - Controller prepares LLDP messages for each interface on every switch
 - Switches forward the given LLDP messages over the specific interface
 - Switches forward any received LLDP messages to the controller
- Receiving LLDP packets from both directions creates a bi-directional link in the fabric
- Controller sets switch expectations
 - Send LLDP packets every 15 seconds ; Notify controller if no LLDP packet received in 35 seconds
- Controller adds link to forwarding path algorithm



BCF IN-DEPTH

Understanding Logical Components

Spine Logical Entity

- Consists of one or more physical switches
- Identified based on “fabric-role” configuration
- Used for interconnecting “Rack” logical entities

Leaf Logical Entity

- Consists of a single switch identified by “fabric-role”
- One-to-one mapping with HW components (Switch)
- Used for connecting devices to Fabric

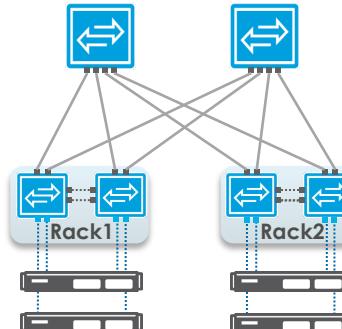
Rack (Leaf Group) Logical Entity

- Consists of one or two leaf switches
- Every leaf switch is member of a “Rack (Leaf Group)”
- Identified based on “Leaf Group” configuration
- Dynamically generated if no configuration
- Used as the destination to deliver traffic

```
switch Spine1
  fabric-role spine
  mac 70:72:CF:C8:F9:25

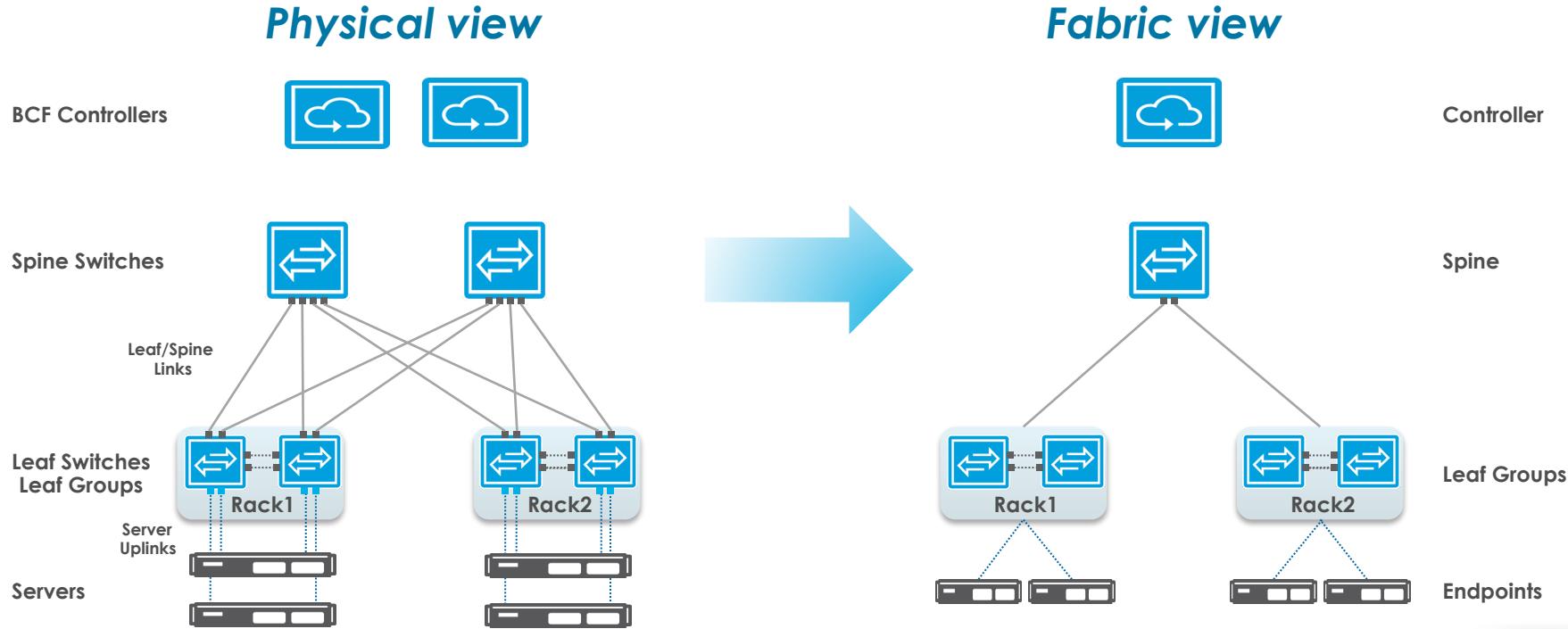
switch R1L1
  fabric-role leaf
  leaf-group Rack1
  mac 70:72:CF:C7:06:BF

switch R1L2
  fabric-role leaf
  leaf-group Rack1
  mac 70:72:CF:C7:00:63
```



BCF IN-DEPTH

Physical View vs. Fabric View

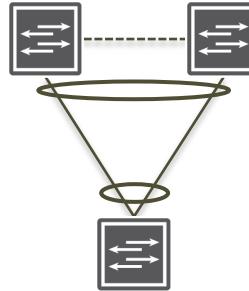


BCF IN-DEPTH

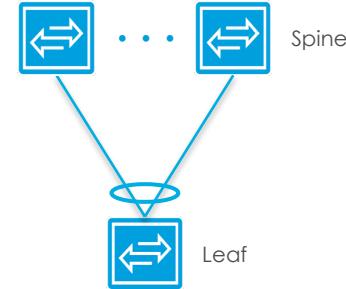
Fabric LAG – Properties



Traditional LAG



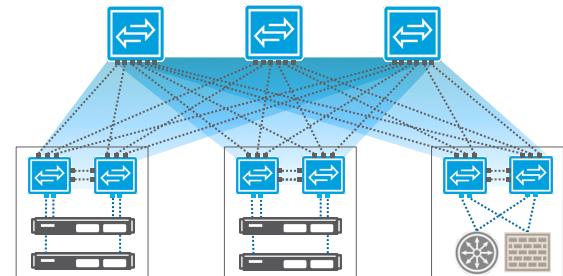
MLAG



BCF Fabric LAG

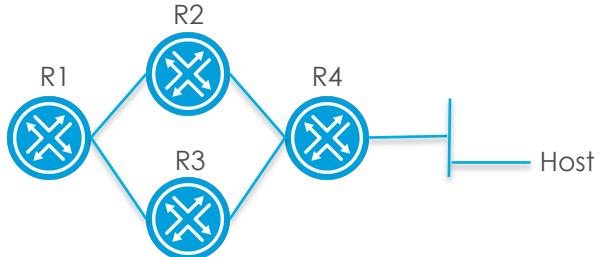
BCF Fabric LAG Properties

- No learning (Learning only performed on edge interfaces of leaf switches)
- No control protocol
- Egress only static LAGs
 - All members of LAG do not need to terminate on same device
- ECMP like forwarding
- Each interface may be a member of multiple LAGs
 - Allows configuration of LAGs for different functions using same interfaces



BCF IN-DEPTH

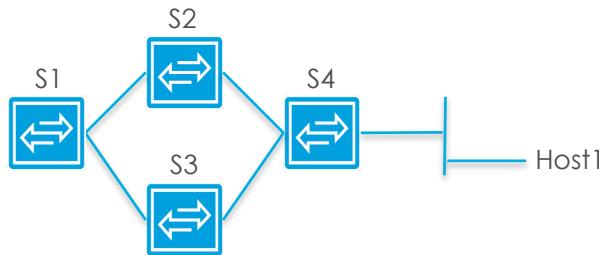
Fabric LAG – ECMP Behavior



R1 can reach Host1 via R2 or R3

Routers

- Routers forward frames using any interface as long as next-hop router has reachability to the destination host
- Routers perform their learning in the control plane (OSPF, BGP, ...)



S1 can only reach Host1 via the switch where it learned Host1's MAC address

Switches

- Switches forward frames using the interface where the switch learned the MAC address for the given destination host
- Switches perform their learning in the data plane (MAC learning)

What if switches do not learn in the data plane?

- Assume that SDN Controller performs this function
- Interfaces are used only for forwarding traffic
- Switches can also perform ECMP-like forwarding

BCF IN-DEPTH

Fabric LAGs

Spine LAG

- On every leaf switch
- Spans all spine switches

Rack 1 LAG

- On every spine switch
- Spans both switches in the leaf group
- On every leaf switch except Rack 1
- On Rack 2 and 3, its members are the same as Spine LAG

Rack 2 LAG

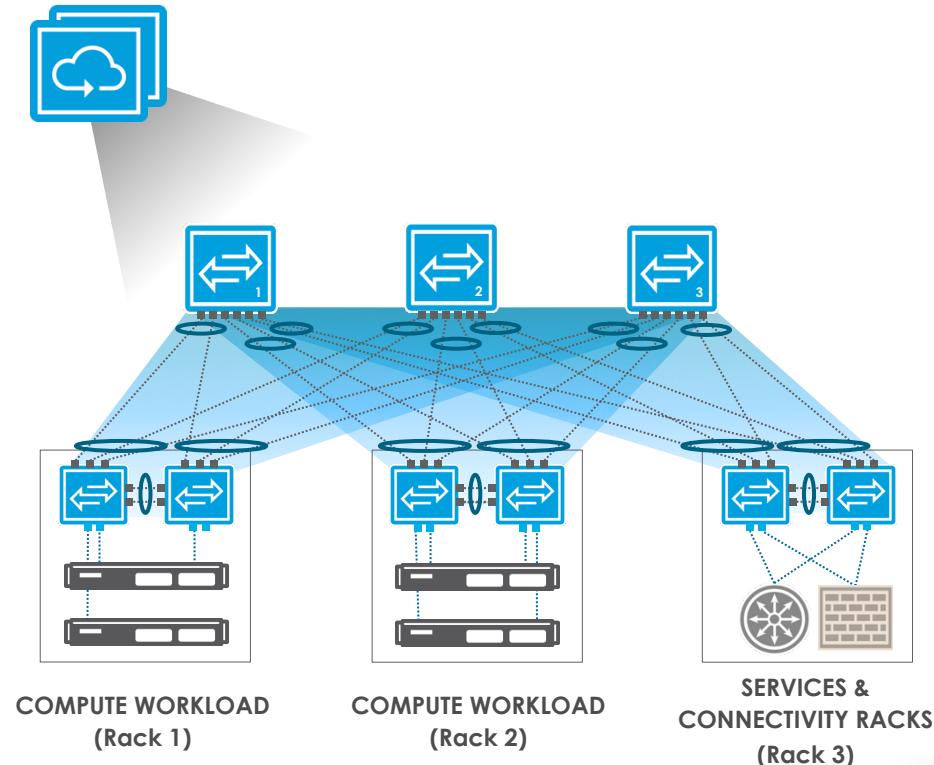
- Same as Rack 1 LAG

Rack 3 LAG

- Same as Rack 1 & 2 LAG

Peer LAG

- Interconnects leaf switches in a Rack



BCF IN-DEPTH

Fabric LAGs

LAG	Description	Host
Rack LAG	<ul style="list-style-type: none">• L2 known unicast forwarding to a given leaf group• There is a Rack LAG for each leaf group in each spine and leaf switch• Consists of all interfaces with connectivity to the destination leaf group	Spine & Leaf
Spine Broadcast LAG	<ul style="list-style-type: none">• L2 BUM forwarding• Consists of all spine facing interfaces where the spine switch has connectivity to all leaf groups	Leaf
Peer LAG	<ul style="list-style-type: none">• L2 unicast and BUM forwarding for workloads connected to a single leaf switch• Contains all interfaces connecting the leaf switches in a leaf group	Leaf
Inter-tenant LAG	<ul style="list-style-type: none">• L3 traffic where next hop is system tenant• Same as Spine Broadcast LAG	Leaf
Any Leaf LAG	<ul style="list-style-type: none">• L3 traffic where destination host is not known• Contains all interfaces to all leaf switches	Spine
Spine LAG	<ul style="list-style-type: none">• Collection of all interfaces connecting to spine switches• Not used for forwarding	Leaf

Questions?

BCF IN-DEPTH

Defining Endpoint

Definition

- Any directly attached device to BCF that sources or receives traffic
- Consists of three or four parameters

Attachment Point

- An interface where the endpoint is attached
- A physical switch interface or interface group (LAG)

Attachment Type

- Encapsulation type/value of the endpoint
- Currently support VLAN and untagged
- Expandable to other encapsulations types in future
 - MPLS, VxLAN, ...

MAC Address

IP Address

L2 Endpoint = Attachment Point +
Attachment Type +
MAC Address

L3 Endpoint = Attachment Point +
Attachment Type +
MAC Address +
IP Address

BCF IN-DEPTH

Defining Endpoint

Attachment Point & Type

- Discovered from configuration
- Provided using member rules

MAC & IP Address

- Configured statically or learned dynamically
- MAC mandatory
- IP optional

Benefits of Static Configuration

- Use alias to reference endpoint
- MAC or IP address never ages out
- Eliminates duplicate or rogue hosts. Same MAC or IP on another interface is ignored

```
tenant T1
segment S1

member interface-group G1 vlan 10
member switch R1L1 interface eth1 vlan untagged
member interface-group any vlan 10
member switch any interface any vlan 20
member switch any interface eth1 vlan 20

endpoint H1
  mac 00:00:01:02:03:04
  ip 10.1.1.1
  attachment-point interface-group G1 vlan 10
endpoint H2
  mac 00:00:05:06:07:08
  ip 10.1.1.2
  attachment-point switch R1L1 interface eth1 vlan untagged
```

The diagram illustrates the structure of a configuration snippet. On the right, two curly braces are positioned to group specific and wildcard rules. The top brace, spanning the first five lines of the configuration, is labeled 'Specific rules'. The bottom brace, spanning the last three lines, is labeled 'Wildcard rules'.

BCF IN-DEPTH

Learning Endpoint – MAC Address

L2 Miss

- Switch sees a frame with unknown MAC address on “Edge” interface
- Packet forwarded to controller
- An entry created in L2 table with pending bit set
 - All future packets with this source MAC address are dropped
 - Entry deleted within 5 seconds or when the MAC is programmed

Controller MAC Address Checks

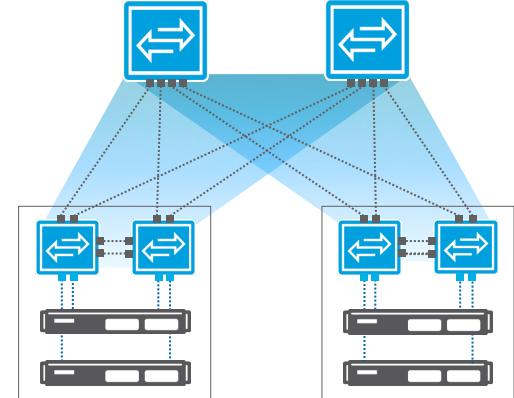
- New host, MAC move, statically configured on another port?

Fabric Update

- Source Leaf-Group switches: End-Point = <MAC, Interface-Group>
 - If end-point is connected to a single switch, peer port is part of the Interface-Group
- All other switches (Spine & Leaf): End-Point = <MAC, Rack LAG>
- Switch expectations set
 - MAC expires in 5 minutes (switches set the source hit bit each time there's a match)

State of Packet

- Controller consumes the packet. This packet does not re-enter the data path.



BCF IN-DEPTH

Learning Endpoint – IP Address

ARP Packet (Request or Reply)

- MAC and IP in payload used to learn IP addresses
- Copy of packet forwarded to controller
 - If “L2 Miss”, packet used to learn MAC and IP at same time
- ARP request packet (broadcast frame) forwarded based on flood table
- ARP request or reply packet (unicast frame) forwarded based on L2 lookup

Controller Checks

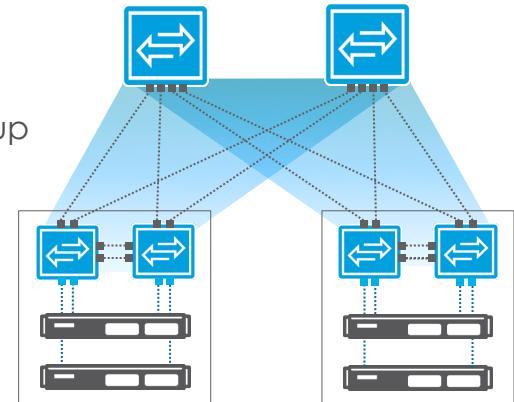
- Packet ignored if MAC and IP already learned

Fabric Update

- Switch host tables updated
- Switch expectations set
 - IP address expires in 5 minutes
 - Source switch's ARP agent to send ARP refresh packets every 4 minutes if needed

State of Packet

- Controller consumes any packet it receives



BCF IN-DEPTH

Epiring Endpoint

Epiring Idle Endpoints

- Attachment point switch periodically checks the L2 table for learned Endpoints
- Notification sent to controller for Endpoints idle at least 300 seconds
 - Only for learned Endpoints (not statically configured Endpoints)
- Controller removes the Endpoint
- Controller updates the attachment point switch and all other switches

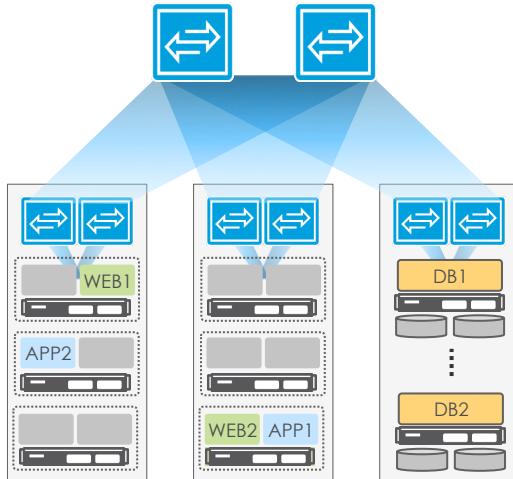
Epiring IP Addresses

- ARP agent sends a unicast ARP every 4 minutes if needed
- If no response, agent sends a second broadcast ARP request 15 seconds later
 - Also handles host migration
- Notification sent to controller if more than 300 seconds since last successful ARP
 - Only for learned IP (not statically configured IP)
- Controller removes the IP from Endpoint
- Controller updates the attachment point switch and all other switches

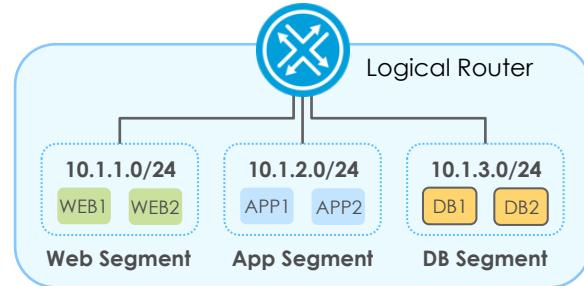
Questions?

BCF IN-DEPTH

Segmentation and Routing



```
tenant Dev
logical-router
  interface segment Web
    ip address 10.1.1.1/24
  interface segment App
    ip address 10.1.2.1/24
  interface segment DB
    ip address 10.1.3.1/24
segment Web
...
segment App
...
segment DB
...
```



Segments and Routes completely separated from physical infrastructure

BCF IN-DEPTH

Segmentation and Routing

Segmentation

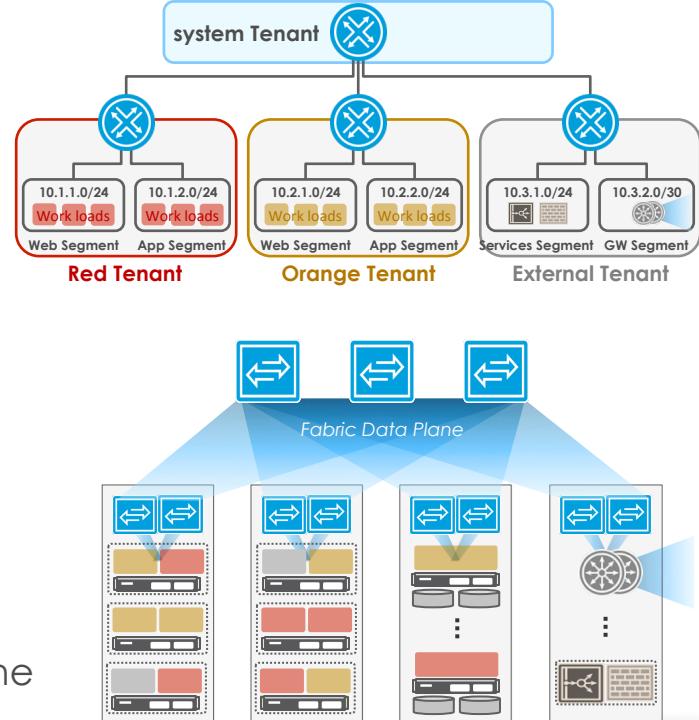
- Based on attachment points and attachment types
- Internal VLAN assigned to each segment
- VLAN used by host only locally significant
- VLAN translation table maps internal VLAN to external VLAN
- Controller programs each switch with MAC tables
 - L2 table – contains learned MACs
 - VLAN Translation table – contains the mapping between internal and external VLANs

BCF IN-DEPTH

Segmentation and Routing

Routing

- Tenant routers located on leaf switches
 - Every leaf switch contains a router for every tenant
- System router located on spine switches
- Fully distributed, thus endpoints of a segment may be anywhere in the fabric
- Route learning based on static routes and BGP
 - System router supports static routing only
- Based on logical entities (tenants & segments) not physical entities (rack or switch)
- Inter-tenant routing requires at least one spine
 - Even if source and destination is on the same rack
- Inter-segment (intra-tenant) works even without a spine
- No difference in routing from traditional routing process



Questions?

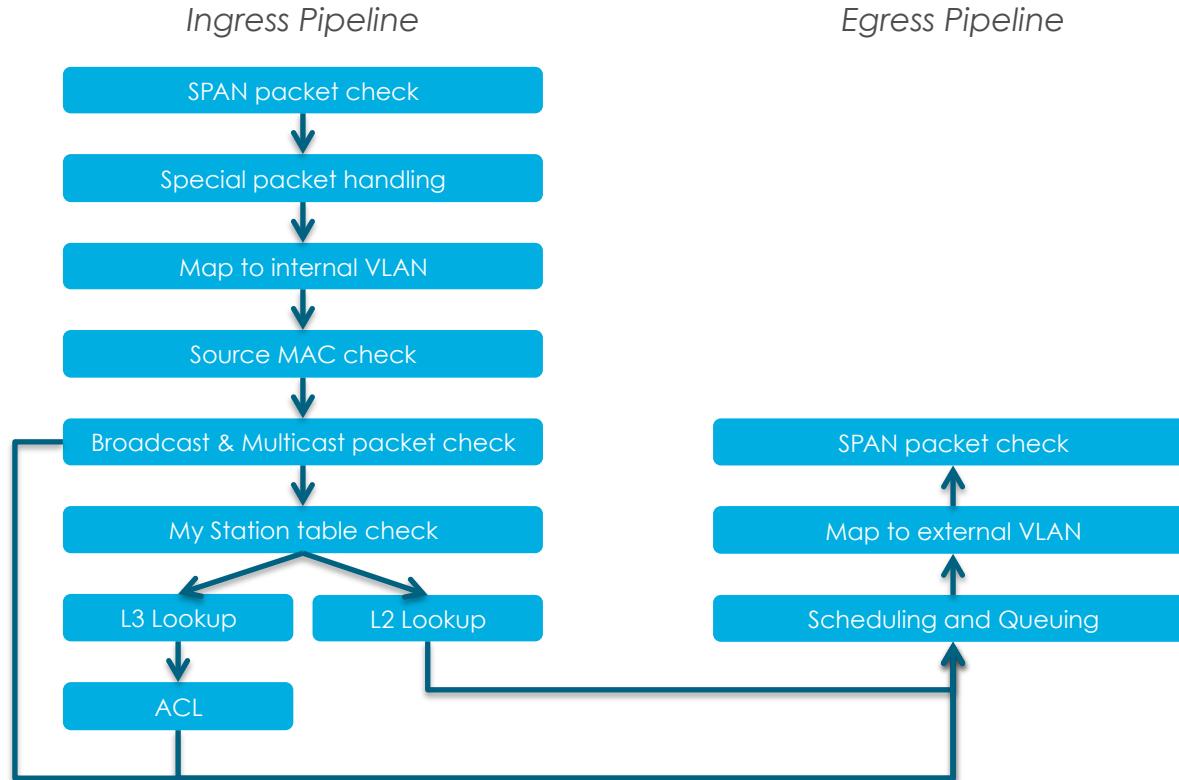
BCF IN-DEPTH

Packet Forwarding

- Forwarding Pipeline
- Intra-segment Packet Walk
- Inter-segment Packet Walk
- Inter-tenant Packet Walk
- Broadcast Packet Walk

BCF IN-DEPTH

Forwarding Pipeline



BCF IN-DEPTH

Packet Walk – Intra-segment

App1 → App2

- Same segment
- Same leaf group

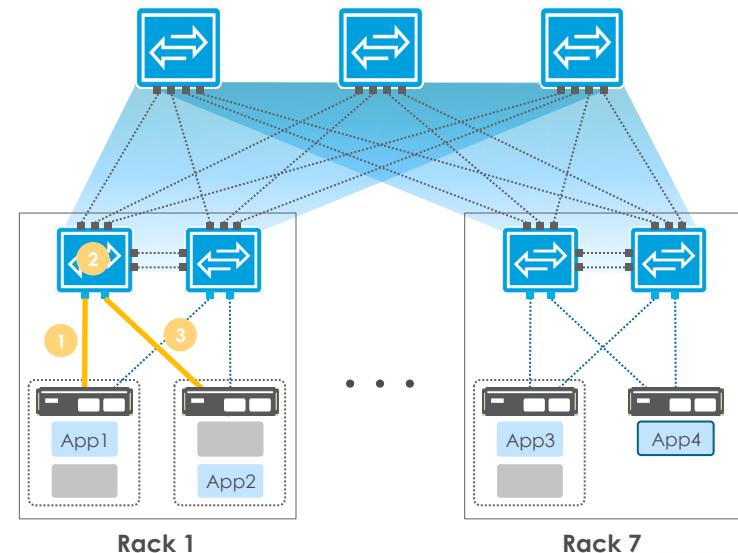
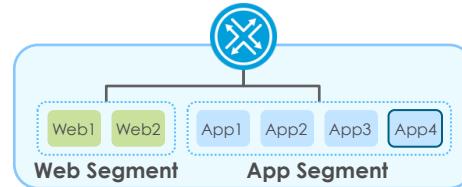
Assumes

- Source MAC has been learned
- ARPs have been resolved

1 App1 sends the packet to a leaf switch

2 Leaf switch performs L2 look-up on destination MAC address in the segment MAC table

3 Leaf switch forwards it on the LAG for App2



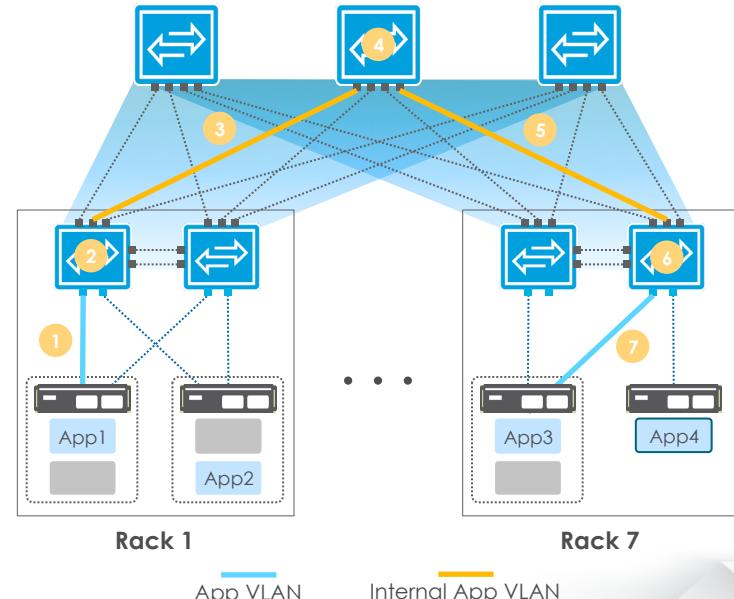
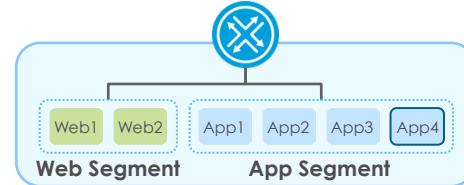
BCF IN-DEPTH

Packet Walk – Intra-segment

App1 → App3

- Same segment
- Different leaf group

- 1 App1 sends the packet to a leaf switch
- 2 Leaf switch performs L2 look-up on destination MAC address
- 3 Leaf switch forwards it on Rack 7 fabric LAG
VLAN: Internal App
- 4 Spine performs L2 look-up on destination MAC address
- 5 Spine forwards it on Rack 7 fabric LAG
VLAN: Internal App
- 6 Leaf switch performs L2 look-up on destination MAC address
- 7 Leaf switch forwards it on the LAG for App3
VLAN: App segment



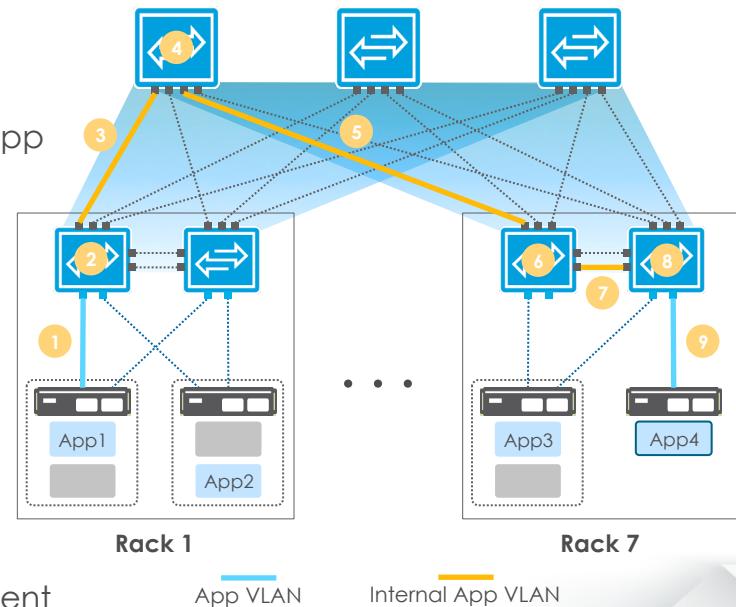
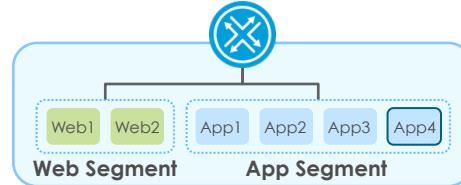
BCF IN-DEPTH

Packet Walk – Intra-segment

App1 → App4

- Same segment; Different leaf group;
Connected to a single leaf switch

- 1 App1 sends the packet to a leaf switch
- 2 Leaf switch performs L2 look-up on destination MAC address
- 3 Leaf switch forwards it on Rack 7 fabric LAG. VLAN: Internal App
- 4 Spine performs L2 look-up on destination MAC address
- 5 Spine forwards it on Rack 7 fabric LAG. VLAN: Internal App
- 6 Leaf switch performs L2 look-up on destination MAC address
- 7 Leaf switch forwards it on the LAG for App4 which contains the peer-group fabric LAG. VLAN: Internal App
- 8 Leaf switch performs L2 look-up on destination MAC address
- 9 Leaf switch forwards it on the LAG for App4. VLAN: App segment



BCF IN-DEPTH

Packet Walk – Intra-segment Forwarding (1)

Ingress Leaf Switch

- Packet is mapped to an internal VLAN for the given segment
- Source MAC check passes (If not, MAC Address learning is initiated)
- Destination MAC is checked in My-Station table
 - If there's a match, packet is processed using L3 lookup (Packet is destined for the router)
 - If there's no match, packet is processed using L2 lookup
- L2 lookup table provides the destination RACK LAG or Interface Group
- Leaf forwards it to Spine on Rack LAG using the internal VLAN for the segment

Spine Switch

- Skips the source MAC check (no learning performed on Fabric Links)
- Performs My-Station table check (as no match) and forwards it for L2 lookup
- Performs L2 lookup using the VRF for the internal VLAN and forwards it to the Rack LAG
- No changes made to the frame – L2 forwarding only

BCF IN-DEPTH

Packet Walk – Intra-segment Forwarding (2)

Egress Leaf Switch

- Skips source MAC check (again no learning on Fabric Links)
- Performs My-Station table check and directs the packet for L2 processing
- Performs L2 lookup using the VRF for the internal VLAN ID and forwards it to the destination Interface Group
- All destinations are Interface Groups including hosts with a single connection
 - If host has a single connection and it is connected to the other leaf switch, destination interface group consists of inter-switch links
- Packet is mapped from the internal VLAN to the host-facing VLAN

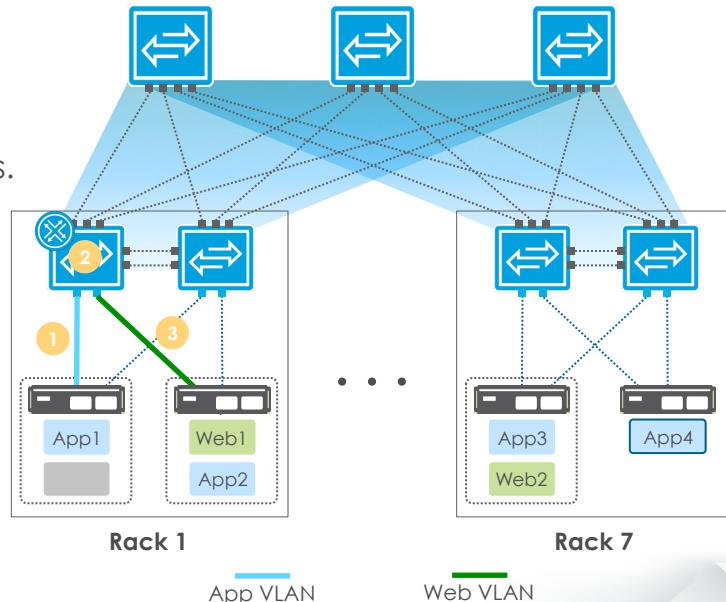
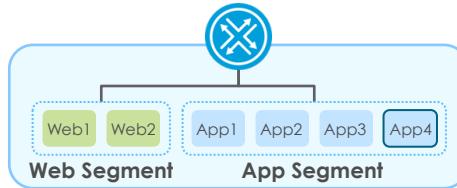
BCF IN-DEPTH

Packet Walk – Inter-segment

App1 → Web1

- Same tenant
- Same leaf group

- 1 App1 sends the packet to a leaf switch (default gateway)
- 2 Destination MAC address matches logical router MAC address.
Leaf switch performs L3 look-up on destination IP address.
- 3 Leaf switch forwards it on the LAG for Web1
DMAC: Web1
VLAN: Web segment



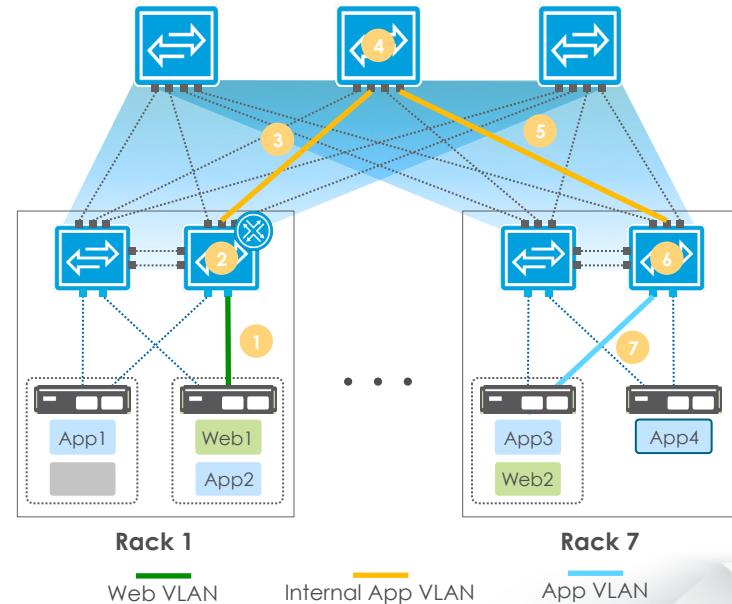
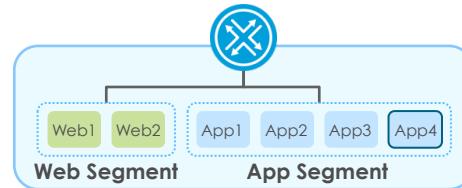
BCF IN-DEPTH

Packet Walk – Inter-segment

Web1 → App3

- Same tenant
 - Different leaf group

- 1 Web1 sends the packet to a leaf switch (default gateway)
 - 2 Leaf switch performs L3 look-up on destination IP address
 - 3 Leaf switch forwards it on Rack 7 fabric LAG
DMAC: App3 VLAN: Internal App
 - 4 Spine performs L2 look-up on destination MAC address
 - 5 Spine forwards it on Rack 7 fabric LAG
DMAC: App3 VLAN: Internal App
 - 6 Leaf switch performs L2 look-up on destination MAC address
 - 7 Leaf switch forwards it on the LAG for App3
DMAC: App3 VLAN: App segment



BCF IN-DEPTH

Packet Walk – Inter-segment Forwarding

Ingress Leaf Switch

- Packet mapped to an internal VLAN for the given segment
- Source MAC check passes
- Destination MAC matches the logical-router MAC in My-Station table
- L3 lookup table (Host table) provides the destination RACK LAG or Interface Group
 - If no match, IP Address learning initiated
- Packet forwarded to Spine using RACK LAG with the internal VLAN of destination segment

Spine Switch

- Performs L2 lookup and forwards to the destination Leaf-Group via the RACK LAG

Egress Leaf Switch

- Performs L2 lookup and forwards it to the destination Interface Group
- May forward to peer switch to reach any single connection hosts
- Packet mapped from the internal VLAN to the host-facing VLAN

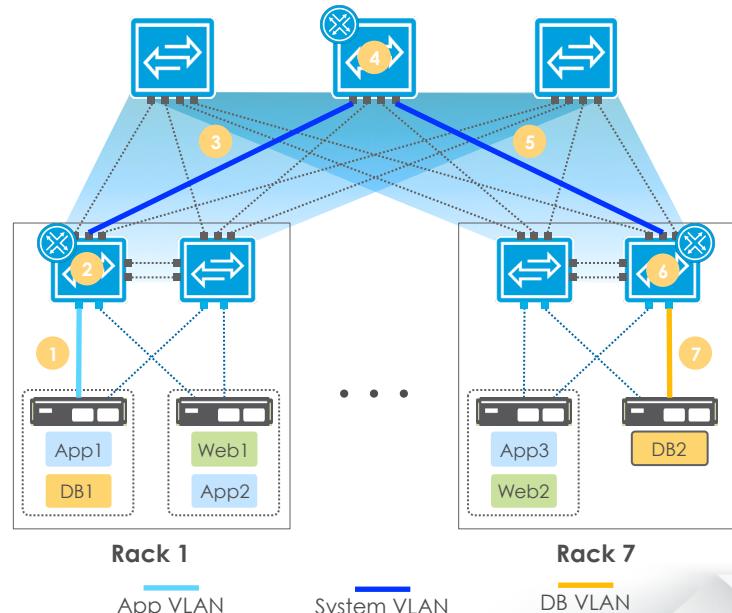
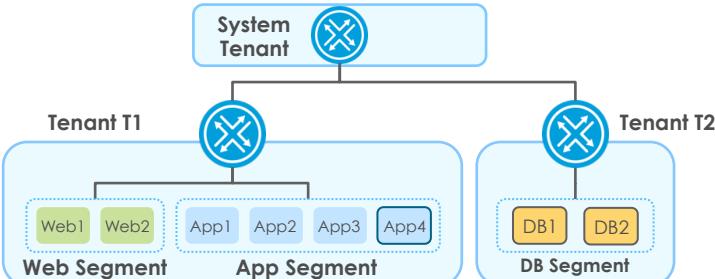
BCF IN-DEPTH

Packet Walk – Inter-tenant

App1 → DB2

- Different tenant
- Different leaf group

- 1 App1 sends the packet to a leaf switch (default gateway)
- 2 Leaf switch performs L3 look-up on destination IP address
- 3 Leaf switch forwards it on Inter-tenant fabric LAG
DMAC: System Router VLAN: 4094
- 4 Spine performs L3 look-up on destination IP address
- 5 Spine forwards it on Rack 7 fabric LAG
DMAC: Tenant Router VLAN: 4094
- 6 Leaf switch performs L3 look-up on destination IP address
- 7 Leaf switch forwards it on the LAG for DB2
DMAC: DB2 VLAN: DB segment



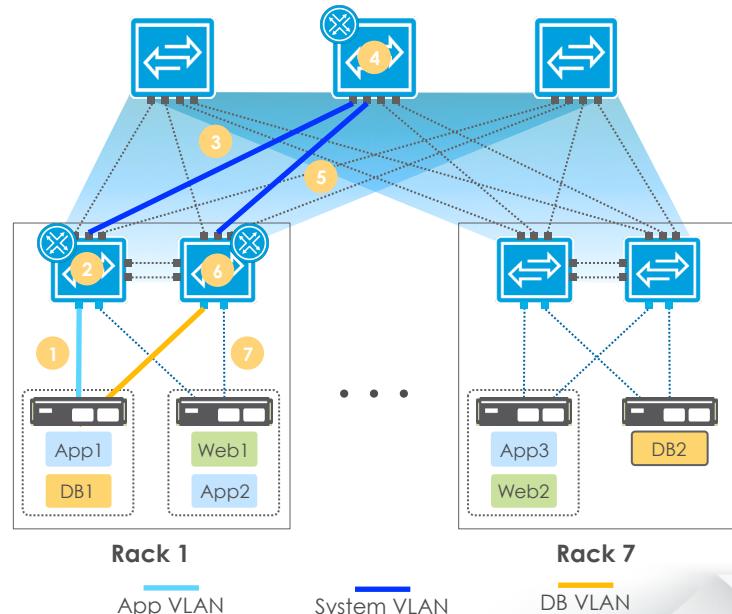
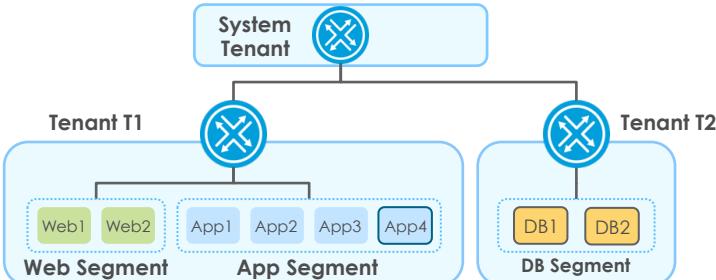
BCF IN-DEPTH

Packet Walk – Inter-tenant

App1 → DB1

- Different tenant
- Same leaf group

- 1 App1 sends the packet to a leaf switch (default gateway)
- 2 Leaf switch performs L3 look-up on destination IP address
- 3 Leaf switch forwards it on Inter-tenant fabric LAG
DMAC: System Router VLAN: 4094
- 4 Spine performs L3 look-up on destination IP address
- 5 Spine forwards it on Rack 1 fabric LAG
DMAC: Tenant Router VLAN: 4094
- 6 Leaf switch performs L3 look-up on destination IP address
- 7 Leaf switch forwards it on the LAG for DB1
DMAC: DB1 VLAN: DB segment



BCF IN-DEPTH

Packet Walk – Inter-tenant Forwarding

Ingress Leaf Switch

- Packet mapped to an internal VLAN for the given segment
- Source MAC check passes
- Destination MAC matches the logical-router MAC in My-Station table
- Destination IP matches the default route in L3 CIDR table
- Packet forwarded to Spine via the Inter-Tenant LAG using System Router VLAN 4094
 - Destination MAC address replaced with System Router MAC address

Spine Switch

- Performs L3 lookup and forwards to the destination Leaf-Group via the RACK LAG
 - Destination MAC address replaced with Tenant Router MAC address

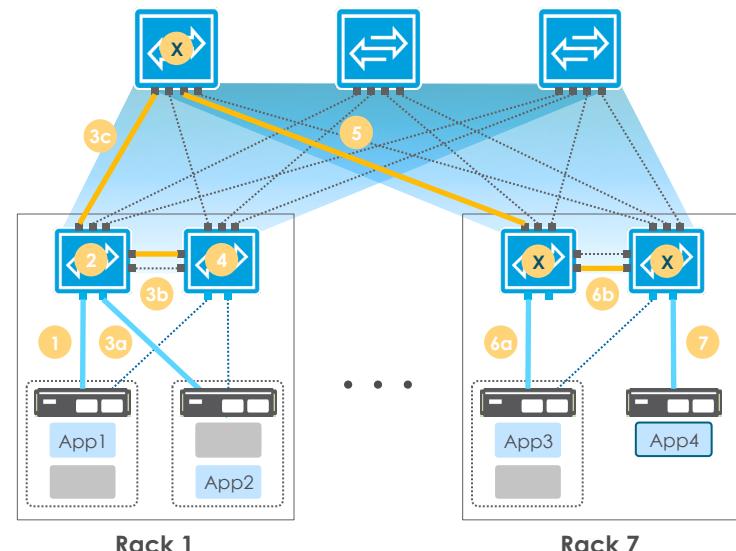
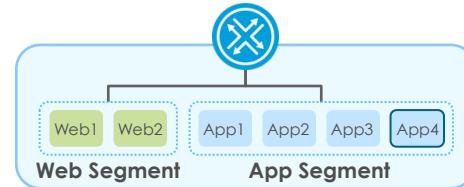
Egress Leaf Switch

- Performs L3 lookup in the destination VRF and forwards to the destination Interface Group
- Packet mapped from the System Router VLAN (4094) to the host-facing VLAN

BCF IN-DEPTH

Packet Walk – Broadcast Packets

- 1 App1 sends the packet to a leaf switch
- 2 Leaf switch identifies the broadcast packet
- 3a Leaf switch forwards it to other hosts on the same segment
- 3b Leaf switch forwards it to its peer switch
- 3c Leaf switch forwards it to the spine
- 4 Peer switch identifies the broadcast packet. Forwards to Any single connected hosts. Does not forward to spine
- 5 Spine forwards it to other leaf groups
- 6a Leaf switch forwards it to other hosts on the same segment
- 6b Leaf switch forwards it to its peer switch
- 7 Peer switch forwards it to any single connected hosts
- X Switch identifies the broadcast packets



BCF IN-DEPTH

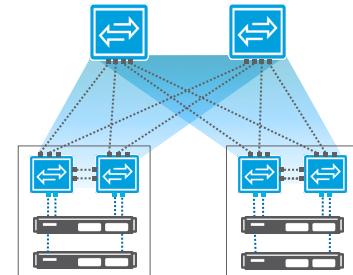
Packet Walk – Broadcast Flooding

Ingress Leaf Switch

- Packet mapped to an Internal VLAN. Source MAC check passes
- Broadcast packet is identified prior to My Station table check
- Flood table checked for flooding destinations (LAGs where packet will be replicated)
 - Packet forwarded to all interfaces configured with the same segment
 - Packet forwarded to Peer via the Peer LAG
 - Packet forwarded to Spine via the Spine-Broadcast LAG

Peer Leaf Switch

- Broadcast packet check identifies the broadcast packet
- Flood table checked to determine flooding destinations
 - Packet forwarded to any host with a single connection on same segment
 - Packet not forwarded to all hosts on the segment
 - Packet not forwarded to Spine



BCF IN-DEPTH

Packet Walk – Broadcast Flooding

Spine Switch

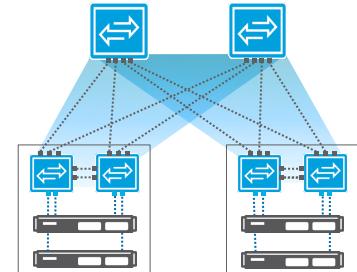
- Broadcast packet check identifies the broadcast packet
- Flood table checked to determine flooding destinations
 - Packet forwarded to all Leaf-Groups using RACK LAGs where segment exists

Egress Leaf Switches

- Broadcast packet check identifies the broadcast packet
- Flood table checked to determine flooding destinations
 - Packet forwarded to all interfaces configured with same segment
 - Packet forwarded to Peer via the Peer LAG

Egress Leaf Peer Switch

- Broadcast packet check identifies the broadcast packet
- Flood table checked to determine flooding destinations
 - Packet forwarded to any host with a single connection on same segment



Questions?

Module 11 – Contacting Support

CONTACTING TECHNICAL SUPPORT

Opening a Support Case

Opening a support case

- Email: Send email to support@bigswitch.com
- Telephone: Call 800-653-0565, option 3
- Web Portal: <http://www.bigswitch.com/support>
 1. Login to Support portal
 2. Select "Cases"
 3. Select "Create New"

No web portal login id

- Contact sales representative

CONTACTING TECHNICAL SUPPORT

Minimum Information

Problem description

- In as much detail as possible
 - What triggered the issue (if identifiable)
 - Symptoms of the issue
- For installation issues, also include
 - HW make and model
 - ONIE and CPLD version of installed SW

Support bundle

- CLI `controller# support`
- GUI Menu entry: Visibility -> Support Bundle

CONTACTING TECHNICAL SUPPORT

Uploading Files (Support Bundle)

1. ftp ftp.bigswitch.com
2. cd public
3. mkdir <dir-name>
4. cd <dir-name>
5. hash
6. put <file-name>
7. bye

CONTACTING TECHNICAL SUPPORT

Web Portal

The screenshot shows the 'Case Edit' screen for creating a new case. The interface is divided into several sections:

- Search:** Includes a search bar with dropdown options for 'Search All' or 'Go!' and a link to 'Advanced Search...'. There is also a 'Create New...' button.
- Recent Items:** A list of recent items with icons: 00001054, SwitchLight-Ins..., BigTap-3.0.1-Sy..., bigtap-3.0.1-up..., bigtap-3.0.1-20..., bigtap-3.0.1-us..., switchlight-2.0..., and switchlight-2.0...
- Case Edit - New Case:** The main editing area with tabs for 'Case Edit' (selected), 'Submit', 'Submit & Add Attachment', and 'Cancel'.
- Case Information:** Fields for 'Case Owner' (Customer Test) and 'Contact Name' (Customer Test). A note indicates that 'Case Owner' is required information.
- Additional Information:** Fields for 'Status' (New), 'Priority' (Priority 3 - Medium), 'Type' (None), and 'Deployment Status' (None).
- Description Information:** Fields for 'Subject' (empty text input), 'Description' (empty text area), 'Product Version' (None), and 'Switch Light Version' (None).
- Optional:** A checkbox labeled 'Send notification email to contact' which is checked.

At the bottom of the form are three buttons: 'Submit', 'Submit & Add Attachment', and 'Cancel'.

[Home](#) | [Cases](#) | [Libraries](#) | [Content](#)

Questions?

Feedback Survey

Thank You