53-1004145-01
12 February 2016

Network OS IP Fabrics

Configuration Guide

Supporting Network OS 7.0.0

BROCADE

Contents

Preface

Document conventions

The document conventions describe text formatting conventions, command syntax conventions, and important notice formats used in Brocade technical documentation.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used in the flow of the text to highlight specific words or phrases.

Format	Description
bold text	Identifies command names
	Identifies keywords and operands
	Identifies the names of user-manipulated GUI elements
	Identifies text to enter at the GUI
italic text	Identifies emphasis
	Identifies variables
	Identifies document titles
`Courier font`	Identifies CLI output
	Identifies command syntax examples

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
value	In Fibre Channel products, a fixed value provided as input to a command option is printed in plain text, for example, **--show** WWN.

Convention	Description
[ ]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ **x** | **y** | **z** }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options. In Fibre Channel products, square brackets may be used instead for this purpose.
x | **y**	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a “soft” line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Brocade resources

Visit the Brocade website to locate related documentation for your product and additional Brocade resources.

You can download additional publications supporting your product at www.brocade.com. Select the Brocade Products tab to locate your product, then click the Brocade product name or image to open the individual product page. The user manuals are available in the resources module at the bottom of the page under the Documentation category.

To get up-to-the-minute information on Brocade products and resources, go to MyBrocade. You can register at no cost to obtain a user ID and password.

Release notes are available on MyBrocade under Product Downloads.

White papers, online demonstrations, and data sheets are available through the Brocade website.

Contacting Brocade Technical Support

As a Brocade customer, you can contact Brocade Technical Support 24x7 online, by telephone, or by e-mail. Brocade OEM customers contact their OEM/Solutions provider.

Brocade customers

For product support information and the latest information on contacting the Technical Assistance Center, go to http://www.brocade.com/services-support/index.html.

If you have purchased Brocade product support directly from Brocade, use one of the following methods to contact the Brocade Technical Assistance Center 24x7.

Online	Telephone	E-mail
Preferred method of contact for non-urgent issues: • My Cases through MyBrocade • Software downloads and licensing tools • Knowledge Base	Required for Sev 1-Critical and Sev 2-High issues: • Continental US: 1-800-752-8061 • Europe, Middle East, Africa, and Asia Pacific: +800-AT FIBREE (+800 28 34 27 33) • For areas unable to access toll free number: +1-408-333-6061 • Toll-free numbers are available in many countries.	support@brocade.com Please include: • Problem summary • Serial number • Installation details • Environment description

Brocade OEM customers

If you have purchased Brocade product support from a Brocade OEM/Solution Provider, contact your OEM/Solution Provider for all of your product support needs.

- OEM/Solution Providers are trained and certified by Brocade to support Brocade® products.
- Brocade provides backline support for issues that cannot be resolved by the OEM/Solution Provider.

- Brocade Supplemental Support augments your existing OEM support contract, providing direct access to Brocade expertise. For more information, contact Brocade or your OEM.
- For questions regarding service levels and response times, contact your OEM/Solution Provider.

Document feedback

To send feedback and report errors in the documentation you can use the feedback form posted with the document or you can e-mail the documentation team.

Quality is our first concern at Brocade and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you. You can provide feedback in two ways:

- Through the online feedback form in the HTML documents posted on www.brocade.com.
- By sending your feedback to documentation@brocade.com.

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

About this document

Supported hardware and software

This section is specific to support for IP Fabrics. In those instances in which procedures or parts of procedures documented here apply to some switches but not to others, this guide identifies exactly which switches are supported for certain features and which are not.

Although many different software and hardware configurations are tested and supported by Brocade Communications Systems, Inc. for Network OS, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release of Network OS:

- Brocade VDX 6740
  - Brocade VDX 6740-48
  - Brocade VDX 6740-64
- Brocade VDX 6740T
  - Brocade VDX 6740T-48
  - Brocade VDX 6740T-64
  - Brocade VDX 6740T-1G
- Brocade VDX 6940-36Q
- Brocade VDX 6940-144S
- Brocade VDX 8770
  - Brocade VDX 8770-4
  - Brocade VDX 8770-8

To obtain information about a Network OS version other than this release, refer to the documentation specific to that version.

Using the Network OS CLI

For complete instructions and support for using the Network OS command line interface (CLI), refer to the *Network OS Command Reference*.

What's new in this document

This document is released in conjunction with Network OS 7.0.0. This is the first release of this document.

For complete information, refer to the *Network OS Release Notes*.

Overview of IP Fabrics

Overview

Data center networking architectures have evolved with the changing requirements of the modern data center and cloud environments.

The traffic patterns in data center networks are changing rapidly from north-south to east-west. Cloud applications are often multi-tiered and hosted at different endpoints connected to the network. The communication between these application tiers is a major contributor to the overall traffic in a data center.

These traffic patterns are the primary reasons that data center networks need to evolve into scale-out architectures. Scale-out architectures are built to maximize the throughput for east-west traffic. In addition to providing high east-west throughput, scale-out architectures provide a mechanism to add capacity to the network horizontally, without reducing the provisioned capacity between the existing endpoints. The de-facto industry standard for implementing scale-out architectures is using Clos topologies. These topologies include the 3-stage folded Clos (or leaf-spine topology) and the optimized 5-stage folded Clos. These topologies are described in the next section.

Brocade IP fabric provides a Layer 3 Clos deployment for data center sites. With Brocade IP fabric, all the links in the Clos topology are Layer 3 links. The Brocade IP fabric includes the networking architecture, the protocols used to build the network, turnkey automation features used to provision, manage, and monitor the networking infrastructure and the hardware differentiation with Brocade VDX switches.

Because the infrastructure is built on IP, advantages like loop-free communication using industry-standard routing protocols, ECMP, very high solution scale, and standards-based interoperablility are leveraged.

These are some of the key benefits of deploying a data center site with Brocade IP fabrics:

- Highly scalable infrastructure: Because the Clos topology is built using IP protocols, the scale of the infrastructure is very high. These port and rack scales are documented with descriptions of the Brocade IP fabric deployment topologies.
- Standards-based and interoperable protocols: The Brocade IP fabric is built using industry-standard protocols like the Border Gateway Protocol (BGP) and Open Shortest Path First (OSPF). These protocols are well understood and provide a solid foundation for a highly scalable solution. In addition, industry-standard overlay control and data plane protocols like BGP EVPN and Virtual Extensible Local Area Network (VXLAN) are used to extend Layer 2 domain and extend tenancy domains by enabling Layer 2 communications and VM mobility.
- Active-active vLAG pairs: By supporting vLAG pairs on leaf switches, dual-homing of the networking endpoints are supported. This provides higher redundancy. Also, because the links are active-active, vLAG pairs provide higher throughput to the endpoints. vLAG pairs are supported for all 10 GbE, 40 GbE, and 100 GbE interface speeds, and up to 32 links can participate in a vLAG.

- Layer 2 extensions: In order to enable Layer 2 domain extension across the Layer 3 infrastructure, VXLAN encapsulation is leveraged. The use of VXLAN provides a very large number of Layer 2 domains to support large-scale multitenancy over the infrastructure. In addition, Brocade BGP EVPN network virtualization provides the control plane for the VXLAN, enabling automatic VTEP discovery and control-plane learning of remote MAC addresses and MAC-IP bindings, thus eliminating broadcast, unknown unicast, and multicast (BUM) traffic. (For details, refer to Controllerless Network Virtualization with BGP EVPN on page 17.)
- Multitenancy at Layers 2 and 3: Brocade IP fabric provides multitenancy at Layers 2 and 3, enabling traffic isolation and segmentation across the fabric. Layer 2 multitenancy allows an extended range of up to 8000 Layer 2 domains to exist at each ToR switch, while isolating overlapping 802.1q tenant networks into separate Layer 2 domains. Layer 3 multitenancy using VRFs, multi- VRF routing protocols, and BGP EVPN allows large-scale Layer 3 multitenancy. Specifically, Brocade BGP EVPN Network Virtualization leverages BGP EVPN to provide a control plane for MAC address learning and VRF routing for tenant prefixes and host routes, which reduces BUM traffic and optimizes the traffic patterns in the network.
- Support for unnumbered interfaces: Using Brocade Network OS support for IP unnumbered interfaces, only one IP address per switch is required to configure the routing protocol peering. This significantly reduces the planning and use of IP addresses and simplifies operations.
- Turnkey automation: Brocade automated provisioning dramatically reduces the deployment time of network devices and network virtualization. Prepackaged, server-based automation scripts provision Brocade IP fabric devices for service with minimal effort.
- Programmable automation: Brocade server-based automation provides support for common industry automation tools such as Python Ansible, Puppet, and YANG model-based REST and NETCONF APIs. Prepackaged PyNOS scripting library and editable automation scripts execute predefined provisioning tasks, while allowing customization for addressing unique requirements to meet technical or business objectives when the enterprise is ready.
- Ecosystem integration: The Brocade IP fabric integrates with leading industry solutions and products like VMware vSphere, NSX, and vRealize. Cloud orchestration and control are provided through OpenStack and OpenDaylight-based Brocade SDN Controller support.

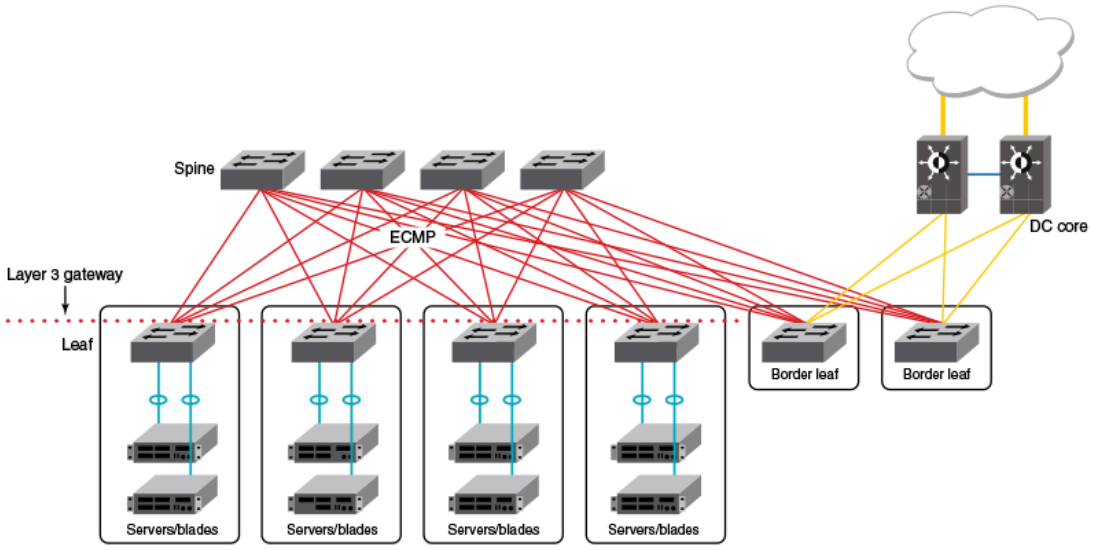
NOTE
For supported features, refer to the Brocade IP Fabrics Data Sheet.

Physical topologies and scale

The basic IP Fabrics physical topologies represent the underlay network and offer different scaling factors.

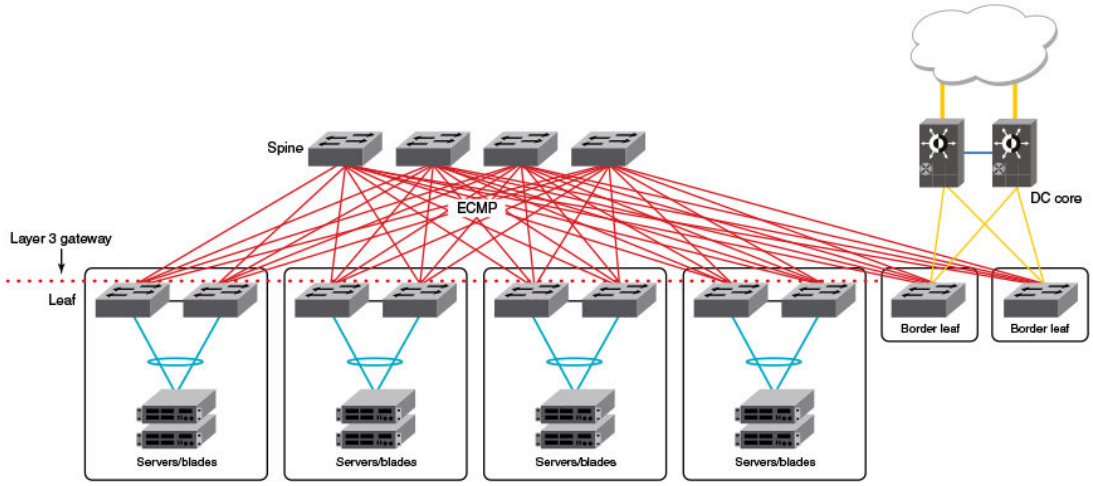
The following topology illustrates a 3-stage folded Clos topology with leafs and spines, with connectivity to the DC core through border leaf nodes. These nodes provide external connectivity to the data center fabric and also provide connectivity to network services such as firewalls and load balancers.

FIGURE 1 A 3-stage folded Clos topology



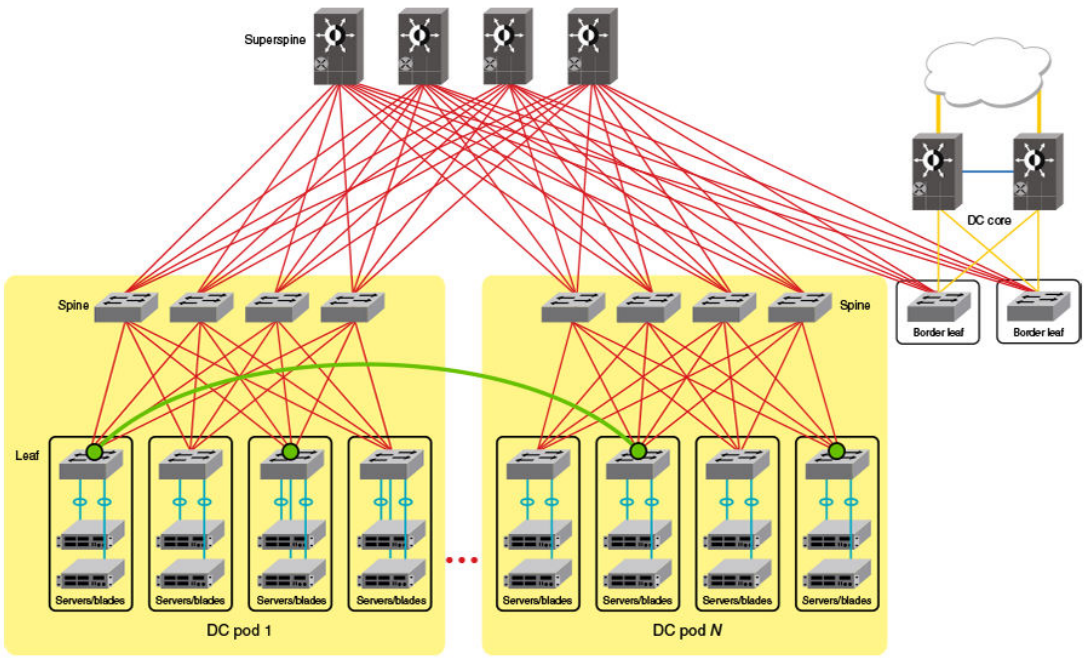
The following topology is similar to the above topology, but with support for optional high availability (HA) provided through redundant servers or blades dual-homed by means of vLAGs. (Refer to Multihoming on page 15.)

FIGURE 2 A 3-stage folded Clos topology with redundant servers or blades



The following topology uses superspine nodes to interconnect multiple data center pods. Here the border leaf nodes connect directly to the superspine nodes.

FIGURE 3 Optimized 5-stage folded Clos topology



NOTE
For design considerations and scaling details, refer to the Brocade Data Center Fabric Architectures white paper. For details of BGP underlay topologies, refer to Controllerless Network Virtualization with BGP EVPN on page 17.

IP unnumbered

The IP unnumbered feature reduces the number of IPv4 or IPv6 addresses required within an IP Fabric, simplifying configuration and maintenance.

With standard /31 addressing, an IP Fabric with four spine nodes and 36 leaf nodes would require 144 links, resulting in 288 addresses. In addition, 40 loopback interfaces are required, for a total of 328 addresses. The solution is to use the IP unnumbered feature, whereby an IP address is borrowed from a "donor" interface. This is configured by means of the **ip unnumbered** or the **ipv6 unnumbered** command on a loopback interface and is applied to a physical port.

This reduces the burden on the hardware tables, and in the IP Fabric deployment example simplifies the IP address assignment in the fabric. This reduces the number of required addresses to only the 40 /32 addresses required by the loopbacks. The IP addresses and MAC addresses of neighbors are discovered by means of LLDP.

For an example of how to configure this feature, refer to Configuring the interfaces on page 40.

NOTE
ARP and Neighbor Discovery (ND) resolution is disabled on unnumbered interfaces. Refer to the Controllerless Network Virtualization with BGP EVPN on page 17 chapter for the role of these features.

Multihoming

Multihoming is an optional feature that supports high availability.

When servers must be multihomed (specifically, dual homed; see Physical topologies and scale on page 12) to more than one leaf switch in an IP Fabric (as for optional HA support), the leaf switches must be configured as a vLAG pair. In this case, (1) RBridge IDs must be unique for each node in the pair, in order to form the local VCS Fabric; (2) the same loopback addresses must be configured on each node to support distributed VTEP if required; (3) both nodes must have the same VCS ID; and (4) a unique router ID (configured by means of the **ip router-id** command) must be configured on each RBridge, so that each RBridge can associate unique route identifiers.

NOTE
For the details of vLAG configuration, refer to the "Link Aggregation" chapter in the *Network OS Layer 2 Switching Configuration Guide*.

Note the following considerations and limitations for this scenario:

- A pair of switches to which other servers create dual-homed connections forms a vLAG pair.
- A VXLAN tunnel is not established between the nodes in a vLAG pair.
- Distributed VXLAN gateway functionality is used to extend VXLAN tunnels to other leaf nodes.
- The Fabric-Virtual-Gateway and VRRP-E features are options for providing redundant gateway functionality.
- If two leaf nodes are connected by means of an external Layer 2 switch, this can result in a multihoming situation for hosts connected to that switch. This topology is not supported.

When two leaf nodes such as those shown above operate in a vLAG pair, they form independent BGP sessions with other routers, including other nodes in the same IP Fabric. It is recommended that the same **local-as** value be configured for all switches belonging to the same vLAG pair. For leaf switches to accept BGP updates, **neighbor allowas-in** must be configured, to disable the AS_PATH check function from rejecting routes that contain the recipient BGP speaker's AS number (ASN).

As noted above, all switches in the vLAG pair must be configured with the same loopback VTEP IP address. This ensures that VTEP discovery does not form a VXLAN tunnel between the two nodes. (The tunnel source and destination IP addresses are the same.) This supports redundancy for the tunnels, as provided in the previous release.

Additional considerations and limitations for IP Fabrics

This section lists considerations and limitations for this feature not addressed elsewhere.

- IP Fabrics do not support fabric cluster mode.
- An IP Fabric can include a two-node vLAG pair or a standalone switch in logical chassis cluster mode.

- With the exception of dual homing and resulting vLAG pair, TRILL is not supported.
- It may be necessary to change an RBridge ID from the default value. Note the following.

When two or more VDX switches are physically connected, each switch, by default, has an RBridge ID of 1 and a VCS ID of 1. The switches will not try to form a vLAG pair until they have different RBridge IDs. Also by default, each switch is a standalone VCS Fabric in logical chassis cluster mode. Because each switch has the same RBridge ID within the same VCS ID, no VCS cluster formation will be attempted. You do not *want* a VCS cluster to be formed. If you want to give each switch a unique RBridge ID to simplify your switch management, you must also change its VCS ID to be unique within the IP Fabric to prevent a VCS Fabric from being formed.

NOTE
The only exception to the above within an IP Fabric is when you want to configure multihoming. For more information, refer to (Optional) Configuring support for dual-homed servers on page 50.

Controllerless Network Virtualization with BGP EVPN

Overview of controllerless network virtualization with BGP EVPN

Layer 2 extension mechanisms using VXLAN rely on “flood and learn” mechanisms. These mechanisms are very inefficient, making MAC address convergence longer and resulting in unnecessary flooding.

Also, in a data center environment with VXLAN-based Layer 2 extension mechanisms, a Layer 2 domain and an associated subnet might exist across multiple racks and even across all racks in a data center site. With traditional underlay routing mechanisms, routed traffic destined to a VM or a host belonging to the subnet follows an inefficient path in the network, because the network infrastructure is aware only of the existence of the distributed Layer 3 subnet, but is not aware of the exact location of the hosts behind a leaf switch.

With Brocade BGP EVPN network virtualization, network virtualization is achieved through creation of a VXLAN-based overlay network. Brocade BGP EVPN network virtualization leverages BGP-EVPN to provide a control plane for the virtual overlay network. BGP EVPN enables control-plane learning for end hosts behind remote VXLAN tunnel endpoints (VTEPs). This learning includes reachability for Layer 2 MAC addresses and Layer 3 host routes.

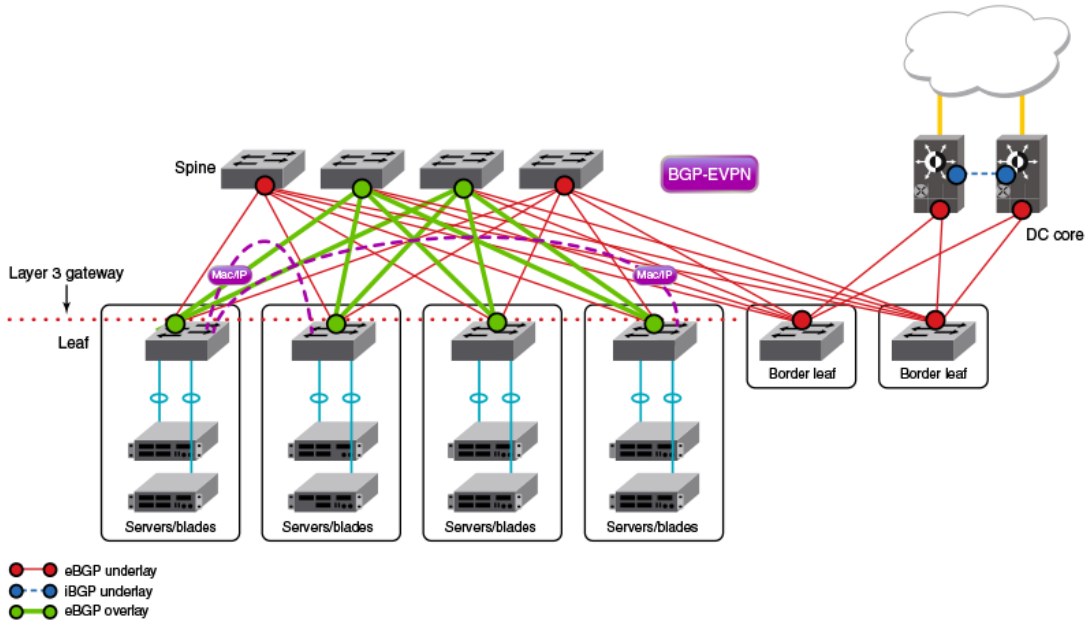
Some key features and benefits of Brocade BGP EVPN network virtualization are summarized as follows:

- Active-active vLAG pairs: vLAG pairs for multiswitch port channel for dual homing of network endpoints are supported at the leaf. Both the switches in the vLAG pair participate in the BGP-EVPN operations and are capable of actively forwarding traffic.
- Static anycast gateway: With static anycast gateway technology, each leaf is assigned the same default gateway IP and MAC addresses for all the connected subnets. This ensures that local traffic is terminated and routed at Layer 3 at the leaf. This also eliminates any suboptimal inefficiencies found with centralized gateways. All leaves are simultaneously active forwarders for all default traffic for which they are enabled. Also, because the static anycast gateway does not rely on any control plane protocol, it can scale to large deployments.

- Efficient VXLAN routing: With the existence of active-active vLAG pairs and the static anycast gateway, all traffic is routed and switched at the leaf. Routed traffic from the network endpoints is terminated in the leaf and is then encapsulated in VXLAN header to be sent to the remote site. Similarly, traffic from the remote leaf node is VXLAN-encapsulated and needs to be decapsulated and routed to the destination. This VXLAN routing operation into and out of the tunnel on the leaf switches is enabled in the Brocade VDX 6740 and 6940 platform ASICs. VXLAN routing performed in a single pass is more efficient than competitive ASICs.
- Data plane IP and MAC learning: With IP host routes and MAC addresses learned from the data plane and advertised with BGP EVPN, the leaf switches are aware of the reachability information for the hosts in the network. Any traffic destined to the hosts takes the most efficient route in the network.
- Layer 2 and Layer 3 multitenancy: BGP EVPN provides control plane for VRF routing as well as for Layer 2 VXLAN extension. BGP EVPN enables a multitenant infrastructure and extends it across the data center site to enable traffic isolation between the Layer 2 and Layer 3 domains, while providing efficient routing and switching between the tenant endpoints.
- Dynamic tunnel discovery: With BGP EVPN, the remote VTEPs are automatically discovered. The resulting VXLAN tunnels are also automatically created. This significantly reduces Operational Expense (OpEx) and eliminates errors in configuration.
- ARP/ND suppression: As the BGP EVPN EVI leaves discover remote IP and MAC addresses, they use this information to populate their local ARP tables. Using these entries, the leaf switches respond to any local ARP queries. This eliminates the need for flooding ARP requests in the network infrastructure.
- Conversational ARP/ND learning: Conversational ARP/ND reduces the number of cached ARP/ND entries by programming only active flows into the forwarding plane. This helps to optimize utilization of hardware resources. In many scenarios, there are software requirements for ARP and ND entries beyond the hardware capacity. Conversational ARP/ND limits storage-in-hardware to active ARP/ND entries; aged-out entries are deleted automatically.
- VM mobility support: If a VM moves behind a leaf switch, with data plane learning, the leaf switch discovers the VM and learns its addressing information. It advertises the reachability to its peers, and when the peers receive the updated information for the reachability of the VM, they update their forwarding tables accordingly. BGP EVPN-assisted VM mobility leads to faster convergence in the network.
- Simpler deployment: With multi-VRF routing protocols, one routing protocols session is required per VRF. With BGP EVPN, VRF routing and MAC address reachability information is propagated over the same BGP sessions as the underlay, with the addition of the L2VPN EVPN address family. This significantly reduces OpEx and eliminates errors in configuration.
- Open standards and interoperability: BGP EVPN is based on the open standard protocol and is interoperable with implementations from other vendors. This allows the BGP EVPN-based solution to fit seamlessly in a multivendor environment

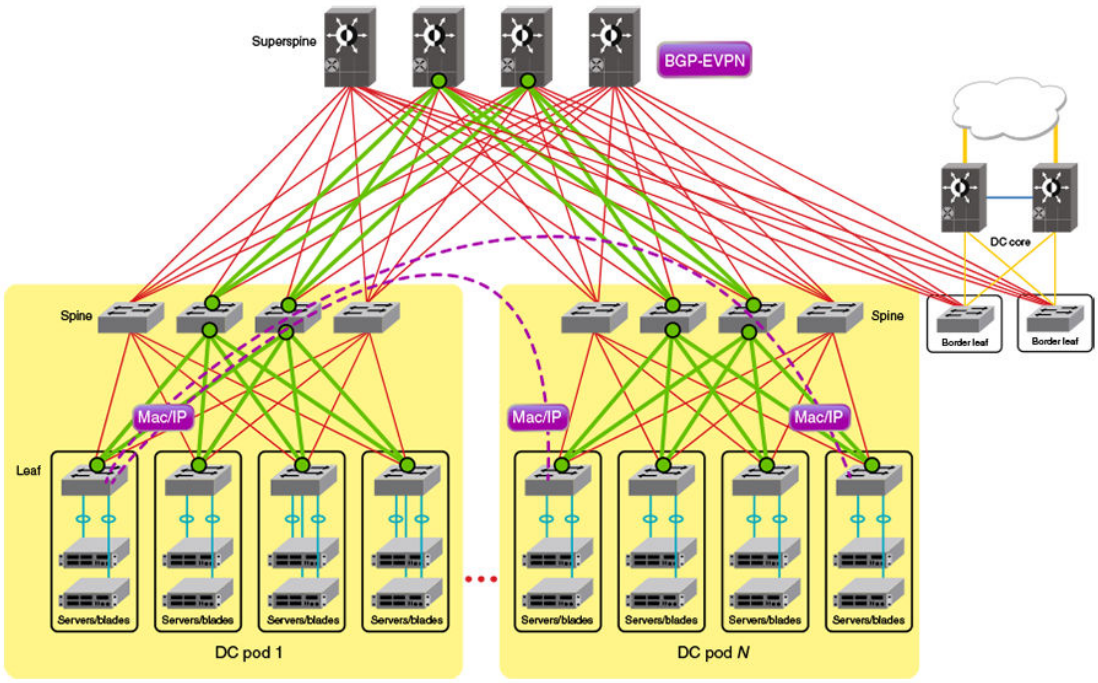
The following illustration depicts a 3-stage Clos topology that uses BGP EVPN. VXLAN tunnels are represented by dashed lines. This topology allows the provider to extend VLANs across racks with support from a Layer 3 underlay. Note that there is no controller, as VTEP autodiscovery is used. VTEPs are automatically discovered across the racks, and all MAC and IP address learning occurs automatically on the control plane. In this case, manual or controller intervention is not required. BGP EVPN can be extended to the border leaf nodes.

FIGURE 4 Overlay for 3-stage Clos network without controller



The following figure depicts a topology similar to that shown above, but with an optional optimized 5-stage folded Clos topology (superspine) for data center connectivity. Here the border leaf nodes connect directly to the superspine nodes. BGP EVPN can be extended to the border leaf nodes.

FIGURE 5 Overlay for optimized 5-stage folded Clos topology without controller



BGP-based underlay architecture supporting BGP EVPN

This section illustrates the basic BGP underlay architecture that is required to support the BGP EVPN overlay.

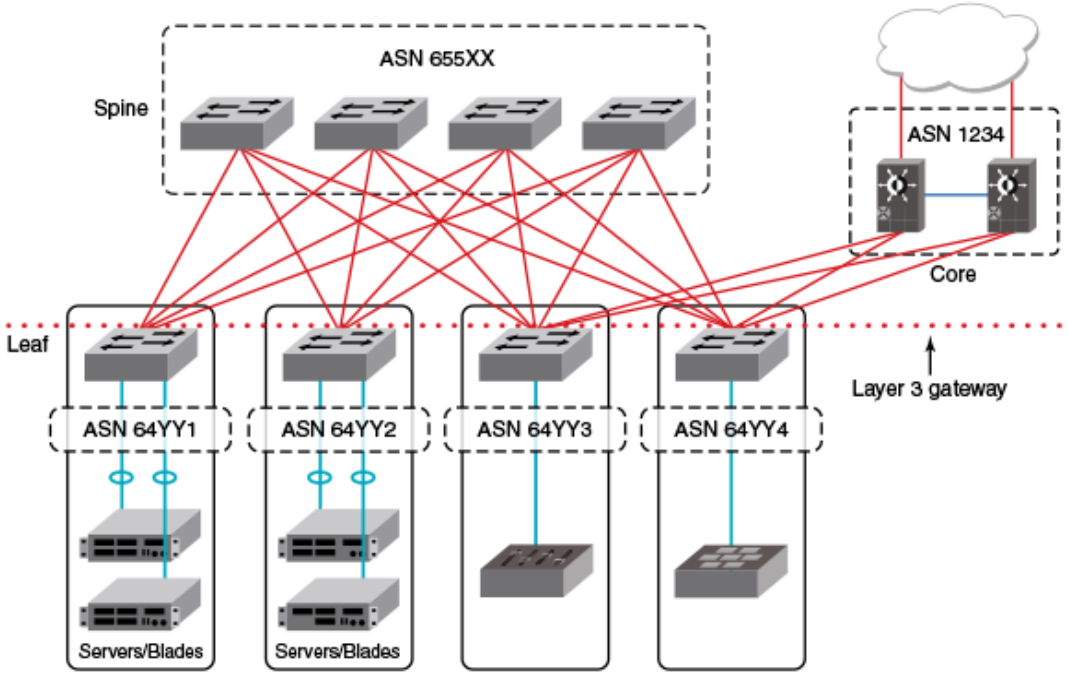
eBGP-based BGP EVPN

When eBGP is used for BGP EVPN in an IP Fabric, spine switches are configured to be in one autonomous system (AS). Each leaf switch or ToR pair is configured to be in a different AS from that of the spine switches, and advertises the routes by using local VTEP IP addresses as the next hop address.

For the BGP EVPN address family, spine switches are configured to advertise these routes to other eBGP peers, leaving the next-hop attribute unchanged. The spines are configured with separate VCS IDs.

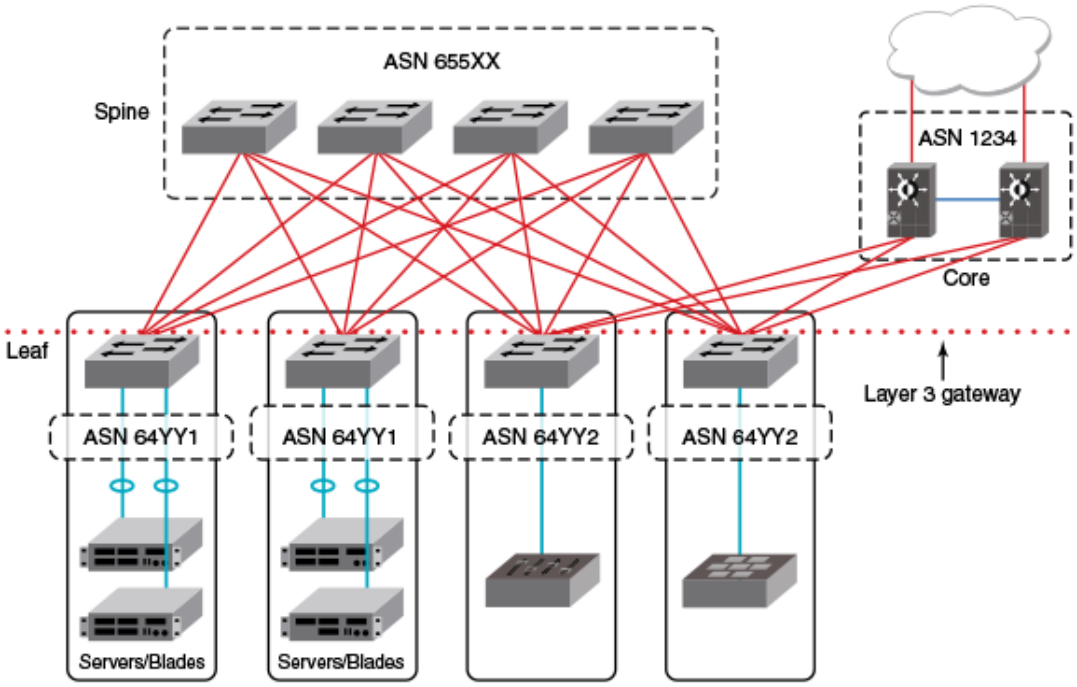
The following figure shows the eBGP underlay configuration in an IP Fabric where the spine switches are configured to be in one AS. Each leaf switch is configured to be in a separate AS.

FIGURE 6 eBGP underlay configuration with leaf switches in a separate AS



The following figure shows the eBGP underlay configuration in an IP Fabric where the spine switches are configured to be in one AS. Leaf 1 and leaf 2 are configured to be in one AS, and leaf 3 and leaf 4 are configured to be in another AS.

FIGURE 7 eBGP underlay configuration with leaf switches in two separate AS



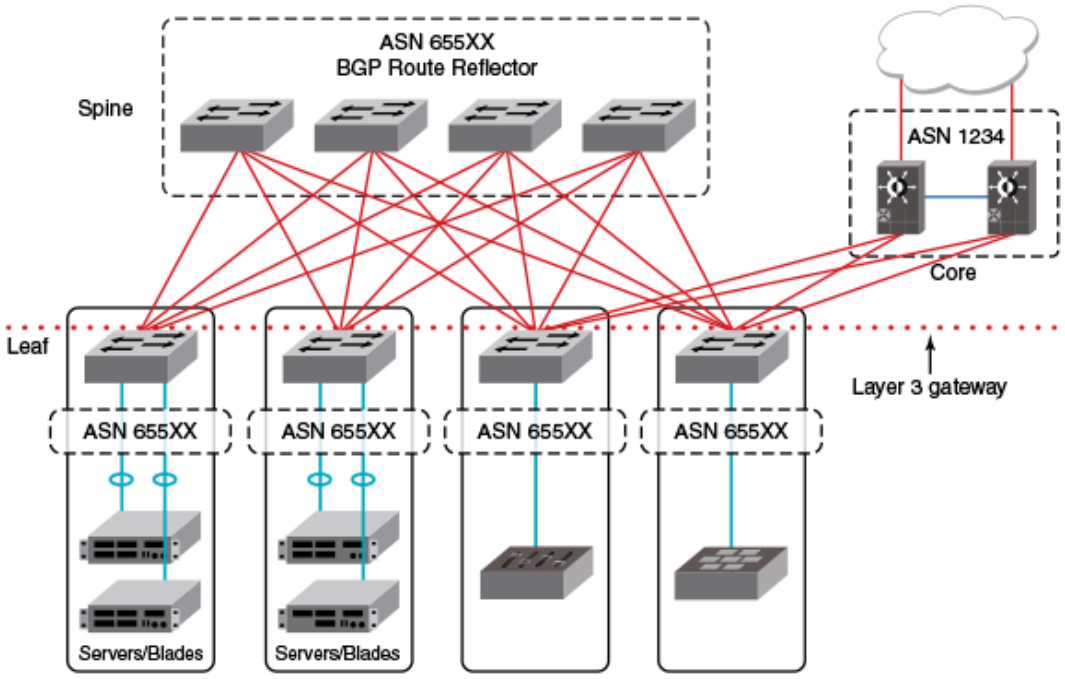
iBGP-based BGP EVPN

When iBGP is used for BGP EVPN in an IP Fabric, spine switches must be configured as route reflectors (RRs). This is because iBGP generally requires a switch to peer with every other device in the IP Fabric. When the spine switch is configured as an RR this situation is avoided.

However, BGP RRs only reflect the best route. Only the single, best prefix is reflected to each leaf, which is configured as an RR client, even if multiple Equal-Cost Multipath (ECMP) routes exist. To enable faster convergence with ECMP routes, and ensure that a BGP RR sends multiple paths to the clients, the BGP add path feature must be configured in IPv4 or IPv6 address-family unicast configuration mode to advertise additional ECMP paths to the clients. To meet all IP Fabrics requirements when using iBGP for an IP Fabric, spine switches must support BGP RR and BGP add path. The entire IP Fabric can then be managed as a single ASN.

The following figure shows the iBGP underlay configuration for an IP Fabric. The entire IP Fabric is managed as a single ASN.

FIGURE 8 iBGP underlay configuration an IP Fabric



BGP add path

The BGP add path feature provides the ability for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

When iBGP is used for BGP EVPN in an IP Fabric, spine switches are configured as route reflectors (RRs) so that a switch is not required to peer with every other device in the IP Fabric. Leaf switches are configured as RR clients.

BGP add path is supported for the BGP IPv4 and IPv6 unicast address families. It is not supported for the BGP L2VPN EVPN address family. BGP add path is not required for the L2VPN EVPN address family as a unique RD at each RBridge ensures that the same prefixes from multiple sources are not suppressed.

For more information on BGP add path, refer to the BGP and BGP4+ chapters in the *Network OS Layer 3 Routing Configuration Guide*.

BGP EVPN NLRI

BGP EVPN distributes Network Layer Reachability Information (NLRI) for a network. BGP EVPN NLRI includes both Layer 2 and Layer 3 reachability information for end hosts residing in the network, and advertises both the MAC and IP addresses of the EVPN VXLAN end hosts.

The following table shows BGP EVPN support for NLRI.

TABLE 1 BGP EVPN support for NLRI

Support for NLRI	Description
Auto-discovery (AD) per ES	An ESI can be associated with more than one EVPN instance. When an interface with an associated ESI and EVPN instance comes online, an auto-discovery (AD) route per ES route is originated and installed in the corresponding EVPN instance. The ESI can be distinguished locally through the IP address of the peer.
MAC and MAC IP	The MAC and MAC-IP addresses of the EVPN VXLAN end hosts are advertised to peers, and the distribution of these addresses through BGP EVPN reduces unknown unicast flooding in the VXLAN.
Ethernet Tag Routes	Multicast Ethernet Tag routes advertise VLAN membership to peers, indicating the type of multicast tunnel on which flooded traffic can be received for a VLAN.
ES-Routes	Multihoming support is provided through the advertisement of BGP Ethernet Segment Routes (ES-Routes). An Ethernet Segment (ES) is the set of links connected to the same multihomed host. An Ethernet Segment Identifier (ESI) is associated with each Ethernet segment and advertised in the Ethernet Segment Route (ES-Route). For BGP EVPN, BGP initiates an ES-Route in the EVPN routing table and advertises it to EVPN neighbors, thus distributing NLRI for the network.
Prefix Routes	IPv4 and IPv6 prefix routes belonging to tenants (VRFs) are exchanged providing inter-subnet connectivity within the Data center.

MAC address learning

Data plane (Layer 2) MAC address learning is enabled by default at the end points of the statically configured VXLAN tunnel..

Where BGP routing is used, MAC learning can be enabled by means of the **mac-learning protocol bgp** command. This delegates the responsibility for MAC learning to the BGP control-plane protocol.

For more information on MAC address learning, refer to Configuring MAC learning of Layer 2 extension site through BGP on page 58, as well as to the **mac-learning protocol bgp** command in the *Network OS Command Reference*.

EVPN instances

An EVPN instance (EVI) is an EVPN routing and forwarding instance spanning across the provider edges participating in that EVPN. EVIs are created using the **evpn-instance** command. Each EVI is identified by this configured name and is assigned an EVPN instance ID by the device.

Each EVI has a unique route distinguisher (RD) and one or more route targets (RT). RTs control the routes to be imported into and exported from the EVPN instance. An EVPN Routing table, containing information about the various routes associated with the EVI, is maintained for each EVI instance.

NOTE
For Network OS 7.0.0, only one EVPN instance (EVI) is supported.

EVIs can be configured with the following features:

- Route distinguisher (RD): Unique route distinguishers are assigned for an EVI. This value is automatically derived globally using the **rd auto** command so that each EVI has an associated RD

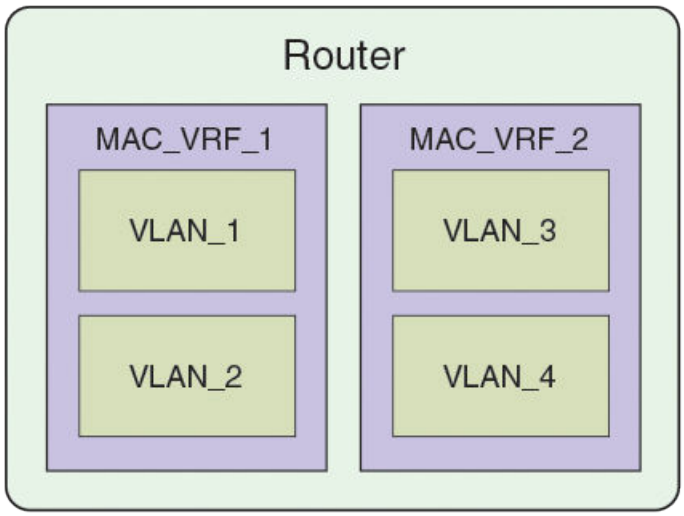
that is unique across the entire fabric. RDs can also be manually configured for a specific virtual network identifier (VNI) under an EVI using the **rd (VNI)** command.

- Route targets: The Route Target (RT) extended community attribute, enabling logical grouping of EVPN routes that can be imported or exported, can be specified for an EVI. These route targets can be globally derived automatically using the **route-target (EVPN)** command. The **ignore-as** option for the **route-target (EVPN)** command should be used in instances where the leaf nodes are under different autonomous systems and you want to import the routes from one leaf node to another. Route targets can also be manually configured for a specific virtual network identifier (VNI) under an EVI using the **route-target (VNI)** command. BGP EVPN uses these route targets to control the advertisement of EVPN routes.
- Virtual network identifiers (VNIs): VNIs can be added and removed for an EVI using the **vni** command.
- Duplicate MAC detection timer: When the same host MAC address is learned by two different devices, continuous MAC moves are triggered by the traffic originating from these hosts. A duplicate MAC detection timer can be configured, using the **duplicate-mac-timer** command, to detect these continuous MAC moves. This timer specifies both a time interval and the maximum threshold of MAC moves that can occur within the configured time interval before the MAC address is treated as a duplicate address and further processing of updates for that MAC address are blocked.

Refer to the *Network OS Command Reference* for more information on the commands discussed in this section.

The following figure shows an RBridge with two configured EVIs, MAC_VRF_1 and MAC_VRF_2. Each EVI has a separate MAC-to-VLAN table.

FIGURE 9 RBridge with two configured EVIs



BGP L2VPN EVPN address family

The BGP L2VPN EVPN address-family configuration level provides access to commands that allow you to configure BGP EVPN. The BGP L2VPN EVPN address family uses components of BGP that are independent of the IPv4 and IPv6 unicast address families. Both iBGP and eBGP are supported.

The L2VPN EVPN address family supports the EVPN Subsequent Address Family Identifier (SAFI), an address qualifier that provides additional information about the Network Layer Reachability Information (NLRI) type for a given attribute.

The commands that you enter at this level apply only to the BGP L2VPN EVPN address family. BGP L2VPN EVPN address family capability can be negotiated along with the IPv4 or IPv6 address family. A separate BGP session is not required for the BGP EVPN address family.

The following configuration options are allowed under BGP address-family L2VPN EVPN configuration mode:

```
device(config-bgp-evpn)# ?

Possible completions:

 client-to-client-reflection   Configure client to client route reflection
 graceful-restart              Enables the BGP graceful restart capability
 neighbor                      Specify a neighbor router
 retain                        Retain route targets
 vtep-discovery                Enable VTEP discovery
```

The following neighbor configuration options are allowed under BGP address-family L2VPN EVPN configuration mode:

```
device(config-bgp-evpn)# neighbor 10.1.1.1 ?

Possible completions:

 activate                Allow exchange of route in the current family mode
 allowas-in              Disables the AS_PATH check of the routes learned from the AS
 maximum-prefix
 next-hop-unchanged      Next hop unchanged
 route-map               Apply route map
 route-reflector-client  Configure a neighbor as Route Reflector client
 send-community          Send community attribute to this neighbor
```

Automatic VXLAN tunnel endpoint discovery

VXLAN tunnel endpoint (VTEP) IP addresses are carried in every BGP EVPN route so that the leaf device receiving the BGP EVPN updates triggers VXLAN tunnel creation using this remote VTEP IP address. Remote VTEP discovery is enabled by default for BGP EVPN when the BGP L2VPN EVPN address family is enabled.

BGP devices can determine which VLANs are common between two devices and extend those VLANs over the VXLAN tunnel between the two devices. VTEP IP address is carried in BGP EVPN updates in next-hop network address field of the MP_REACH_NLRI attributes. BGP deletes VXLAN tunnels associated with remote VTEP when all of the routes from remote VTEP are withdrawn. If you want to configure a tunnel explicitly (statically) or in an NSX Controller deployment, the BGP Automatic VTEP feature should be disabled by means of the **no vtep-discovery** command.

BGP next hop unchanged

The BGP next hop unchanged feature is supported for the L2VPN EVPN address family, allowing BGP to send updates to eBGP peers with the next hop in the MP_REACH_NLRI attribute unchanged.

By default, eBGP BGP speakers change the next hop while sending the updates to eBGP neighbors. The BGP next-hop-unchanged feature overrides this behavior so that the next hop address remains unchanged while updates are sent to eBGP peers. The BGP speaker is forced to retain the next hop address in the BGP updates received from neighbors. BGP next-hop-unchanged should be configured on the spine switch for each EVPN neighbor if the leaf switch is an eBGP peer. BGP next-hop-unchanged should be configured on the leaf switch for each EVPN neighbor if the spine switch is an eBGP peer. This means that eBGP BGP speakers will not change the next hop while sending updates to eBGP neighbors in an IP Fabric. Therefore, by configuring BGP next-hop-unchanged under the BGP L2VPN EVPN address family, changes to the next hop address in BGP EVPN updates are prevented.

NOTE
BGP next-hop-unchanged should be configured on leaf nodes and on spine nodes for all types of BGP deployments.

BGP retain route target

The BGP retain route target feature allows a spine node to accept all route targets (RTs).

Because Spine switches do not have EVPN instances (EVIs) and VRFs configured, BGP EVPN routes are filtered as otherwise none of the RTs in the routes match the local route-target import configuration. When spine switches are configured as RRs, or establish eBGP peering with the leaf nodes, the BGP retain route target feature should be enabled so that the Spine switches do not do route-target filtering. This means that all route targets are retained and the route-target attributes in the EVPN routes are not modified or removed.

BGP retain route target is supported for the BGP L2VPN EVPN address family only and is used in BGP EVPN configurations. It is not supported for the IPv4 and IPv6 unicast address families.

BGP legacy features supported for the L2VPN EVPN address family

The following BGP features, already supported for IPv4 and IPv6 unicast address families, are supported for the BGP L2VPN EVPN address family and used in BGP EVPN configurations:

- BGP extended community: The BGP extended community feature filters routes based on a regular expression specified when a route has multiple community values in it. The BGP extended community feature is supported for the L2VPN EVPN address family for both spine and leaf nodes. When a spine or leaf node receives BGP EVPN routes from other spine nodes, it checks the route-target extended community attribute of the routes. If the route-target is the same as the import target of the given EVPN instance on the local leaf node, the leaf node adds the route to the EVPN routing table. If the spine node is configured with the BGP retain route target feature, this checking is bypassed in the spine. If a static MAC address is redistributed to BGP, it is advertised with the

"Sticky MAC" extended community attribute. The next hop router MAC address for prefix routes is advertised as the default gateway extended community attribute.

- BGP graceful restart: BGP graceful restart (GR) can be configured for the L2VPN EVPN address family. When GR capability is negotiated, neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart. GR helps retain the peer routes when a restart occurs during HA failover. When the GR feature is enabled for the L2VPN EVPN address family, existing sessions are not affected and require neighbor reset to negotiate the GR capability.
- BGP peer groups: BGP peer groups can be configured for the L2VPN EVPN address family so that neighbors with the same attributes and parameters can be grouped together.
- BGP route reflection: The BGP route reflection feature is supported for the L2VPN EVPN address family. When iBGP is used for BGP EVPN in an IP Fabric, spine switches are configured as route reflectors (RRs) and leaf switches are configured as route reflector clients. You can configure a leaf switch as a route reflector client from the spine switch using the **neighbor route-reflector-client** command. Each leaf switch should be configured so that they all belong in the same cluster.
- Client-to-client-reflection: The BGP client-to-client reflection feature is supported for the L2VPN EVPN address family. For iBGP, if the leaf switches are configured as route reflector clients and are connected in a full iBGP mesh, you can disable client-to-client reflection on the spine switch which is configured as an RR. By default, in an IP Fabric, the clients of a route reflector are not required to be fully meshed and the routes from a client are reflected to other clients.
- Disabling the BGP AS_PATH check: A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected. When eBGP is configured for BGP EVPN in an IP Fabric, the AS_PATH check function can be disabled on a leaf switch so that route updates from the spine layer are not discarded by the leaf.

Ethernet Segment Identifiers for BGP routing

An Ethernet Segment is a set of Ethernet links that connect a multihomed device to a BGP router. Ethernet Segments are assigned a unique identifier, referred to as an Ethernet Segment Identifier (ESI). ESIs can be configured on port-channel interfaces.

NOTE
You must specify an ESI value under a port-channel interface if you want MAC addresses or ARP/ND learnt on the port-channel interface to be advertised to BGP EVPN neighbors.

Ethernet links that connect single-homed devices are not required to have an ESI value associated with them. For BGP EVPN, each BGP device advertises an Ethernet Segment Route (ES-Route) informing other BGP devices that it is connected to that ES. The other BGP devices can then detect if they are connected to the same ES. The user can use the **esi** command in port-channel configuration mode to configure the port-channel to derive the ESI value automatically by means of Link Aggregation Control Protocol (LACP). ESI value can be manually also configured on the port channel. This is needed for static port channels where LACP is not involved.

Multi-VRF for BGP EVPN

Multi-VRF can be configured for an IP Fabric. The link between the leaf and spine switches must be configured in the default VRF. For a nondefault VRF, traffic is routed at the leaf switch and is forwarded to a VXLAN tunnel.

Multi-VRF can be configured for an IP Fabric using asymmetric routing or symmetric routing. For asymmetric routing, all VLANs are configured on every leaf switch and enabled for IP routing and forwarding. Traffic flows through different routes in different directions. This severely affects scaling with each switch carrying the MAC and MAC-IP routes for hosts that are not necessarily locally connected.

For symmetric routing, a VLAN is only configured at the leaf switch where a local host exists. A unique, common Layer 3 virtual network identifier (VNI) number is generated for all leaf nodes for the VRF instance using the **vni (VRF)** command. The ingress leaf switch performs a Layer 3 lookup and routes the packets towards the remote leaf switch over the common Layer 3 VNI. The inner MAC destination address (DA) of the packet is rewritten to that of the gateway MAC address advertised by the remote leaf switch and the packet is then encapsulated in a VXLAN tunnel and sent to the remote leaf switch. The remote leaf switch then terminates the VXLAN tunnel. Because the MAC DA of the inner packet is now that of the gateway MAC address advertised by the leaf switch, it performs Layer 3 lookup and the packet is routed to the locally attached host.

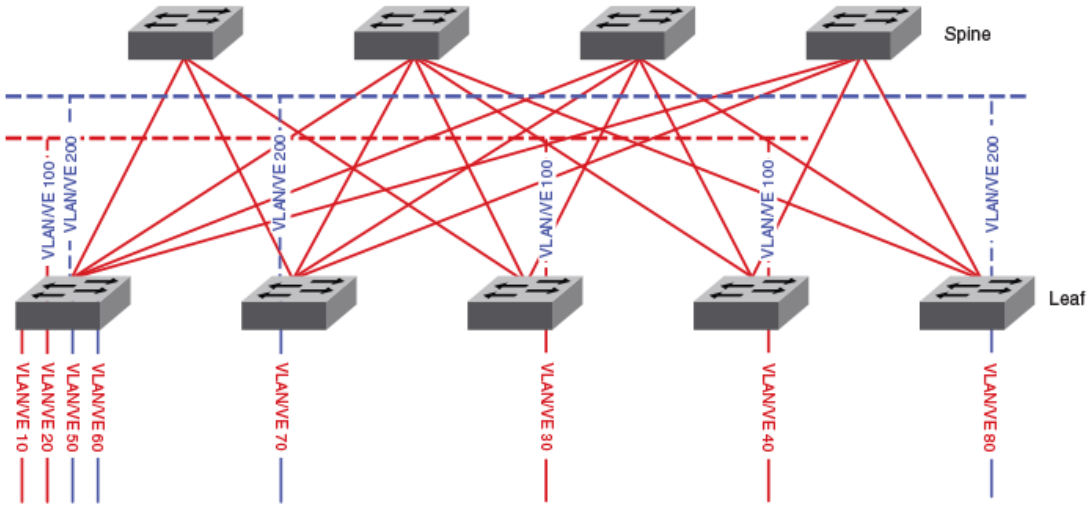
VRFs can be configured with the following features:

- Virtual network identifiers (VNIs): Layer 3 VNIs can be added and removed for a VRF instance using the **vni (VRF)** command. The VNI configuration under a VRF instance tells BGP to include the Layer 3 VNI and its associated MAC address in EVPN updates.
- Route distinguisher (RD): Unique route distinguishers are assigned for a VRF instance. RDs are manually configured for a VRF instance using the **rd (VRF)** command.
- Route targets: The route target (RT) extended community attribute, enabling logical grouping of EVPN routes that can be imported or exported, can be specified for a VRF instance. Route targets are manually configured for a specific VRF instance using the **route-target (VRF)** command. BGP EVPN uses these route targets to control the exchange of EVPN routes.

Refer to the *Network OS Command Reference* for more information on the commands discussed in this section.

The follwoing figure illustrates Multi-VRF topology using BGP EVPN. In this figure VRF red has VLAN 10, 20, 30, and 40. VLAN 100 is the Layer 3 VNI for VRF red. VRF blue has VLAN 50, 60, 70, and 80. VLAN 200 is the Layer 3 VNI for VRF blue.

FIGURE 10 Multi-VRF for BGP EVPN



Static Anycast Gateway

Overview of static anycast gateway

Static anycast gateway enables you to configure identical IP/MAC gateway addresses to virtual Ethernet (VE) interfaces on leaf switches in an IP Fabric, increasing routing efficiency.

Static anycast gateway provides seamless virtual machine (VM) mobility across all of the leaf (ToR) switches . Even if hosts move among leaf switches, there is no need to reconfigure the default gateway. In this way, forwarding behavior is optimized.

Considerations and limitations for static anycast gateway

There are considerations and limitations that you need to be aware of when implementing static anycast gateway.

Static anycast gateway is supported only in an IP Fabric, with the following technologies enabled:

- BGP EVPN
- ARP or ND suppression

Static anycast gateway is not supported along with Fabric-Virtual-Gateway (FVG).

The maximum number of virtual Ethernet (VE) interfaces supported under static anycast gateway is 4000, and the maximum number of virtual IP addresses supported per VE interface is 32.

The following tables summarize support for VRRP and VRRP-E with static anycast gateway on the supported platforms.

TABLE 2 Support for static anycast gateway with VRRP and VRRP-E (Brocade VDX 6740 series)

Static anycast gateway MAC address	VRRP (IPv4 or IPv6)	VRRP-E (IPv4 or IPv6)
Default	No	Yes
Nondefault	No	No

TABLE 3 Support for static anycast gateway with VRRP and VRRP-E (Brocade VDX 6940 series)

Static anycast gateway MAC address	VRRP (IPv4 or IPv6)	VRRP-E (IPv4 or IPv6)
Default	Yes	Yes
Nondefault	Yes	No

Configuring MAC static anycast gateway addresses

To implement static anycast gateway, you need to specify gateway MAC addresses for IPv4 and IPv6 traffic on each RBridge.

NOTE
Depending on your needs, you can specify a MAC address for IPv4 traffic, for IPv6 traffic, or for both.

1. Enter **configure terminal** to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **rbridge-id** command to access RBridge ID configuration mode.
   ```
   device(config)# rbridge-id 1
   ```
3. To specify a static anycast gateway MAC address for IPv4 traffic, enter the **ip anycast-gateway-mac** command, with one of the following options:
   - To specify the default static anycast gateway MAC address for IPv4 traffic, enter the following command:
     ```
     device(config-rbridge-id-1)# ip anycast-gateway-mac default-mac
     ```

     NOTE
     "Default" means that an automatically generated MAC address is used.
   - To specify a nondefault static anycast-gateway MAC address for IPv4 traffic, enter a command such as the following:
     ```
     device(config-rbridge-id-1)# ip anycast-gateway-mac 2222.2244.4444
     ```
4. To specify a static anycast gateway MAC address for IPv6 traffic, enter the **ipv6 anycast-gateway-mac** command, with one of the following options:
   - To specify the default static anycast gateway MAC address for IPv6 traffic, enter the following command:
     ```
     device(config-rbridge-id-1)# ipv6 anycast-gateway-mac default-mac
     ```
   - To specify a nondefault static anycast gateway MAC address for IPv6 traffic, enter a command such as the following:
     ```
     device(config-rbridge-id-1)# ipv6 anycast-gateway-mac 2222.2266.6666
     ```

NOTE
The first three bytes (six digits) of a nondefault IPv4 static anycast gateway MAC address must be identical with the first three bytes of a corresponding IPv6 static anycast gateway MAC address.

Configuring IP static anycast gateway addresses

To implement static anycast gateway, you must specify IPv4 and IPv6 static anycast gateway addresses on a virtual Ethernet (VE) interface.

NOTE
Depending on your needs, you can specify an IPv4 address, an IPv6 address, or both.

1. Enter **configure terminal** to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **rbridge-id** command to access RBridge ID configuration mode.
   ```
   device(config)# rbridge-id 1
   ```
3. Enter the **interface ve** command to access VE (RBridge) interface configuration mode.
   ```
   device(config-rbridge-id-1)# interface ve 10
   ```
4. To specify an IPv4 static anycast gateway address, enter the **ip anycast-address** command.
   ```
   device(config-rbridge-Ve-1)# ip anycast-address 2.2.2.2/24
   ```
5. To specify an IPv6 static anycast gateway address, enter the **ipv6 anycast-address** command.
   ```
   device(config-rbridge-Ve-1)# ipv6 anycast-address 1234:10::10:100/64
   ```

Show commands for static anycast gateway

Use the following **show** commands to verify configurations of static anycast gateway.

TABLE 4 Show commands for static anycast gateway

Command	Description
show ip anycast-gateway	Displays IPv4 static anycast gateway details for all virtual Ethernet (VE) interfaces or for a specified VE interface. You can also filter by RBridge and by VRF.
show ipv6 anycast-gateway	Displays IPv6 static anycast gateway details for all VE interfaces or for a specified VE interface. You can also filter by RBridge and by VRF.

ARP and ND Scaling Enhancements

ARP and Neighbor Discovery (ND)

When forwarding traffic, a device needs to know the destination's MAC address, because each IP packet is encapsulated in a MAC packet. The MAC address is needed not only for the packet's final destination but also for a next hop towards the destination.

The technology by which a device gets the MAC address varies between IPv4 and IPv6, as follows:

- IPv4: Address Resolution Protocol (ARP)
- IPv6: Neighbor Discovery (ND)

IPv4 traffic

When the destination's IP address is known, to get the MAC address, a device first searches its ARP cache. A match for the IP address supplies the corresponding MAC address. Otherwise, the device broadcasts an ARP request. The network devices receive such ARP requests, and the host with a matching IP address sends an ARP reply that includes its MAC address.

IPv6 traffic

When the destination's IPv6 address is known, to get the MAC address, a device first searches its neighbor cache. A match for the IPv6 address supplies the corresponding MAC address. Otherwise, the device broadcasts a neighbor solicitation (NS) request. The network devices receive the NS request, and the host with a matching IPv6 address sends a reply that includes its MAC address.

ARP and ND scaling enhancements

In many network configurations, ARP and Neighbor Discovery (ND) traffic and caching consume significant resources, which can be optimized as discussed in this chapter.

ARP and ND suppression

In a data center fabric, the ARP and ND suppression options can help reduce ARP and ND control traffic.

The default scenario leads to excess control traffic.

1. A device needs to forward traffic to an IP address, but the corresponding MAC address is not in its ARP or ND cache.
2. The device broadcasts an ARP or ND request throughout the IP Fabric.

When you enable ARP and ND suppression, excess control traffic is reduced.

1. A device needs to forward traffic to an IP address, but the corresponding MAC address is not in its ARP or ND cache.
2. The device broadcasts an ARP or ND request.
3. The leaf BGP EVPN control plane intercepts the request and looks for a match in its local cache.
   - If there is a match, the control plane responds to the device (rather than broadcasting the original request to the entire IP Fabric).
   - Only if there is no match, does the leaf control plane broadcast the request to the entire IP Fabric.

NOTE
Brocade recommends that you enable ARP and ND suppression on all VLANs in an IP Fabric.

Supported platforms

The ARP and ND suppression features are supported on the following platforms:

- Brocade VDX 6740 series
- Brocade VDX 6940 series

- Brocade VDX 2741 (not supported for IP Fabrics)
- Brocade VDX 2746 (not supported for IP Fabrics)

Enabling and disabling ARP and ND suppression on a VLAN

ARP and ND suppression can reduce ARP/ND traffic.

1. Enter **configure terminal** to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **interface vlan** command to access VLAN configuration mode.
   ```
   device(config)# interface vlan 110
   ```
3. To enable ARP suppression, enter **suppress-arp**.
   ```
   device(config-Vlan-110)# suppress-arp
   ```
   To disable ARP suppression, enter **no suppress-arp**.
4. To enable ND suppression, enter **suppress-nd**.
   ```
   device(config-Vlan-110)# suppress-nd
   ```
   To disable ND suppression, enter **no suppress-nd**.

The following example enables ARP suppression on VLAN 110.

```
device# configure terminal
device(config)# interface vlan 110
device(config-Vlan-110)# suppress-arp
```

Enabling and disabling ARP learning

Use this procedure to enable ARP learning not only locally, but from all ARP requests. ARP learning decreases the time needed to populate the ARP cache.

1. Enter **configure terminal** to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **rbridge-id** command to access RBridge ID configuration mode.
   ```
   device(config)# rbridge-id 1
   ```
3. Enter the **interface ve** command to access VE configuration mode.
   ```
   device(config-rbridge-id-1)# interface ve 110
   ```
4. To enable fabric learning, enter the **ip arp learn-any** command.
   ```
   device(config-ve-110)# ip arp learn-any
   ```
   To disable fabric learning, enter the **no ip arp learn-any** command.

ARP and ND suppression show and clear commands

There is a full range of ARP and ND suppression **show** and **clear** commands. They are documented in the *Network OS Command Reference*, and listed in the following tables with descriptions.

TABLE 5 ARP and ND suppression show commands

Command	Description
show ip arp suppression-cache	Displays IPv4 ARP suppression information.
show ip arp suppression-statistics	Displays IPv4 ARP suppression statistics.
show ip arp suppression-status	Displays the IPv4 ARP suppression status.

TABLE 5 ARP and ND suppression show commands (Continued)

Command	Description
show ipv6 nd suppression-cache	Displays IPv6 ND suppression information.
show ipv6 nd suppression-statistics	Displays IPv6 ND suppression statistics.
show ipv6 nd suppression-status	Displays the IPv6 ND suppression status.

TABLE 6 ARP and ND suppression clear commands

Command	Description
clear ip arp suppression-cache	Clears the IPv4 ARP suppression cache. You can also clear the cache for a specified VLAN.
clear ip arp suppression-statistics	Clears the IPv4 ARP suppression statistical information. You can also clear statistics for a specified VLAN.
clear ipv6 nd suppression-cache	Clears the IPv6 ND suppression cache. You can also clear the cache for a specified VLAN.
clear ipv6 nd suppression-statistics	Clears the IPv6 ND suppression statistical information. You can also clear statistics for a specified VLAN.

Conversational ARP and ND

Conversational ARP and ND reduce the number of cached ARP and ND entries by programming only active flows into the forwarding plane. This feature helps to optimize utilization of hardware resources.

In many use-case scenarios—especially in an IP Fabric—there are software requirements for ARP and ND entries beyond the hardware capacity. Conversational ARP and ND limit storage-in-hardware to active ARP and ND entries; aged-out entries are deleted automatically.

By default, the aging-out threshold is 300 seconds. (You can change the threshold to any integer value from 60 through 100,000 seconds, either before or during enablement.) Any entry that does not have at least one conversation before aging-out is deleted from the cache. Each conversation restarts the clock for that entry.

However, aging-out is also influenced by the enablement and disablement cycle:

1. When you enable conversational ARP and ND, a fast-aging policy of 30 seconds (not configurable) applies to all entries in the ARP and ND caches at that time.
2. From enablement of conversational ARP and ND, the aging-time value applies for existing entries with new traffic and for new entries.
3. Upon disablement, the conversational ARP and ND timers no longer apply. All current entries become permanent as do all new entries.

The following entries are not subject to conversational behavior:

- Static ARPs and NDs
- Next hops

Supported platforms

Conversational ARP and ND are supported on the following platforms:

- Brocade VDX 6740 series
- Brocade VDX 6940 series
- Brocade VDX 2741 (not supported for IP Fabrics)
- Brocade VDX 2746 (not supported for IP Fabrics)

Enabling and disabling conversational ARP and ND

Enabling conversational ARP and ND can reduce the number of cached ARP and ND entries, optimizing utilization of hardware resources.

1. Enter **configure terminal** to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **rbridge-id** command to access RBridge ID configuration mode.
   ```
   device(config)# rbridge-id 1
   ```
3. To specify an aging-time value other than the default 300 seconds, enter the **host-table aging-time conversational** command.
   ```
   device(config-rbridge-id-1)# host-table aging-time conversational 600
   ```
   To restore the default aging-time value of 300 seconds, enter the **no host-table aging-time conversational** command.
4. To enable conversational ARP and ND, enter the **host-table aging-mode conversational** command.
   ```
   device(config-rbridge-id-1)# host-table aging-mode conversational
   ```
   To disable conversational ARP and ND, enter the **no host-table aging-mode conversational** command.

The following example implements conversational ARP and ND. The current aging-time value applies.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# host-table aging-mode conversational
```

Conversational ARP and ND show and clear commands

Conversational ARP and ND **show** and **clear** commands are documented in the *Network OS Command Reference*. They are listed in the following tables with descriptions.

TABLE 7 Conversational ARP and ND show commands

Command	Description
show arp slot	Displays the ARP cache.
show ipv6 neighbor slot	Displays IPv6 neighbor information.

TABLE 8 Conversational ARP and ND clear commands

Command	Description
clear arp	Clears the local ARP cache and then resends ARP requests to the local hosts.

TABLE 8 Conversational ARP and ND clear commands (Continued)

Command	Description
clear arp no-refresh	Clears the ARP cache, without resending ARP requests to the local hosts.
clear ipv6 neighbor	Clears the IPv6 ND cache and then resends ND requests to the local hosts.
clear ipv6 neighbor no-refresh	Clears the ND cache, without resending ND requests to the local hosts.

Standards conformance and RFC support for BGP EVPN

This section lists the IETF RFCs and other draft documents that support the implementation of BGP EVPN in support of Brocade IP Fabrics.

The following table lists and describes the IETF BGP EVPN RFCs and other supporting draft documents.

TABLE 9 IETF BGP EVPN RFCs and other draft documents supporting Brocade IP Fabrics

Document	URL	Description
RFC 7432: "BGP MPLS-Based Ethernet VPN"	http://tools.ietf.org/html/rfc7432	Describes procedures for BGP MPLS-based Ethernet VPNs (EVPN).
"A Network Virtualization Overlay Solution using EVPN"	https://tools.ietf.org/html/draft-ietf-bess-evpn-overlay-01	Describes how Ethernet VPN (EVPN) can be used as an Network Virtualization Overlay (NVO) solution and explores the various tunnel encapsulation options over IP and their impact on the EVPN control-plane and procedures.
"Integrated Routing and Bridging in EVPN"	https://tools.ietf.org/html/draft-ietf-bess-evpn-inter-subnet-forwarding-00	Describes an extensible and flexible multi-homing VPN solution for intra-subnet connectivity among hosts/VMs over an MPLS/IP network.
"IP Prefix Advertisement in EVPN"	https://tools.ietf.org/html/draft-ietf-bess-evpn-prefix-advertisement-02	Defines a new EVPN route type for the advertisement of IP Prefixes and explains some use-case examples where this new route- type is used.

Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay

Configuration overview

This chapter provides the steps to set up a basic Brocade IP Fabric topology at the spine and leaf tiers, as well as to configure optional multihomed servers.

NOTE
This chapter is not meant to represent a production deployment scenario, with a full variety of deployment options. It is meant to facilitate the understanding of the basic steps that are required.

NOTE
Brocade offers Brocade Workflow Composer, an open and programmable turnkey automation solution that enables organizations to:

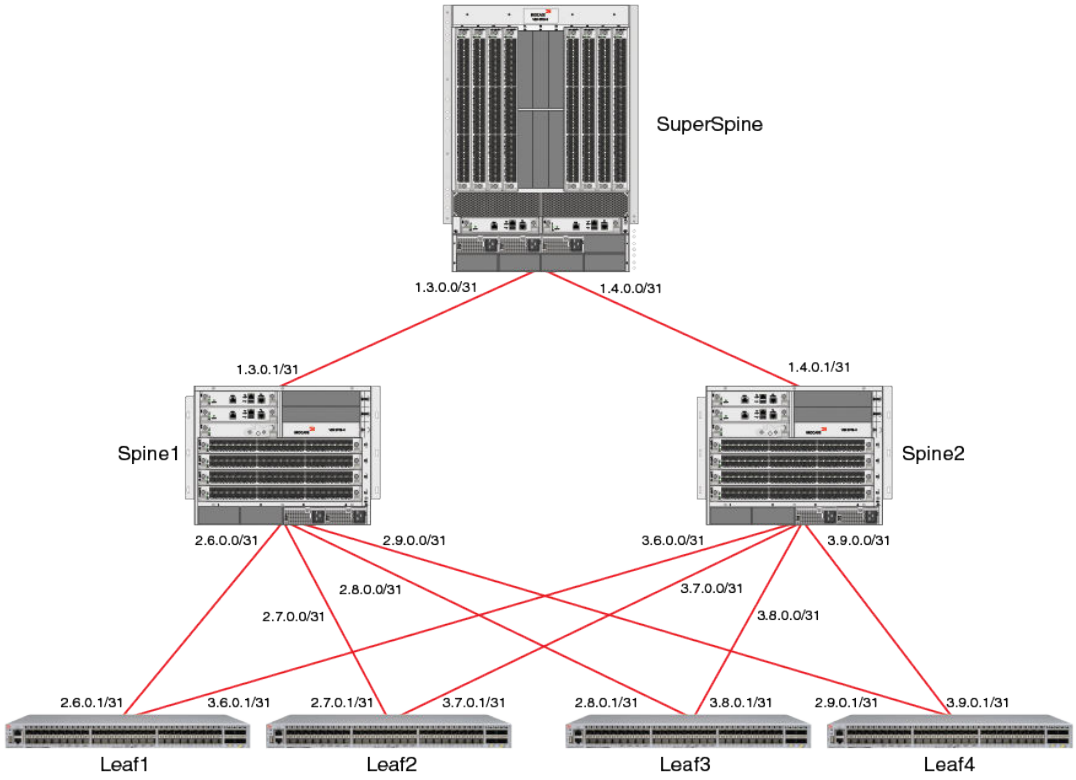
- Easily provision and validate IP Fabric infrastructure
- Customize automation to optimize IT operations

Brocade Workflow Composer's pre-packaged, Python-based automation modules can provision and validate Brocade IP Fabrics and BGP-EVPN with minimal effort, while support for open interfaces and commonly used infrastructure programming tools enables simple and straightforward customization when needed.

For more information, refer to the *Brocade Workflow Composer User Guide* on mybrocade.com. To find this guide, go to the Downloads area of mybrocade.com and type "Brocade Workflow Composer" In the search window.

The following illustrates the topology and subnets of the nodes used in this example configuration.

FIGURE 11 IP Fabrics topology



NOTE
Although this configuration example uses BGP EVPN to implement the overlay network, BGP EVPN is not essential to the configuration of an IP Fabric.

Refer to the topology in Multihoming on page 15 for the basic design. The example topology consists of a pair of switches at the spine tier, four switches at the leaf tier, and an optional superspine switch. The following table summarizes the basic configurations at each tier for an example switch.

TABLE 10 Basic configurations at leaf, spine, and optional superspine tiers

Leaf	Spine	Superspine
VLANs		
Static anycast gateway		
Ethernet interfaces	Ethernet interfaces	Ethernet interfaces
Loopback interfaces	Loopback interfaces	Loopback interfaces
(Optional) Port-channel interface with vLAG and Ethernet Segment Identifier to support dual-homed servers		
Access/trunk ports		
Routed, IP port-channel, or optional unnumbered IP interfaces	Routed, IP port-channel, or optional unnumbered IP interfaces	Routed, IP port-channel, or optional unnumbered IP interfaces

TABLE 10 Basic configurations at leaf, spine, and optional superspine tiers (Continued)

Leaf	Spine	Superspine
VRF instances (route distinguisher, VXLAN Network Identifier, route)		
Address family (IPv4, IPv6, EVPN): route target	Address family (IPv4, IPv6, EVPN)	Address family (IPv4, IPv6, EVPN)
Virtual Ethernet interfaces (VRF forwarding, anycast address)		
(Optional) EVPN instance (add VLANs)		
Overlay gateway (Layer 2 extension, loopback address, add RBridge, map VLANs to VNIs)		
(Optional) Configure interoperability with other vendors where Brocade extended VLANs must be remapped manually to their respective VNIs		
		(Optional) OSPF routing
Configure BGP and IGP (for iBGP-based networks) on all switches, to distribute Layer 3 reachability information; if eBGP is used, configure the **neighbor allowas-in** command, to reduce load on hardware processing; optionally enable Bidirectional Forwarding Detection (BFD) for BGP neighbors, for fast recovery in case of link failure	Configure BGP and IGP (for iBGP-based networks) on all switches, to distribute Layer 3 reachability information; optionally enable Bidirectional Forwarding Detection (BFD) for BGP neighbors, for fast recovery in case of link failure	Configure BGP and IGP (for iBGP-based networks) on all switches, to distribute Layer 3 reachability information; optionally enable Bidirectional Forwarding Detection (BFD) for BGP neighbors, for fast recovery in case of link failure
	BGP neighbor (remote AS, prevent route rejection, load balancing, recursive next-hop lookups, redistribute directly connected routes)	BGP neighbor (local AS, load balancing, recursive next-hop lookups, redistribute directly connected routes)

Configuring the leaf switches

This section presents the tasks to configure the leaf switches.

Creating the VLANs

This task establishes the required VLANs, to support both VRFs and VXLAN Network Identifiers (VNIs).

1. In global configuration mode, create the VLANs required to support service, starting with leaf switch Leaf1.

```
Leaf1# configure terminal
Leaf1(config)# interface vlan 101-108,120,121-128,140,141-148,160,161-168,180
```

NOTE
VLANs 120, 140, 160, and 180 will support VRFs as VXLAN Network Identifiers (VNIs). VLANs must be created before they can be added to a switchport trunk.

2. Repeat Step 1 as appropriate for the remaining leaf switches.
3. Use the **show running-config interface vlan** command to verify the configuration.

   Refer to Verifying the configuration on page 55 for examples of useful **show** commands.

Configuring static anycast gateway MAC addresses

This task configures static anycast gateway MAC addresses for IPv4 and IPv6 support.

1. **NOTE**
   Within a dual-stack IP Fabric, the first three bytes (six digits) of a nondefault IPv4 static-anycast-gateway MAC address must be identical with those of the corresponding IPv6 MAC address. IPv4 and IPv6 addresses can also be used, as well as default MAC addresses. For more information about the static anycast gateway feature, refer to Static Anycast Gateway on page 29.

   ATTENTION
   It is recommend that you do not use this feature unless BGP EVPN is implemented. It is also recommended that ARP/ND suppression be enabled.

   In RBridge ID configuration mode, specify IPv4 and IPv6 static anycast gateway MAC addresses, by using the **ip anycast-gateway-mac** and **ipv6 anycast-gateway-mac** commands, respectively.

```
Leaf1(config)# rbridge-id 1
Leaf1(config-rbridge-id-1)# ip anycast-gateway-mac 0000.abba.baba
Leaf1(config-rbridge-id-1)# ipv6 anycast-gateway-mac 0000.abba.abba
```

2. Repeat Step 1 as appropriate for the remaining leaf switches. The same MAC address must be used in all cases.
3. To verify the configuration, use the **show running-config rbridge-id ip anycast-gateway-mac** and the **show running-config rbridge-id ipv6 anycast-gateway-mac** commands.

   Refer to Verifying the configuration on page 55 for examples of useful **show** commands.

Configuring the interfaces

This task configures the physical interfaces, as well as the virtual (loopback) interfaces configured with Donor IP in the optional IP Unnumbered address configuration. It also references an optional task where servers are dual-homed to a leaf node for high availability.

1. In interface subtype configuration mode, configure an Ethernet interface for point-to-point connectivity between leaf switch Leaf1 and the spine switches.

   NOTE
   This example uses both IPv4 and IPv6 addresses, with /31 and /127 networks, respectively.

```
Leaf1(config-rbridge-id-1)# interface tengigabitethernet 1/0/1
Leaf1(config-if-te-1/0/1)# ip address 2.6.0.1/31
Leaf1(config-if-te-1/0/1)# ipv6 address 1000:2:6::1/127
Leaf1(config-if-te-1/0/1)# no shut
```

   Use the **description** keyword to facilitate management and maintenance of the network.

Repeat the preceding configuration for the second Ethernet interface.

```
interface tengigabitethernet 1/0/5
 ip address 3.6.0.1/31
 ipv6 address 1000:3:6::1/12
 no shut
```

2. In RBridge ID configuration mode, configure a loopback interface.

NOTE
The loopback interface provides the source of the donor addresses used for the IP unnumbered feature. This example uses both IPv4 and IPv6 addresses, with /32 and /128 networks, respectively.

```
Leaf1(config)# rbridge-id 1
Leaf1(config-rbridge-id-1)# interface loopback 1
Leaf1(config-Loopback-1)# ip address 6.0.0.6/32
Leaf1(config-Loopback-1)# ipv6 address 1000:6::6/128
Leaf1(config-Loopback-1)# no shut
```

3. (Optional) You can configure unnumbered IPv4 and IPv6 addresses, with an appropriate loopback address, as follows.

```
Leaf1(config-rbridge-id-1)# interface tengigabitethernet 1/0/1
Leaf1(config-if-te-1/0/1)# ip unnumbered loopback 1
Leaf1(config-if-te-1/0/1)# ipv6 unnumbered loopback 1
Leaf1(config-if-te-1/0/1)# no shut
```

4. In interface subtype configuration mode, configure an Ethernet interface for switchport trunk connectivity between leaf switch Leaf1 and the spine switches.

NOTE
These trunks will carry the ranged VLANs.

```
Leaf1(config-rbridge-id-1)# interface tengigabitethernet 1/0/16
Leaf1(config-if-te-1/0/16)# switchport
Leaf1(config-if-te-1/0/16)# switchport mode trunk
Leaf1(config-if-te-1/0/16)# switchport trunk allowed vlan add 101-108
Leaf1(config-if-te-1/0/16)# switchport trunk allowed vlan add 121-128
Leaf1(config-if-te-1/0/16)# no shut
```

Repeat the preceding configuration for the second Ethernet switchport trunk interface.

```
Leaf1(config-rbridge-id-1)# interface tengigabitethernet 1/0/17
Leaf1(config-if-te-1/0/16)# switchport
Leaf1(config-if-te-1/0/16)# switchport mode trunk
Leaf1(config-if-te-1/0/16)# switchport trunk allowed vlan add 141-148
Leaf1(config-if-te-1/0/16)# switchport trunk allowed vlan add 161-168
Leaf1(config-if-te-1/0/16)# no shut
```

5. Repeat Step 1 through Step 4 for the remaining leaf nodes, as appropriate.
6. (Optional) Where servers are dual-homed to a leaf node, refer to (Optional) Configuring support for dual-homed servers on page 50.
7. To verify the configuration, use the **show running-config interface** command.

   Refer to Verifying the configuration on page 55 for examples of useful **show** commands.

Configuring the VRF instances

This task establishes the VRF instances and applies them to virtual Ethernet (VE) interfaces.

NOTE
This task illustrates the implementation of BGP EVPN Layer 3 multitenancy where BGP EVPN is used for the overlay. BGP EVPN must be enabled to support VXLAN Network Identifiers (VNIs).

1. Create VRF instances, and configure a route distinguisher and specify the Layer 3 VNI to be used for intersubnet forwarding.
   a) Create a VRF instance, in this example "red".
   ```
   Leaf1(config-rbridge-id-1)# vrf red
   Leaf1(config-vrf-red)#
   ```
   b) Configure a route distinguisher (RD) and VNI.
   ```
   Leaf1(config-vrf-red)# rd 6.0.0.6:1
   Leaf1(config-vrf-red)# vni 120
   ```
2. Enter address-family IPv4 unicast configuration mode, configure import and export route targets, and specify EVPN routes as targets.
   ```
   Leaf1(config-vrf-red)# address-family ipv4 unicast
   Leaf1(config-vrf-red-ipv4-unicast)# route-target import 3:3 evpn
   Leaf1(config-vrf-red-ipv4-unicast)# route-target export 3:3 evpn
   Leaf1(config-vrf-red-ipv4-unicast)# exit
   Leaf1(config-vrf-red)#
   ```
3. Repeat the preceding configuration for IPv6.
   ```
   Leaf1(config-vrf-red)# address-family ipv6 unicast
   Leaf1(config-vrf-red-ipv6-unicast)# route-target import 3:3 evpn
   Leaf1(config-vrf-red-ipv6-unicast)# route-target export 3:3 evpn
   ```
4. Repeat Step 1 through Step 3 for all remaining VRF instances, and return to interface subtype configuration mode.
5. In interface subtype configuration mode, configure the VE interfaces corresponding to the ranged VLANs (101-108, 121-128, 141-148, 161-168) used to support the VRF instances with anycast addresses.
   a) Configure VE 101 to forward VRF "red".
   ```
   Leaf1(config-rbridge-id-1)# int ve 101
   Leaf1(config-Ve-101)# vrf forwarding red
   ```
   b) Configure VE 101 to support IPv4 and IPv6 static anycast gateway addresses, and enable the interface.
   ```
   Leaf1(config-Ve-101)# ip anycast-address 101.1.1.254/24
   Leaf1(config-Ve-101)# ipv6 anycast-address 101:1:1::254/64
   Leaf1(config-Ve-101)# no shut
   ```

   NOTE
   The anycast gateway address is used to configure the static anycast gateway MAC addresses for the hosts. This example also shows the optional configuration to support IPv6.
6. Repeat Step 5 for VE interfaces 102-108, 121-128, 141-148, 161-168 and their respective VRF instances.
7. In interface subtype configuration mode, configure the VE interfaces 120, 140, 160, 180 to support their respective VRF instances without anycast addresses.
   a) **NOTE**
   These are Layer 3 VNIs and therefore do not need IP addresses.

Configure VE 120 to forward VRF "red".

```
Leaf1(config-rbridge-id-1)# interface ve 120
Leaf1(config-Ve-120)# vrf forwarding red
```

b) Enable the interface.

```
Leaf1(config-Ve-120)# no shut
```

8. To support IPv6 traffic over the L3 VNI configured above, then configure support for an IPv6 address on the interface.
   a) In RBridge ID configuration mode, enter interface subtype configuration mode and specify the VE interface used above.

```
Leaf1(config-rbridge-id-1)# interface ve 120
```

   b) Enter the **ipv6 address use-link-local-only** command.

```
Leaf1(config-rbridge-Ve-120)# ipv6 address use-link-local-only
```

9. Repeat Step 7 for VE interfaces 140, 160, and 180 to support their respective VRF instances.
10. Repeat Step 1 through Step 9 for the remaining leaf nodes, as appropriate.
11. To verify the configuration, use the **show running-config rbridge-id interface ve** command.
    Refer to Verifying the configuration on page 55 for examples of useful **show** commands.

Configuring BGP routing

This task enables BGP routing and configures a variety of parameters.

1. Enable BGP routing on an RBridge and enter BGP global configuration mode.

```
Leaf1(config-rbridge-id-1)# router bgp
Leaf1(config-bgp-router)#
```

2. Use the **neighbor** command to configure the following attributes.
   a) Configure a local AS number.

NOTE
A local AS number facilitates the merging of networks by allowing a switch in an acquired network to appear to belong to the former AS. In eBGP deployments, the local AS values must be different for each node.

```
Leaf1(config-bgp-router)# local-as 1
```

   b) Configure a remote AS number for both IPv4 and IPv6 addresses to support BGP sessions to the spine.

```
Leaf1(config-bgp-router)# neighbor 2.6.0.0 remote-as 2
Leaf1(config-bgp-router)# neighbor 1000:2:6:: remote-as 2
Leaf1(config-bgp-router)# neighbor 3.6.0.0 remote-as 2
Leaf1(config-bgp-router)# neighbor 1000:3:6:: remote-as 2
```

3. Enter address-family IPv4 unicast configuration mode and configure the following attributes.
   a) Use the **neighbor allowas-in** command to disable the AS_PATH check function for routes learned from a specified location.

```
Leaf1(config-bgp-router)# address-family ipv4 unicast
Leaf1(config-bgp-ipv4u)# neighbor 2.6.0.0 allowas-in 1
Leaf1(config-bgp-ipv4u)# neighbor 3.6.0.0 allowas-in 1
```

NOTE
This prevents BGP from rejecting routes that contain the recipient BGP speaker's AS number, where *number* here specifies the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

b) Configure the maximum number of paths for ECMP load balancing. (The default is 1.)

```
Leaf1(config-bgp-ipv4u)# maximum-paths 8
```

c) Enable BGP recursive next-hop lookups.

```
Leaf1(config-bgp-ipv4u)# next-hop-recursion
```

d) Enable directly connected routes to be redistributed into BGP.

```
Leaf1(config-bgp-ipv4u)# redistribute connected
```

e) Enable OSPF routes to be redistributed into BGP.

```
Leaf1(config-bgp-ipv4u)# redistribute ospf
```

f) Exit address-family IPv4 unicast configuration mode.

```
Leaf1(config-bgp-ipv4u)# exit
Leaf1(config-bgp-router)#
```

4. Enter address-family IPv6 unicast configuration mode and configure the following attributes.
   a) Activate the interface, enabling the exchange of information with BGP neighbors and peer groups, and use the **neighbor allowas-in** command to disable the AS_PATH check function for routes learned from a specified location.

```
Leaf1(config-bgp-router)# address-family ipv6 unicast
Leaf1(config-bgp-ipv6u)# neighbor 1000:2:6:: activate
Leaf1(config-bgp-ipv6u)# neighbor 1000:2:6:: allowas-in 1
Leaf1(config-bgp-ipv6u)# neighbor 1000:3:6:: activate
Leaf1(config-bgp-ipv6u)# neighbor 1000:3:6:: allowas-in 1
```

   b) Configure the maximum number of paths for ECMP load balancing. (The default is 1.)

```
Leaf1(config-bgp-ipv6u)# maximum-paths 8
```

   c) Enable BGP recursive next-hop lookups.

```
Leaf1(config-bgp-ipv6u)# next-hop-recursion
```

   d) Enable directly connected routes to be redistributed into BGP.

```
Leaf1(config-bgp-ipv6u)# redistribute connected
```

   e) Enable OSPF connected routes to be redistributed into BGP, and exit to BGP global configuration mode.

```
Leaf1(config-bgp-ipv6u)# redistribute ospf
Leaf1(config-bgp-ipv6u)# exit
Leaf1(config-bgp-router)#
```

5. (Optional) Configure neighbor Bidirectional Forwarding Detection (BFD), by means of the **neighbor bfd** command, to facilitate fast recovery in case of a link failure.

```
Leaf1(config-bgp-router)# neighbor 2.6.0.0 bfd
Leaf1(config-bgp-router)# neighbor 3.6.0.0 bfd
```

NOTE
For more information, refer to the "BFD" chapter in the *Network OS Layer 3 Routing Configuration Guide*.

6. (Optional) Enter address-family Layer 2 Virtual Private Network (VPN) Ethernet VPN (EVPN) configuration mode and configure the following attributes for IPv4 addresses.
   a) Activate the interface, enabling the exchange of information with BGP neighbors and peer groups, and use the **neighbor allowas-in** command to disable the AS_PATH check function for routes learned from a specified location.

```
Leaf1(config-bgp-router)# address-family l2vpn evpn
Leaf1(config-evpn)# neighbor 2.6.0.0 activate
Leaf1(config-evpn)# neighbor 2.6.0.0 allowas-in 1
Leaf1(config-evpn)# neighbor 3.6.0.0 activate
Leaf1(config-evpn)# neighbor 3.6.0.0 allowas-in 1
```

   NOTE
   This step is applicable only when BGP EVPN is used for the overlay.

   b) Enable BGP to send an update to an eBGP multihop peer with the next-hop attribute unchanged.

```
Leaf1(config-evpn)# neighbor 2.6.0.0 next-hop-unchanged
Leaf1(config-evpn)# neighbor 3.6.0.0 next-hop-unchanged
```

7. In BGP router configuration mode, configure a BGP instance for IPv4 unicast address-family for VRF red and use the **redistribute connected** command to redistribute connected routes, and do the same for IPv6.

```
Leaf1(config-bgp-router)# address-family ipv4 unicast vrf red
Leaf1(config-bgp-ipv4u-vrf)# redistribute connected

Leaf1(config-bgp-router)# address-family ipv6 unicast vrf red
Leaf1(config-bgp-ipv6u-vrf)# redistribute connected
```

8. Repeat Step 1 through Step 7 on the remaining leaf nodes as appropriate.
9. To verify the configuration, use the **show running-config rbridge-id router bgp** , the **show ip bgp summary** , the **show bgp evpn summary**, and the **show bfd neighbors** commands.
   Refer to Verifying the configuration on page 55 for examples of useful **show** commands.

Configuring the overlay gateway

This task creates an overlay gateway instance and configures a variety of parameters.

1. In global configuration mode, create an overlay gateway instance and enter overlay-gateway configuration mode.

```
Leaf1(config)# overlay-gateway Leaf6
```

2. In overlay-gateway configuration mode, specify the type as **layer2-extension**.

```
Leaf1(config-overlay-gw-Leaf6)# type layer2-extension
```

3. Specify the IP interface as Loopback 1.

```
Leaf1(config-overlay-gw-Leaf6)# ip interface Loopback 1
```

4. Attach the RBridge ID of the current leaf switch.

```
Leaf1(config-overlay-gw-Leaf6)# attach rbridge-id add 1
```

5. (Optional) To support dual-homing, you must do the following.

a) Use the **add** keyword to specify the RBridge IDs of each of the two supporting nodes.

```
Leaf1(config-overlay-gw-Leaf6)# attach rbridge-id add 1-2
```

b) In RBridge ID configuration mode, enter the **ip router-id** command to assign a unique router ID to the first node in the vLAG pair.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip router-id 101.29.29.1
```

c) On the second node in pair, repeat Step 5b.

```
device# configure terminal
device(config)# rbridge-id 2
device(config-rbridge-id-2)# ip router-id 101.31.31.1
```

This is required because only one loopback address can be assigned in this configuration, and the two nodes must use different router IDs to distinguish the BGP sessions.

6. Specify the automatic mapping of VLANs to VNIs.

```
Leaf1(config-overlay-gw-Leaf6)# map vlan vni auto
```

NOTE
This is the preferred method for ease of configuration. If the user does not want to use automatic VLAN VNI mapping, then manual VLAN-to-VNI mapping can be used. This manual mapping is required for Cisco interoperability. Refer to Configuring interoperability with other vendors on page 54.

7. Activate the overlay gateway

```
Leaf1(config-overlay-gw-Leaf6)# activate
```

8. Repeat Step 1 through Step 7, as appropriate, for the remaining leaf nodes.

Configuring the spine switches

This task configures underlay physical connectivity between the spine, leaf, and optional superspine switches, and also sets BGP neighbor and AS attributes.

1. In interface subtype configuration mode, configure an Ethernet interface for point-to-point connectivity between spine switch Spine1 and supported leaf switches.

NOTE
This example uses both IPv4 and IPv6 addresses, with /31 and /127 networks, respectively.

```
Spine1# configure terminal
Spine1(config)# rbridge-id 1
Spine1(config-rbridge-id-1)# interface tengigabitethernet 1/1/33
Spine1(config-if-te-1/1/33)# ip address 1.2.1.1/31
Spine1(config-if-te-1/1/33)# ipv6 adddress 1000:1:2:2::1/127
Spine1(config-if-te-1/1/33)# no shut
Spine1(config-if-te-1/1/33)# exit
Spine1(config-rbridge-id-1)#
```

2. Configure loopback interfaces, to be the donor interfaces used for optional IP unnumbered.

NOTE
This example uses both IPv4 and IPv6 addresses, with /32 and /128 networks, respectively.

```
Spine1(config-rbridge-id-1)# interface loopback 1
Spine1(config-Loopback-1)# ip address 2.0.0.2/32
Spine1(config-Loopback-1)# ipv6 address 1000:2::2/128
```

```
Spine1(config-Loopback-1)# no shut
Spine1(config-Loopback-1)# exit
Spine1(config-rbridge-id-1)#
```

3. (Optional) You can configure unnumbered IPv4 and IPv6 addresses, with an appropriate loopback address, as follows.

```
Leaf1(config-rbridge-id-1)# interface tengigabitethernet 1/1/33
Leaf1(config-if-te-1/1/33)# ip unnumbered loopback 1
Leaf1(config-if-te-1/1/33)# ipv6 unnumbered loopback 1
Leaf1(config-if-te-1/1/33)# no shut
```

4. Repeat Step 1 or Step 2 for all Ethernet interfaces supporting the leaf switches.
5. Enable BGP routing and enter BGP global configuration mode.

```
Spine1(config-rbridge-id-1)# router bgp
Spine1(config-bgp-router)#
```

6. Use the **neighbor** command to configure the following attributes.
   a) Configure a local AS number for the spine switch.

NOTE
A local AS number facilitates the merging of networks by allowing a switch in an acquired network to appear to belong to the former AS. In eBGP deployments, the local AS values must be different for each node.

```
Spine1(config-bgp-router)# local-as 2
```

   b) Configure a remote AS number for both IPv4 and IPv6 addresses to support BGP sessions to the superspine.

```
Spine1(config-bgp-router)# neighbor 1.2.1.0 remote-as 1
Spine1(config-bgp-router)# neighbor 1000:1:2:1:: remote-as 1
```

   c) Configure remote AS numbers for both IPv4 and IPv6 addresses to support BGP sessions to the leaf switches.

```
Spine1(config-bgp-router)# neighbor 2.6.0.1 remote-as 1
Spine1(config-bgp-router)# neighbor 1000:2:6:1:: remote-as 1
```

   d) Repeat Step 6a through Step 6c, as appropriate, for the remaining leaf interfaces. The following shows the results of the configuration.

```
neighbor 2.7.0.1 remote-as 3
neighbor 1000:2:7::1 remote-as 3
neighbor 2.8.0.1 remote-as 3
neighbor 1000:2:8::1 remote-as 3
neighbor 2.9.0.1 remote-as 3
neighbor 1000:2:9::1 remote-as 3
```

7. Enter address-family IPv4 unicast configuration mode and configure the following attributes.
   a) (Optional) Use the **neighbor allowas-in** command to disable the AS_PATH check function for routes learned from a superspine.

NOTE
This prevents BGP from rejecting routes that contain the recipient BGP speaker's AS number, where *number* here specifies the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

```
Spine1(config-bgp-router)# address-family ipv4 unicast
Spine1(config-bgp-ipv4u)# neighbor 1.2.1.0 allowas-in 1
```

   b) Configure the maximum number of paths for ECMP load balancing. (The default is 1.)

```
Spine1(config-bgp-ipv4u)# maximum-paths 8
```

c) Enable BGP recursive next-hop lookups.

```
Spine1(config-bgp-ipv4u)# next-hop-recursion
```

d) (Optional) Enable directly connected routes to be redistributed into BGP.

```
Spine1(config-bgp-ipv4u)# redistribute connected
```

e) (Optional) Enable OSPF routes to be redistributed into BGP.

```
Spine1(config-bgp-ipv4u)# redistribute ospf
```

f) Exit address-family IPv4 unicast configuration mode.

```
Spine1(config-bgp-ipv4u)# exit
Spine1(config-bgp-router)#
```

8. Enter address-family IPv6 unicast configuration mode and configure the following attributes.
   a) Activate the interface, enabling the exchange of information with BGP neighbors and peer groups, and optionally use the **neighbor allowas-in** command to disable the AS_PATH check function for routes learned from a superspine.

```
Spine1(config-bgp-router)# address-family ipv6 unicast
Spine1(config-bgp-ipv6u)# neighbor 1000:1:2:1:: activate
Spine1(config-bgp-ipv6u)# neighbor 1000:1:2:1:: allowas-in 1
```

   b) Configure the maximum number of paths for ECMP load balancing. (The default is 1.)

```
Spine1(config-bgp-ipv6u)# maximum-paths 8
```

   c) Enable BGP recursive next-hop lookups.

```
Spine1(config-bgp-ipv6u)# next-hop-recursion
```

   d) Enable directly connected routes to be redistributed into BGP.

```
Spine1(config-bgp-ipv6u)# redistribute connected
```

   e) Enable OSPF routes to be redistributed into BGP.

```
Spine1(config-bgp-ipv6u)# redistribute ospf
```

   f) Enable BGP recursive next-hop lookups.

```
Spine1(config-bgp-ipv6u)# next-hop-recursion
```

   g) Exit address-family IPv6 unicast configuration mode, and enter BGP configuration mode.

```
Spine1(config-bgp-ipv6u)# exit
Spine1(config-bgp-router)#
```

9. (Optional) Enable Bidirectional Forwarding Detection (BFD) for BGP neighbors, to facilitate fast recovery in case of a link failure.

```
Spine1(config-bgp-router)# neighbor 1.2.1.0 bfd
Spine1(config-bgp-router)# neighbor 1000:1:2:1:: bfd
```

   Repeat Step 9 for all IPv4 and IPv6 neighbors.

10. (Optional) Enter address-family Layer 2 Virtual Private Network (VPN) Ethernet VPN (EVPN) configuration mode and configure the following attributes for IPv4 addresses.
    a) Activate the interface, enabling the exchange of information with BGP neighbors and peer groups, and use the **neighbor allowas-in** command to disable the AS_PATH check function for routes learned from a specified location.

```
Spine1(config-bgp-router)# address-family l2vpn evpn
```

```
Spine1(config-evpn)# neighbor 1.2.1.0 activate
Spine1(config-evpn)# neighbor 1.2.1.0 allowas-in 1
```

b) Enable BGP to send an update to an eBGP multihop peer with the next-hop attribute unchanged.

```
Spine1(config-evpn)# neighbor 1.2.1.0 next-hop-unchanged
```

c) Repeat Step 10a through Step 10b, as appropriate, for the remaining IPv4 leaf interfaces. The following shows the results of the configuration.

```
neighbor 2.6.0.1 activate
neighbor 2.6.0.1 next-hop-unchanged
neighbor 2.7.0.1 activate
neighbor 2.7.0.1 next-hop-unchanged
neighbor 2.8.0.1 activate
neighbor 2.8.0.1 next-hop-unchanged
neighbor 2.9.0.1 activate
neighbor 2.9.0.1 next-hop-unchanged
```

11. Enter the **retain-route-target-all** under BGP EVPN address-family configuration mode to configure the route reflector (RR) to accept all route targets (RTs), preventing filtering on routes that do not match the local configuration on the spine.

```
Spine1(config-bgp-router)# address-family l2vpn evpn
Spine1(config-bgp-evpn)# retain route-target all
```

12. (Optional) Where iBGP is used, use the **neighbor update-source** command to default to the loopback address as a backup interface in case of a link failure.

```
Spine1(config-bgp-router)# neighbor 2.0.0.2 update-source loopback 1
Spine1(config-bgp-router)# neighbor 1000:2::2 update-source loopback 1
```

13. Repeat Step 1 through Step 12, as appropriate, for the remaining interfaces.
14. Repeat Step 1 through Step 13, as appropriate, for Spine2.
15. To verify the configuration, use the **show ip bgp summary** , the **show bgp evpn summary** , and the **show bfd neighbors** commands.

    Refer to Verifying the configuration on page 55 for examples of useful **show** commands.

(Optional) Configuring a BGP EVPN instance

A BGP EVPN instance (EVI) can be configured and various commands can be executed in EVPN instance configuration mode, as shown in the following configuration example.

NOTE
BGP EVPN is not essential to the configuration of an IP Fabric, and is one among a variety of overlay solutions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **evpn-instance** command and specify a name to configure an EVPN instance and enter EVPN instance configuration mode.

```
device(config-rbridge-id-1)# evpn-instance myinstance
```

4. Enter the **rd auto** command to enable auto-generation of the RD value.

```
device(config-evpn-instance-myinstance)# rd auto
```

5. Enter the **route-target** command using the **both** and **auto** parameters to configure auto-generation of the import and export route-target community attributes globally.

```
device(config-evpn-instance-myinstance)# route-target both auto
```

6. Enter the **vni** command with the **add** *value* parameter to add VLANs to the EVI.

```
device(config-evpn-instance-myinstance)# vni add 1-100
```

7. (Optional) Enter the **duplicate-mac-timer** *interval* command with the **max-count** *interval* parameter to set the duplicate MAC detection timer interval and the maximum count.

```
device(config-evpn-instance-myinstance)# duplicate-mac-timer 22 max-count 5
```

The following example configures the EVPN instance “myinstance” so that the RD value is generated automatically. The import and export route-target community attributes are also generated automatically. VLANs are added to the EVI. The duplicate MAC detection timer is set to 22 seconds and the number of times a MAC move can be detected in the configured interval before the MAC is suppressed is set to 5.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# evpn-instance myinstance
device(config-evpn-instance-myinstance)# rd auto
device(config-evpn-instance-myinstance)# route-target both auto
device(config-evpn-instance-myinstance)# vni add 1-100
device(config-evpn-instance-myinstance)# duplicate-mac-timer 22 max-count 5
```

(Optional) Configuring support for dual-homed servers

This optional task illustrates the configuration of connectivity between a leaf node (in a pair of supporting nodes) and a server over a port-channel vLAG where high availability (HA) is required through multihoming. This configuration must also be applied to the peer (supporting) leaf node in the HA pair. Both nodes are considered a *vLAG pair*.

This task involves the following:

- Use the **vlag ignore-split** command where a vLAG spans more than one node. This minimizes packet loss if one of the nodes goes down, and also reduces vLAG failover downtime.
- The **switchport** commands set the Layer 2 characteristics of the interface.
- Ethernet Segment Identifiers (ESIs) are used to identify the links connecting multiple ToR devices to a server in a multihomed BGP EVPN environment. As part of the port-channel configuration, this task uses the **esi** command to specify a nondefault Ethernet Segment Identifier (ESI) for the interface if LACP autodiscovery is not deployed. The default ESI value, for single homing, is 0. See Ethernet Segment Identifiers for BGP routing on page 27 in this chapter.
- Ensure that spanning tree is disabled. (Spanning tree is disabled by default.)

1. From global configuration mode, configure a router ID for the first node in the supporting pair.

```
device# config terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip router-id 101.29.29.1
device(config-rbridge-id-1)# exit
device(config)#
```

2. From global configuration mode, specify a port-channel interface and enter port-channel configuration mode.

```
device(config)# interface port-channel 1
device(config-Port-channel-1)#
```

3. Use the **vlag ignore-split** command to minimize packet loss.

```
device(config-Port-channel-1)# vlag ignore-split
```

4. Configure Layer 2 characteristics and specify a VLAN.
   a) Put the interface in Layer 2 mode.

```
device(config-Port-channel-1)# switchport
```

   b) Use the **switchport mode access** command to specify the mode as "access," enabling the port to be assigned to a VLAN.

```
device(config-Port-channel-1)# switchport mode access
```

   c) Use the **switchport access vlan** command to specify a VLAN. (Your requirements will vary.)

```
device(config-Port-channel-1)# switchport access vlan 500
```

5. (Optional) Use the **spanning-tree shutdown** command to ensure that spanning tree (disabled by default) is disabled.

```
device(config-Port-channel-1)# spanning-tree shutdown
```

6. To support optional BGP EVPN, use the **esi auto lacp** command to specify a nondefault Ethernet Segment Identifier that is automatically generated.

NOTE
Where ESI values are assigned manually, the same ESI number must not be repeated on other port channels. You can use the **esi** command to assign a value manually.

```
device(config-Port-channel-1)# esi auto lacp
```

7. Enable the port-channel interface.

```
device(config-Port-channel-1)# no shut
```

8. Repeat Step 1 through Step 7 on the peer leaf node of the HA (vLAG) pair, using the **ip router-id** command to assign a unique router ID as in Step 1.

```
device# config terminal
device(config)# rbridge-id 2
device(config-rbridge-id-2)# ip router-id 101.31.31.1
device(config-rbridge-id-2)# exit
```

Configuring a superspine switch

This task configures a superspine switch that can be used to interconnect one datacenter pod to another. This example uses OSPF as the routing protocol for the interconnect.

1. In interface subtype configuration mode, configure an Ethernet interface for point-to-point connectivity between superspine switch SuperSpine and supported spine switches.

NOTE
This example uses both IPv4 and IPv6 addresses, with /31 and /127 networks, respectively.

```
SuperSpine# configure terminal
SuperSpine(config)# rbridge-id 1
SuperSpine(config-rbridge-id-1)# interface fortygigabitethernet 1/4/11
SuperSpine(config-if-fo-1/4/11)# ip address 1.4.0.0/31
SuperSpine(config-if-fo-1/4/11)# ipv6 adddress 1000:1::/127
SuperSpine(config-if-fo-1/4/11)# no shut
SuperSpine(config-if-fo-1/4/11)# exit
SuperSpine(config-rbridge-id-1)#
```

2. Configure a loopback interface.

NOTE
This example uses both IPv4 and IPv6 addresses, with /32 and /128 networks, respectively.

```
SuperSpine(config-rbridge-id-1)# interface loopback 1
SuperSpine(config-Loopback-1)# ip address 1.0.0.1/32
SuperSpine(config-Loopback-1)# ipv6 address 1000:1::1/128
SuperSpine(config-Loopback-1)# no shut
SuperSpine(config-Loopback-1)# exit
SuperSpine(config-rbridge-id-1)#
```

3. (Optional) You can configure unnumbered IPv4 and IPv6 addresses, with an appropriate loopback address, as follows.

```
Leaf1(config-rbridge-id-1)# interface fortygigabitethernet 1/4/11
Leaf1(config-if-fo-1/4/11)# ip unnumbered loopback 1
Leaf1(config-if-fo-1/4/11)# ipv6 unnumbered loopback 1
Leaf1(config-if-fo-1/4/11)# no shut
```

4. Repeat Step 1 and Step 3, as appropriate, for all Ethernet interfaces supporting the spine switches.
5. Enable BGP routing and enter BGP global configuration mode.
```
SuperSpine(config-rbridge-id-1)# router bgp
SuperSpine(config-bgp-router)#
```

6. Configure a local AS for the superspine switch.
```
SuperSpine(config-bgp-router)# local-as 1
```

NOTE
A local AS number facilitates the merging of networks by allowing a switch in an acquired network to appear to belong to the former AS. In eBGP deployments, the local AS values must be different for each node.

7. Use the **neighbor** command to configure the following attributes.
   a) Configure a remote AS number for both IPv4 and IPv6 addresses to support BGP sessions to the spine switches.
```
SuperSpine(config-bgp-router)# neighbor 1.2.1.1 remote-as 2
SuperSpine(config-bgp-router)# neighbor 1000:1:2:1:: remote-as 2
SuperSpine(config-bgp-router)# neighbor 1.3.1.1 remote-as 2
SuperSpine(config-bgp-router)# neighbor 1000:1:5:1:: remote-as 2
```

8. Enter address-family IPv4 unicast configuration mode and configure the following attributes.
   a) Configure the maximum number of paths for ECMP load balancing. (The default is 1.)
```
SuperSpine(config-bgp-ipv4u)# maximum-paths 8
```
   b) Enable BGP recursive next-hop lookups.
```
SuperSpine(config-bgp-ipv4u)# next-hop-recursion
```

c) Enable directly connected routes to be redistributed into BGP.

```
SuperSpine(config-bgp-ipv4u)# redistribute connected
```

d) Enable OSPF connected routes to be redistributed into BGP, and exit to BGP configuration mode.

```
Leaf1(config-bgp-ipv4u)# redistribute ospf
Leaf1(config-bgp-ipv4u)# exit
Leaf1(config-bgp-router)#
```

9. Enter address-family IPv6 unicast configuration mode and configure the following attributes.
   a) Activate the interface, enabling the exchange of information with BGP neighbors and peer groups.

```
SuperSpine(config-bgp-router)# address-family ipv6 unicast
SuperSpine(config-bgp-ipv6u)# neighbor 1000:1:2:1::1 activate
SuperSpine(config-bgp-ipv6u)# neighbor 1000:1:3:1::1 activate
```

   b) Configure the maximum number of paths for ECMP load balancing. (The default is 1.)

```
SuperSpine(config-bgp-ipv6u)# maximum-paths 8
```

   c) Enable BGP recursive next-hop lookups.

```
SuperSpine(config-bgp-ipv6u)# next-hop-recursion
```

   d) Enable directly connected routes to be redistributed into BGP.

```
SuperSpine(config-bgp-ipv6u)# redistribute connected
```

   e) Enable OSPF connected routes to be redistributed into BGP, and exit to BGP configuration mode.

```
SuperSpine(config-bgp-ipv6u)# redistribute ospf
SuperSpine(config-bgp-ipv6u)# exit
SuperSpine(config-bgp-router)#
```

10. (Optional) Enter address-family Layer 2 Virtual Private Network (VPN) Ethernet VPN (EVPN) configuration mode and configure the following attributes for IPv4 addresses.
    a) Activate the interface, enabling the exchange of information with BGP neighbors and peer groups.

```
SuperSpine(config-bgp-router)# address-family l2vpn evpn
SuperSpine(config-evpn)# neighbor 1.2.1.0 activate
```

    b) Enable BGP to send an update to an eBGP multihop peer with the next-hop attribute unchanged.

```
SuperSpine(config-evpn)# neighbor 1.2.1.0 next-hop-unchanged
```

    c) Repeat Step 10a through Step 10b, as appropriate, for the remaining IPv4 leaf interfaces.
11. Enter the **retain-route-target-all** under BGP EVPN address-family configuration mode to configure the route reflector (RR) to accept all route targets (RTs), preventing filtering on routes that do not match the local configuration on the spine.

```
SuperSpine(config-bgp-router)# address-family l2vpn evpn
SuperSpine(config-bgp-evpn)# retain route-target all
```

12. (Optional) Configure neighbor Bidirectional Forwarding Detection (BFD), by means of the **neighbor bfd** command, to facilitate fast recovery in case of a link failure.

```
SuperSpine(config-bgp-router)# neighbor 1.2.1.1 bfd
SuperSpine(config-bgp-router)# neighbor 1000:1:2:1:: bfd
SuperSpine(config-bgp-router)# neighbor 1.3.1.1 bfd
SuperSpine(config-bgp-router)# neighbor 1000:1:3:1:: bfd
```

13. Repeat Step 1 through Step 9, as appropriate, for a peer superspine switch.

Configuring interoperability with other vendors

When BGP EVPN is used, the automatic mapping of VLANs to VNIs may not work with all vendors.

Some vendors do not use the range of available VNIs (1 through 8191) that Brocade does. As a result of the automatic mapping process that is initiated by the **map vlan vni auto** command in overlay-gateway configuration mode (entered by means of the **overlay-gateway** command), a vendor's VLANs that do not conform to the Brocade extended VLAN range must be remapped manually to their respective VNIs.

In addition, the route target (RT) and route distinguisher (RD) values are set automatically. Because such route targets use AS numbers in the administrator field, routes cannot be exchanged between leaf nodes belonging to different AS numbers. In such cases manual adjustments are required, as illustrated in this task.

1. From global configuration mode, enter overlay-gateway configuration mode and specify an overlay gateway.
```
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)#
```
2. In overlay-gateway configuration mode, do the following.
   a) Use the the **map vlan** command to map a VLAN to a VNI manually, as in the following example.
```
device(config-overlay-gw-gatweway1)# map vlan 100 vni 10100
```
   b) Repeat Step 2a as appropriate for additional manual mappings.
3. In overlay-gateway configuration mode, use the the **map vlan** command to map a VLAN to a VNI manually as in the following example.
4. In RBridge ID configuration mode, enter EVPN configuration mode.
```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# evpn-instance myinstance
device(config-evpn-instance-myinstance)#
```
5. In EVPN instance configuration mode, use the **route-target (EVPN)** command to specify auto-generation of the import and export route-target community attributes and ignore the ASN, and the **rd auto (EVPN)** command to enable auto-generation of a route distinguisher (RD) for an EVPN instance.
```
device(config-evpn-instance-myinstance)# route-target both auto ignore-as
device(config-evpn-instance-myinstance)# rd auto
```
6. In EVPN instance configuration mode, configure the VPN route distinguisher (RD) manually.
   a) Enter the **vni (EVPN)** command to specify the VNI and enter VNI instance configuration mode, where you use the **rd (VNI)** command and the **route-target (VNI)** command to specify manually the RD and RT, respectively.
```
device(config-evpn-instance-myinstance)# vni 10100
device(evpn-vni-10100)# rd 100:100
device(evpn-vni-10100)# route-target import 1:1
device(evpn-vni-10100)# route-target export 1:1
```
   b) Repeat Step 6a for all RDs and RTs that need to be mapped manually.

Verifying the configuration

This section presents a variety of **show** commands that can be used to verify configurations on a leaf or spine.

Verifying configurations on a leaf

To verify the Ethernet configuration, use the **show running-config interface** command, as in the following example.

```
Leaf1# show running-config interface TenGigabitEthernet 1/0/16
interface TenGigabitEthernet 1/0/16
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 22-25
 switchport trunk tag native-vlan
 spanning-tree shutdown
 fabric isl enable
 fabric trunk enable
 no shutdown
```

To verify a VLAN configuration, use the **show running-config interface vlan** command, as in the following example.

```
Leaf1# show running-config interface vlan 25
interface Vlan 25
 suppress-nd
 suppress-arp
```

To verify a virtual Ethernet (VE) configuration, use the **show running-config interface rbridge-id interface ve** command, as in the following example.

```
Leaf1# show running-config rbridge-id 1 interface ve 25
rbridge-id 1
interface Ve 25
 vrf forwarding vrf2
 ipv6 anycast-address 2:25:1::254/64
 ip anycast-address 2.25.1.254/24
 no shutdown
```

To verify IPv4 or IPv6 static-anycast-gateway configurations, use the **show running-config rbridge-id ip anycast-gateway-mac** command or the **show running-config rbridge-id ipv6 anycast-gateway-mac** command, respectively, as in the following examples.

```
Leaf1# show running-config rbridge-id 1 ip anycast-gateway-mac
rbridge-id 1
ip anycast-gateway-mac 0000.abba.abba
Leaf1# show running-config rbridge-id 1 ipv6 anycast-gateway-mac
rbridge-id 1
ipv6 anycast-gateway-mac 0000.abba.abba
```

To verify the entire BGP configuration, use the **show running-config rbridge-id router bgp** command, as in the following examples.

```
Leaf1# show running-config rbridge-id 1 router bgp
rbridge-id 1
router bgp
  local-as 65006
  neighbor 1000:2:6:: remote-as 65002
  neighbor 1000:2:6:: password 2 $TDk1UHNkRC1afDg=
  neighbor 1000:3:6:: remote-as 65002
  neighbor 1000:3:6:: password 2 $TDk1UHNkRC1afDg=
  neighbor 2.6.0.0 remote-as 65002
  neighbor 2.6.0.0 password 2 $TDk1UHNkRC1afDg=
  neighbor 2.6.0.0 bfd
  neighbor 3.6.0.0 remote-as 65002
  neighbor 3.6.0.0 password 2 $TDk1UHNkRC1afDg=
  neighbor 3.6.0.0 bfd
  address-family ipv4 unicast

   redistribute connected
```

```
   neighbor 3.6.0.0 allowas-in 1
   neighbor 2.6.0.0 allowas-in 1
   maximum-paths 8
   next-hop-recursion
  !
  address-family ipv4 unicast vrf vrf12
   redistribute connected
   maximum-paths 8
  !
  address-family ipv4 unicast vrf vrf13
   redistribute connected
   maximum-paths 8
  !
  <---output omitted--->
  !
  address-family ipv4 unicast vrf vrf8
   redistribute connected
   maximum-paths 8
  !
  address-family ipv4 unicast vrf vrf9
   redistribute connected
   maximum-paths 8
  !
  address-family ipv6 unicast

   redistribute connected
   neighbor 1000:3:6:: allowas-in 1
   neighbor 1000:3:6:: activate
   neighbor 1000:2:6:: allowas-in 1
   neighbor 1000:2:6:: activate
   maximum-paths 8
   next-hop-recursion
  !
  address-family ipv6 unicast vrf vrf12
   redistribute connected
   maximum-paths 8
  !
  address-family ipv6 unicast vrf vrf13
   redistribute connected
   maximum-paths 8
  !
   <---output omitted--->
  !
  address-family ipv6 unicast vrf vrf8
   redistribute connected
   maximum-paths 8
  !
  address-family ipv6 unicast vrf vrf9
   redistribute connected
   maximum-paths 8
  !
  address-family l2vpn evpn
   neighbor 3.6.0.0 activate
   neighbor 3.6.0.0 allowas-in 1
   neighbor 3.6.0.0 next-hop-unchanged
   neighbor 2.6.0.0 activate
   neighbor 2.6.0.0 allowas-in 1
   neighbor 2.6.0.0 next-hop-unchanged
```

To view summary BGP information, use the **show ip bgp summary** command, as in the following example.

```
Leaf1# show ip bgp summary
  BGP4 Summary
  Router ID: 6.0.0.6   Local AS Number: 65006
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 8
  Number of Neighbors Configured: 2, UP: 1
  Number of Routes Installed: 17, Uses 2040 bytes
  Number of Routes Advertising to All Neighbors: 19 (16 entries), Uses 960 bytes
  Number of Attribute Entries Installed: 6, Uses 690 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent
ToSend
  2.6.0.0           65002       CONN      1h23m33s    0        0        0        16
  3.6.0.0           65002       ESTAB    23h28m32s    14       0        3        0
```

To verify the BGP EVPN configuration, use the **show bgp evpn summary** command, as in the following example.

```
Leaf1# show bgp evpn summary
  BGP4 Summary
  Router ID: 6.0.0.6   Local AS Number: 65006
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 2, UP: 1
  Number of Routes Installed: 4536, Uses 544320 bytes
  Number of Routes Advertising to All Neighbors: 5652 (4536 entries), Uses 272160
bytes
  Number of Attribute Entries Installed: 2327, Uses 267605 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent
ToSend
  2.6.0.0           65002       CONN      1h23m39s    0        0        0        4536
  3.6.0.0           65002       ESTAB    23h28m37s    3420     24       1116     0
```

To verify the BFD neighbors configuration, use the **show bfd neighbors** command, as in the following example.

```
Leaf1# show bfd neighbors
Flags: * indicates State is inconsistent across the cluster
OurAddr                                 NeighAddr                       State
Int             Rbridge-id
=======                                 =========                       =====
===             ==========
3.6.0.1                                 3.6.0.0                         UP        Te
1/0/5        1
```

Verifying configurations on a spine

To verify the BGP configuration, use the **show ip bgp summary** command, as in the following example.

```
Spine2# show ip bgp summary
  BGP4 Summary
  Router ID: 3.0.0.3   Local AS Number: 65002
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 8
  Number of Neighbors Configured: 5, UP: 5
  Number of Routes Installed: 22, Uses 2640 bytes
  Number of Routes Advertising to All Neighbors: 70 (16 entries), Uses 960 bytes
  Number of Attribute Entries Installed: 7, Uses 805 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent
ToSend
  1.3.0.0           65001       ESTAB    23h14m16s    4        0        13       0
  3.6.0.1           65006       ESTAB    23h30m19s    3        0        14       0
  3.7.0.1           65078       ESTAB    23h21m35s    3        0        15       0
  3.8.0.1           65078       ESTAB    23h21m47s    3        0        14       0
  3.9.0.1           65009       ESTAB    23h24m15s    3        0        14       0
```

To verify the BGP EVPN configuration, use the **show bgp evpn summary** command, as in the following example.

```
Spine2# show bgp evpn summary
  BGP4 Summary
  Router ID: 3.0.0.3   Local AS Number: 65002
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 5, UP: 5
  Number of Routes Installed: 4536, Uses 544320 bytes
  Number of Routes Advertising to All Neighbors: 18144 (4536 entries), Uses 272160
bytes
  Number of Attribute Entries Installed: 2200, Uses 253000 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent
ToSend
  1.3.0.0           65001       ESTAB    23h14m37s    0        0        4536     0
  3.6.0.1           65006       ESTAB    23h30m40s    1116     0        3420     0
  3.7.0.1           65078       ESTAB    23h21m57s    1152     0        3384     0
  3.8.0.1           65078       ESTAB    23h22m 9s    1152     0        3384     0
  3.9.0.1           65009       ESTAB    23h24m36s    1116     0        3420     0
```

To verify the BFD neighbors configuration, use the **show bfd neighbors** command, as in the following example.

```
Spine2# show bfd neighbors
Flags: * indicates State is inconsistent across the cluster
OurAddr                                 NeighAddr
State     Int             Rbridge-id
=======                                 =========
=====     ===             ==========
3.6.0.0                                 3.6.0.1
UP        Te 1/1/5        1
```

Configuring additional features for IP Fabrics

This section presents tasks that enhance the implementation and management of IP Fabrics. The configuration of BGP EVPN, although demonstrated as the overlay implementation shown in this guide, is not essential to the deployment of an IP Fabric.

Configuring MAC learning of Layer 2 extension site through BGP

The user can choose the way MAC learning is achieved for a Layer 2 extension tunnel.

BGP routing must be configured.

Data plane (Layer 2) MAC address learning is enabled by default at the remote site. However, this leads to scalability issues. Where BGP routing is used, do the following to enable MAC learning by means of BGP instead. This delegates the responsibility for MAC learning on a tunnel to the Layer 3 control-plane protocol, such as BGP EVPN.

1. Enter VXLAN overlay-gateway site configuration mode.
   ```
   device# config terminal
   device(config)# overlay-gateway gateway1
   device(config-overlay-gw-gateway1)# site mysite
   device(config-overlay-gw-gateway1-site-mysite)#
   ```
2. Enter the **mac-learning protocol bgp** command.
   ```
   device(config-overlay-gw-gateway1-site-mysite)# mac-learning protocol bgp
   ```
3. To disable BGP MAC learning and return to the default of Layer 2 learning, use the **no mac-learning protocol bgp** command.
   ```
   device(config-overlay-gw-gateway1-site-mysite)# no mac-learning protocol bgp
   ```

Disabling automatic VXLAN tunnel endpoint discovery by BGP

Automatic VXLAN tunnel endpoint (VTEP) discovery by BGP is enabled by default and can be disabled by means of the following procedure.

1. Enter the **configure terminal** command to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.
   ```
   device(config)# rbridge-id 1
   ```
3. Enter the **router bgp** command to enable BGP routing.
   ```
   device(config-rbridge-id-1)# router bgp
   ```

```
device(config-bgp-router)#
```

4. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)#
```

5. Enter the **no vtep-discovery** command to disable automatic VTEP discovery by BGP.

```
device(config-bgp-evpn)# no vtep-discovery
```

The following example disables automatic VTEP discovery by BGP.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# no vtep-discovery
```

Configuring BGP next hop unchanged

BGP next hop unchanged can be configured for the L2VPN EVPN address family, allowing BGP to send updates to eBGP peers with the next hop attribute unchanged. The **neighbor next-hop-unchanged** command should be configured for the Spine switch for each EVPN neighbor if the leaf switch is an eBGP peer. The **neighbor next-hop-unchanged** command should be configured for the leaf switch for each EVPN neighbor if the spine switch is an eBGP peer.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **neighbor** *ip address* **activate** command to establish a BGP EVPN session with the eBGP neighbor.

```
device(config-bgp-evpn)# neighbor 10.1.1.1 activate
```

8. Enter the **neighbor** *ip address* **neighbor next-hop-unchanged** command to configure BGP next hop unchanged.

```
device(config-bgp-evpn)# neighbor 10.1.1.1 next-hop-unchanged
```

The following example establishes a BGP EVPN session with an eBGP neighbor and configures BGP next hop unchanged so that updates can be sent to the neighbor with the next hop attribute unchanged.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.1.1 activate
device(config-bgp-evpn)# neighbor 10.1.1.1 next-hop-unchanged
```

Configuring BGP retain route target

BGP retain route target can be configured for the L2VPN EVPN address family so that a route reflector (RR) accepts all route targets (RTs). For iBGP, enable this feature on a spine switch so that route-target filtering is not performed and all route targets are retained and route-target attributes in the EVPN routes are not modified or removed.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **retain route-target all** command to configure the RR to accept all route targets RTs.

```
device(config-bgp-evpn)# retain route-target all
```

The following example configures a RR to accept all RTs.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all
```

Applying a BGP extended community filter for the L2VPN EVPN address family

A BGP extended community filter can be applied for the L2VPN EVPN address family.

BGP communities must already be defined. For more information on defining BGP communities, refer to the “BGP4” and “BGP4+” chapters in the *Network OS Layer 3 Routing Configuration Guide*.

1. Enter the **configure terminal** command to access global configuration mode.
```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.
```
device(config)# rbridge-id 1
```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.
```
device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 123:2
```

4. Enter the **router bgp** command to enable BGP routing.
```
device(config-rbridge-id-1)# router bgp
```

5. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.
```
device(config-bgp-router)# local-as 1000
```

6. Enter the **neighbor** *ip-address* **remote-as** command to specify the ASN in which the remote neighbor resides.
```
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
```

7. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.
```
device(config-bgp-router)# address-family l2vpn evpn
```

8. Enter the **neighbor** *ip-address* **activate** command to enable the exchange of information with the neighbor.
```
device(config-bgp-evpn)# neighbor 10.1.2.3 activate
```

9. Enter the **neighbor** *ip-address* **route-map** command and specify the **in** keyword to apply a route map to incoming routes.
```
device(config-bgp-evpn)# neighbor 10.1.2.3 route-map in extComRmapt
```

10. Enter the **neighbor** *ip-address* **route-map** command and specify the **out** keyword to apply a route map to outgoing routes.
```
device(config-bgp-evpn)# neighbor 10.1.2.3 route-map out sendExtComRmap
```

11. Enter the **neighbor** *ip-address* **send-community** command and specify the **both** keyword to enable the sending of standard and extended attributes in updates to the specified BGP neighbor.
```
device(config-bgp-evpn)# neighbor 10.1.2.3 send-community both
```

The following example applies a BGP extended community filter for the L2VPN EVPN address family. The steps for configuring a route map are not included in this example.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 123:2
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.2.3 activate
device(config-bgp-evpn)# neighbor 10.1.2.3 route-map in extComRmapt
device(config-bgp-evpn)# neighbor 10.1.2.3 route-map out sendExtComRmap
device(config-bgp-evpn)# neighbor 10.1.2.3 send-community both
```

Configuring BGP graceful restart for the L2VPN EVPN address family

Graceful restart can be configured for BGP EVPN in the L2VPN EVPN address family configuration mode, helping to retain peer routes and ensure that no route and topology changes occur in the network for the duration of a restart.

1. Enter the **configure terminal** command to access global configuration mode.
```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.
```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.
```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.
```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ip address* **remote-as** command to specify the ASN in which the neighbor resides.
```
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.
```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-evpn)# graceful-restart
```

8. Do any of the following:

   - Enter the **graceful-restart** command and use the **purge-time** parameter to overwrite the default purge time value.

     ```
     device(config-bgp-evpn)# graceful-restart purge-time 300
     ```

   - Enter the **graceful-restart** command and use the **restart-time** parameter to overwrite the default restart time advertised to graceful restart-capable neighbors.

     ```
     device(config-bgp-evpn)# graceful-restart restart-time 180
     ```

   - Enter the **graceful-restart** command and use the **stale-routes-time** parameter to overwrite the default amount of time that a helper device will wait for an EOR message from a peer.

     ```
     device(config-bgp-evpn)# graceful-restart stale-routes-time 100
     ```

The following example enables the graceful restart feature.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart
```

The following example enables the graceful restart feature and sets the purge time to 300 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart purge-time 180
```

The following example enables the graceful restart feature and sets the restart time to 180 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart restart-time 180
```

The following example enables the graceful restart feature and sets the stale-routes time to 100 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart stale-routes-time 100
```

Configuring BGP peer groups for the L2VPN EVPN address family

A peer group can be created and activated in the L2VPN EVPN address family configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-1)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 peer-group
   ```

6. Enter the **neighbor** *peer-group-name* **remote-as** command to specify the ASN of the peer group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
   ```

7. Enter the **neighbor** *ipv6-address* **peer-group** command to associate a neighbor with the peer group.

   ```
   device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
   ```

8. Enter the **neighbor** *ipv6-address* **peer-group** command to associate a second neighbor with the peer group.

   ```
   device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
   ```

9. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

   ```
   device(config-bgp-router)# address-family l2vpn evpn
   ```

10. Enter the **neighbor** *peer-group-name* **activate** command to establish a BGP EVPN session with the peer group.

    ```
    device(config-bgp-evpn)# neighbor mypeergroup1 activate
    ```

The following example creates a peer group, specifying two neighbors to belong to the peer group, and activates the peer group in L2VPN EVPN.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor mypeergroup1 activate
```

Configuring a route reflector client for the L2VPN EVPN address family

A BGP peer can be configured as a route reflector (RR) client for the L2VPN EVPN address family. When iBGP is used for BGP EVPN in an IP Fabric, spine switches are configured as RRs and leaf switches are configured as RR clients. The following task configures an RR client.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-1)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
   ```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

   ```
   device(config-bgp-router)# address-family l2vpn evpn
   ```

7. Enter the **neighbor** *ip address* **route-reflector-client** command to configure a specified neighbor to be a route reflector client.

   ```
   device(config-bgp-evpn)# neighbor 10.1.1.1 route-reflector-client
   ```

The following example configures a neighbor with the IP address 10.1.1.1 to be a route reflector client in L2VPN EVPN configuration mode.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.1.1 route-reflector-client
```

Disabling client-to-client reflection for the L2VPN EVPN address family

For iBGP, if the leaf switches are configured as route reflector clients and are connected in a full iBGP mesh, you can disable client-to-client reflection on the spine switch that is configured as a route reflector (RR).

1. Enter the **configure terminal** command to access global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.
   ```
   device(config)# rbridge-id 1
   ```
3. Enter the **router bgp** command to enable BGP routing.
   ```
   device(config-rbridge-id-1)# router bgp
   ```
4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.
   ```
   device(config-bgp-router)# local-as 1000
   ```
5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.
   ```
   device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
   ```
6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.
   ```
   device(config-bgp-router)# address-family l2vpn evpn
   ```
7. Enter the **no client-to-client-reflection** command to disable client-to-client reflection.
   ```
   device(config-bgp-evpn)# no client-to-client-reflection
   ```

The following example disables client-to-client reflection in L2VPN EVPN configuration mode.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# no client-to-client-reflection
```

Disabling the BGP AS_PATH check function for the L2VPN EVPN address family

A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected. For eBGP, the AS_PATH check function can be disabled for a Leaf switch so that route updates from the Spine layer are not discarded by the Leaf.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv6-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 2001:db8:e0ff:783a::4 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **neighbor** *ipv6 address* **activate** command to establish a BGP EVPN session with the eBGP neighbor.

```
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 activate
```

8. Enter the **neighbor** *ipv6-address* **allowas-in** command and specify a **number** to disable the BGP AS_PATH check function, and specify the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

```
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 allowas-in 1
```

This example specifies that the AS path of a received route may contain the recipient BGP speaker's AS number once and still be accepted in L2VPN EVPN.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:e0ff:783a::4 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 allowas-in 1
```

Appendix A: Supported BGP EVPN commands

Configuration commands supporting BGP EVPN

The following configuration commands support BGP EVPN.

For more details, refer to the *Network OS Command Reference*.

RBridge ID configuration mode

- **evpn-instance**

BGP configuration mode

- **address-family l2vpn evpn**

BGP address-family L2VPN EVPN configuration mode

- **client-to-client-reflection**
- **graceful-restart (BGP)**
- **neighbor activate**
- **neighbor allowas-in**
- **neighbor maximum-prefix**
- **neighbor next-hop-unchanged**
- **neighbor route-map**
- **neighbor route-reflector-client**
- **neighbor send-community**
- **retain route-target all**
- **vtep-discovery**

EVPN instance configuration mode

- **duplicate-mac-timer**
- **rd auto (EVPN)**
- **route-target (EVPN)**
- **vni (EVPN)**

VNI configuration mode

- **rd (VNI)**
- **route-target (VNI)**

VRF configuration mode

- **rd (VRF)**
- **route-target (VRF)**
- **vni (VRF)**

Port-channel configuration mode

- **esi**

VXLAN overlay-gateway site configuration mode

- **mac-learning protocol bgp**

BGP EVPN show commands

The following show commands support BGP EVPN.

For more details, refer to the *Network OS Command Reference*.

- **show bgp evpn dampened-routes**
- **show bgp evpn interface port-channel**
- **show bgp evpn interface tunnel**
- **show bgp evpn l2route detail**
- **show bgp evpn l2route next-hop**
- **show bgp evpn l2route summary**
- **show bgp evpn l2route type arp**
- **show bgp evpn l2route type auto-discovery**
- **show bgp evpn l2route type ethernet-segment**
- **show bgp evpn l2route type inclusive-multicast**
- **show bgp evpn l2route type mac**
- **show bgp evpn l2route type nd**
- **show bgp evpn l2route unreachable**
- **show bgp evpn l3vni**
- **show bgp evpn neighbors**
- **show bgp evpn neighbors advertised-routes detail**
- **show bgp evpn neighbors advertised-routes type**
- **show bgp evpn neighbors routes best**
- **show bgp evpn neighbors routes detail**
- **show bgp evpn neighbors routes not-installed-best**
- **show bgp evpn neighbors routes type**
- **show bgp evpn neighbors routes unreachable**
- **show bgp evpn neighbors routes-summary**
- **show bgp evpn routes**
- **show bgp evpn routes best**
- **show bgp evpn routes detail**
- **show bgp evpn routes local**
- **show bgp evpn routes next-hop**
- **show bgp evpn routes no-best**

- **show bgp evpn routes not-installed-best**
- **show bgp evpn routes rd**
- **show bgp evpn routes rd type**
- **show bgp evpn routes summary**
- **show bgp evpn routes type arp**
- **show bgp evpn routes type auto-discovery**
- **show bgp evpn routes type ethernet-segment**
- **show bgp evpn routes type inclusive-multicast**
- **show bgp evpn routes type ipv4-prefix**
- **show bgp evpn routes type ipv6-prefix**
- **show bgp evpn routes type mac**
- **show bgp evpn routes type nd**
- **show bgp evpn routes unreachable**
- **show bgp evpn summary**

Appendix B: Sample BGP EVPN configuration files

Sample BGP EVPN superspine configuration using eBGP

The following example is a sample BGP EVPN superspine configuration using eBGP in an IP Fabric. The **neighbor remote-as** command is used so that the switches are configured to be in one autonomous system (AS). The switches are configured to advertise routes to other eBGP peers leaving the next-hop attribute unchanged using the **neighbor next-hop-unchanged** command.

NOTE
This configuration sample demonstrates BGP EVPN configuration using eBGP on the superspine switch in an IP Fabric. For full configuration details for configuring an IP Fabric involving all the steps required, refer to the Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay on page 37 section. For more details on the commands used in this sample, refer to the *Network OS Command Reference*. Refer to the section eBGP-based BGP EVPN on page 20 for more information on eBGP-based BGP EVPN.

```
router bgp
 local-as 1
! IPv4/IPv6 BGP Sessions to Spine switches
 neighbor 10.2.1.1 remote-as 2
 neighbor 10.3.1.1 remote-as 2
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
address-family l2vpn evpn
 retain route-target all
! Activate EVPN Session to Spine switches
 neighbor 10.2.1.1 activate
 neighbor 10.3.1.1 activate
 neighbor 10.2.1.1 next-hop-unchanged
 neighbor 10.3.1.1 next-hop-unchanged
```

Sample BGP EVPN spine configuration using eBGP

The following example is a sample BGP EVPN spine configuration using eBGP in an IP Fabric. There are two spine switches, spine 1 and spine 2. The spine switches are configured to be in one autonomous system (AS) using the **neighbor remote-as** command and are configured to advertise

routes to other eBGP peers leaving the next-hop attribute unchanged using the **neighbor next-hop-unchanged** command.

NOTE
This configuration sample demonstrates BGP EVPN configuration using eBGP on two spine switches in an IP Fabric. For full configuration details for configuring an IP fabric involving all the steps required, refer to the Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay on page 37 section. For more details on the commands used in this sample, refer to the *Network OS Command Reference*.

Spine 1

```
router bgp
 local-as 2
!BGP Sessions to the SuperSpine
 neighbor 10.2.1.0 remote-as 1
!IPv4/IPv6 BGP Sessions to the leaf switches
 neighbor 10.6.0.1 remote-as 3
 neighbor 10.7.0.1 remote-as 3
 neighbor 10.8.0.1 remote-as 3
 neighbor 10.9.0.1 remote-as 3
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
 neighbor 10.2.1.0 allowas-in 1
address-family l2vpn evpn
 retain route-target all
 neighbor 10.2.1.0 activate
 neighbor 10.2.1.0 allowas-in 1
 neighbor 10.2.1.0 next-hop-unchanged
 neighbor 10.6.0.1 activate
 neighbor 10.6.0.1 next-hop-unchanged
 neighbor 10.7.0.1 activate
 neighbor 10.7.0.1 next-hop-unchanged
 neighbor 10.8.0.1 activate
 neighbor 10.8.0.1 next-hop-unchanged
 neighbor 10.9.0.1 activate
 neighbor 10.9.0.1 next-hop-unchanged
```

Spine 2

```
router bgp
 local-as 2
!BGP Sessions to the SuperSpine
 neighbor 10.3.1.0 remote-as 1
!IPv4/IPv6 BGP Sessions to the leaf switches
 neighbor 10.6.1.1 remote-as 3
 neighbor 10.7.1.1 remote-as 3
 neighbor 10.8.1.1 remote-as 3
 neighbor 10.9.1.1 remote-as 3
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
 neighbor 10.3.1.0 allowas-in 1
address-family l2vpn evpn
 retain route-target all
 neighbor 10.3.1.0 activate
 neighbor 10.3.1.0 allowas-in 1
 neighbor 10.3.1.0 next-hop-unchanged
 neighbor 10.6.1.1 activate
 neighbor 10.6.1.1 next-hop-unchanged
 neighbor 10.7.1.1 activate
 neighbor 10.7.1.1 next-hop-unchanged
 neighbor 10.8.1.1 activate
 neighbor 10.8.1.1 next-hop-unchanged
 neighbor 10.9.1.1 activate
 neighbor 10.9.1.1 next-hop-unchanged
```

Sample BGP EVPN leaf configuration using eBGP

The following example is a sample BGP EVPN leaf configuration using eBGP in an IP Fabric. The AS_PATH check function is disabled for the leaf switch using the **neighbor allowas-in** command so that route updates from the spine layer are not discarded by the leaf.

NOTE
This configuration sample demonstrates BGP EVPN configuration using eBGP on a leaf switch in an IP Fabric. For full configuration details for configuring an IP fabric involving all the steps required, refer to the Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay on page 37 section. For more details on the commands used in this sample, refer to the *Network OS Command Reference*.

```
evpn-instance default
 rd auto
 route-target both auto
 vni add 101-108
 vni add 121-128
 vni add 141-148
 vni add 161-168
 vni add 120
 vni add 140
 vni add 160
 vni add 180
!
router bgp
 local-as 3
neighbor 10.6.0.0 remote-as 2
neighbor 10.7.0.0 remote-as 2
 !
 address-family ipv4 unicast
  maximum-paths 8
  next-hop-recursion
  redistribute connected
  neighbor 10.6.0.0 allowas-in 1
  neighbor 10.7.0.0 allowas-in 1
 !
 address-family l2vpn evpn
  neighbor 10.6.0.0 activate
  neighbor 10.7.0.0 activate
  neighbor 10.6.0.0 allowas-in 1
  neighbor 10.7.0.0 allowas-in 1
  neighbor 10.6.0.0 next-hop-unchanged
  neighbor 10.7.0.0 next-hop-unchanged
 !
overlay-gateway Leaf6
 type layer2-extension
 ip interface Loopback 1
 attach rbridge-id add 57
 map vlan vni auto
 activate
```

Sample BGP EVPN superspine configuration using iBGP

The following example is a sample BGP EVPN superspine configuration using iBGP in an IP Fabric.

NOTE
This configuration sample demonstrates BGP EVPN configuration using iBGP on the superspine in an IP Fabric. For full configuration details for configuring an IP Fabric involving all the steps required, refer to the Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay on page 37 section. For more

details on the commands used in this sample, refer to the *Network OS Command Reference*. Refer to the section iBGP-based BGP EVPN on page 21 for more information on eBGP-based BGP EVPN.

```
router ospf
 area 0
 redistribute connected
!
router bgp
 local-as 1
! IPv4/IPv6 BGP Sessions to Spine switches
 neighbor 10.4.1.1 remote-as 3
 neighbor 10.5.1.1 remote-as 3
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
address-family l2vpn evpn
 retain route-target all
! Activate EVPN Session to Spine switches
 neighbor 10.4.1.1 activate
 neighbor 10.5.1.1 activate
 neighbor 10.4.1.1 next-hop-unchanged
 neighbor 10.5.1.1 next-hop-unchanged
int fo 100/4/11
 ip address 10.4.0.0/31
 ipv6 address 1000:1:4::/127
 ip ospf area 0
 no shutdown
 !
int fo 100/4/12
 ip address 10.5.0.0/31
 ipv6 address 1000:1:5::/127
 ip ospf area 0
 no shutdown
```

Sample BGP EVPN spine configuration using iBGP

The following example is a sample BGP EVPN spine configuration using iBGP in an IP Fabric. The spine switch is configured as a route reflector (RR) and is configured to accept received updates containing at least one route target. Neighboring leaf switches are configured as route reflector (RR) clients using the **route-reflector-client** command.

NOTE
This configuration sample demonstrates BGP EVPN configuration using iBGP on a spine switch in an IP Fabric. For full configuration details for configuring an IP Fabric involving all the steps required, refer to the Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay on page 37 section. For more details on the commands used in this sample, refer to the *Network OS Command Reference*.

```
router ospf
 area 0
 redistribute connected
!
router bgp
 local-as 3
 cluster-id ipv4-address 10.4.4.4
!BGP Sessions to the SuperSpine
 neighbor 10.4.1.0 remote-as 1
!IPv4/IPv6 BGP Sessions to the leaf switches
 neighbor 10.0.0.10 remote-as 3
 neighbor 10.0.0.10 update-source loopback 1
 neighbor 10.0.0.11 remote-as 3
```

```
 neighbor 10.0.0.11 update-source loopback 1
 neighbor 10.0.0.12 remote-as 3
 neighbor 10.0.0.12 update-source loopback 1
 neighbor 10.0.0.13 remote-as 3
 neighbor 10.0.0.13 update-source loopback 1
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
 client-to-client-reflection
 neighbor 10.4.1.0 allowas-in 1
 neighbor 10.0.0.10 route-reflector-client
 neighbor 10.0.0.11 route-reflector-client
 neighbor 10.0.0.12 route-reflector-client
 neighbor 10.0.0.13 route-reflector-client
address-family l2vpn evpn
 retain route-target all
 neighbor 10.4.1.0 activate
 neighbor 10.4.1.0 allowas-in 1
 neighbor 10.4.1.0 next-hop-unchanged
 neighbor 10.0.0.10 activate
 neighbor 10.0.0.10 next-hop-unchanged
 neighbor 10.0.0.11 activate
 neighbor 10.0.0.11 next-hop-unchanged
 neighbor 10.0.0.12 activate
 neighbor 10.0.0.12 next-hop-unchanged
 neighbor 10.0.0.13 activate
 neighbor 10.0.0.13 next-hop-unchanged
!
```

Sample BGP EVPN leaf configuration using iBGP

The following example is a sample BGP EVPN leaf configuration using iBGP in an IP Fabric. The leaf switch is configured to communicate with a neighbor through a specified loopback interface.

NOTE
This configuration sample demonstrates BGP EVPN configuration using iBGP on a leaf in an IP Fabric. For full configuration details for configuring an IP Fabric involving all the steps required, refer to the Configuring a Brocade IP Fabric with Optional BGP EVPN Overlay on page 37 section. For more details on the commands used in this sample, refer to the *Network OS Command Reference*.

```
evpn-instance default
 rd auto
 route-target both auto ignore-as
 vlan add 101-108
 vlan add 121-128
 vlan add 141-148
 vlan add 161-168
 vlan add 120
 vlan add 140
 vlan add 160
 vlan add 180
!
router bgp
 local-as 3
neighbor 10.0.0.4 remote-as 3
neighbor 10.0.0.4 update-source loopback 1
neighbor 10.0.0.5 remote-as 3
neighbor 10.0.0.5 update-source loopback 1
 !
 address-family ipv4 unicast
  maximum-paths 8
  next-hop-recursion
  redistribute connected
 !
 address-family l2vpn evpn
  neighbor 10.0.0.4 activate
```

```
  neighbor 10.0.0.5 activate
  neighbor 10.0.0.4 next-hop-unchanged
  neighbor 10.0.0.5 next-hop-unchanged
 !
router ospf
 area 0
 redistribute connected
!
overlay-gateway Leaf10
 type layer2-extension
 ip interface Loopback 1
 attach rbridge-id add 76
 map vlan vni auto
 activate
int te 76/0/1
 ip address 10.10.0.1/31
 ipv6 address 1000:4:10::1/127
 ip ospf area 0
 no shut
!
int te 76/0/2
 ip address 10.10.0.2/31
 ipv6 address 1000:5:10::1/127
 ip ospf area 0
 no shut
!
int te 76/0/16
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 101-108
 switchport trunk allowed vlan add 121-128
 no shut
!
int te 76/0/17
 switchport trunk allowed vlan add 141-148
 switchport trunk allowed vlan add 161-168
 no shut
!
```

Appendix C: Sample Topology Configuration Files

Leaf

The following configuration file is Leaf1.cfg.

```
int vlan 101-108
int vlan 120
int vlan 121-128
int vlan 140
int vlan 141-148
int vlan 160
int vlan 161-168
int vlan 180
rbridge-id 1
 switch-attributes host-name Leaf1
ip anycast-gateway-mac 0000.ABBA.BABA
ipv6 anycast-gateway-mac 0000.ABBA.ABBA
vrf red
 rd 6.0.0.6:1
 vni 120
 address-family ipv4 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
 address-family ipv6 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
!
vrf blue
 rd 6.0.0.6:2
 vni 140
 address-family ipv4 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
 address-family ipv6 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
!
vrf green
 rd 6.0.0.6:3
 vni 160
 address-family ipv4 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
 address-family ipv6 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
!
vrf yellow
 rd 6.0.0.6:4
 vni 180
 address-family ipv4 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
 address-family ipv6 unicast
  route-target import 2:2 evpn
  route-target export 2:2 evpn
!
!!!!!Loopback interface
int loopback 1
 ip address 6.0.0.6/32
```

```
 ipv6 address 1000:6::6/128
 no shut
!
int ve 101
 vrf forwarding red
 ip anycast-address 101.1.1.254/24
 ipv6 anycast-address 101:1:1::254/64
 no shutdown
 !
int ve 102
 vrf forwarding red
 ip anycast-address 102.1.1.254/24
 ipv6 anycast-address 102:1:1::254/64
 no shutdown
 !
int ve 103
 vrf forwarding red
 ip anycast-address 103.1.1.254/24
 ipv6 anycast-address 103:1:1::254/64
 no shutdown
 !
int ve 104
 vrf forwarding red
 ip anycast-address 104.1.1.254/24
 ipv6 anycast-address 104:1:1::254/64
 no shutdown
 !
int ve 105
 vrf forwarding red
 ip anycast-address 105.1.1.254/24
 ipv6 anycast-address 105:1:1::254/64
 no shutdown
 !
int ve 106
 vrf forwarding red
 ip anycast-address 106.1.1.254/24
 ipv6 anycast-address 106:1:1::254/64
 no shutdown
 !
int ve 107
 vrf forwarding red
 ip anycast-address 107.1.1.254/24
 ipv6 anycast-address 107:1:1::254/64
 no shutdown
 !
int ve 108
 vrf forwarding red
 ip anycast-address 108.1.1.254/24
 ipv6 anycast-address 108:1:1::254/64
 no shutdown
 !
int ve 121
 vrf forwarding blue
 ip anycast-address 121.1.1.254/24
 ipv6 anycast-address 121:1:1::254/64
 no shutdown
 !
int ve 122
 vrf forwarding blue
 ip anycast-address 122.1.1.254/24
 ipv6 anycast-address 122:1:1::254/64
 no shutdown
 !
int ve 123
 vrf forwarding blue
 ip anycast-address 123.1.1.254/24
 ipv6 anycast-address 123:1:1::254/64
 no shutdown
 !
int ve 124
 vrf forwarding blue
 ip anycast-address 124.1.1.254/24
 ipv6 anycast-address 124:1:1::254/64
 no shutdown
 !
int ve 125
 vrf forwarding blue
 ip anycast-address 125.1.1.254/24
```

```
 ipv6 anycast-address 125:1:1::254/64
 no shutdown
 !
int ve 126
 vrf forwarding blue
 ip anycast-address 126.1.1.254/24
 ipv6 anycast-address 126:1:1::254/64
 no shutdown
 !
int ve 127
 vrf forwarding blue
 ip anycast-address 127.1.1.254/24
 ipv6 anycast-address 127:1:1::254/64
 no shutdown
 !
int ve 128
 vrf forwarding blue
 ip anycast-address 128.1.1.254/24
 ipv6 anycast-address 128:1:1::254/64
 no shutdown
 !
int ve 141
 vrf forwarding green
 ip anycast-address 141.1.1.254/24
 ipv6 anycast-address 141:1:1::254/64
 no shutdown
 !
int ve 142
 vrf forwarding green
 ip anycast-address 142.1.1.254/24
 ipv6 anycast-address 142:1:1::254/64
 no shutdown
 !
int ve 143
 vrf forwarding green
 ip anycast-address 143.1.1.254/24
 ipv6 anycast-address 143:1:1::254/64
 no shutdown
 !
int ve 144
 vrf forwarding green
 ip anycast-address 144.1.1.254/24
 ipv6 anycast-address 144:1:1::254/64
 no shutdown
 !
int ve 145
 vrf forwarding green
 ip anycast-address 145.1.1.254/24
 ipv6 anycast-address 145:1:1::254/64
 no shutdown
 !
int ve 146
 vrf forwarding green
 ip anycast-address 146.1.1.254/24
 ipv6 anycast-address 146:1:1::254/64
 no shutdown
 !
int ve 147
 vrf forwarding green
 ip anycast-address 147.1.1.254/24
 ipv6 anycast-address 147:1:1::254/64
 no shutdown
 !
int ve 148
 vrf forwarding green
 ip anycast-address 148.1.1.254/24
 ipv6 anycast-address 148:1:1::254/64
 no shutdown
 !
int ve 161
 vrf forwarding yellow
 ip anycast-address 161.1.1.254/24
 ipv6 anycast-address 161:1:1::254/64
 no shutdown
 !
int ve 162
 vrf forwarding yellow
 ip anycast-address 162.1.1.254/24
```

```
 ipv6 anycast-address 162:1:1::254/64
 no shutdown
 !
int ve 163
 vrf forwarding yellow
 ip anycast-address 163.1.1.254/24
 ipv6 anycast-address 163:1:1::254/64
 no shutdown
 !
int ve 164
 vrf forwarding yellow
 ip anycast-address 164.1.1.254/24
 ipv6 anycast-address 164:1:1::254/64
 no shutdown
 !
int ve 165
 vrf forwarding yellow
 ip anycast-address 165.1.1.254/24
 ipv6 anycast-address 165:1:1::254/64
 no shutdown
 !
int ve 166
 vrf forwarding yellow
 ip anycast-address 166.1.1.254/24
 ipv6 anycast-address 166:1:1::254/64
 no shutdown
 !
int ve 167
 vrf forwarding yellow
 ip anycast-address 167.1.1.254/24
 ipv6 anycast-address 167:1:1::254/64
 no shutdown
 !
int ve 168
 vrf forwarding yellow
 ip anycast-address 168.1.1.254/24
 ipv6 anycast-address 168:1:1::254/64
 no shutdown
 !
int ve 120
 vrf forwarding red
    ipv6 address use-link-local-only
 no shut
!
int ve 140
 vrf forwarding blue
    ipv6 address use-link-local-only
 no shut
!
int ve 160
 vrf forwarding green
    ipv6 address use-link-local-only
 no shut
!
int ve 180
 vrf forwarding yellow
    ipv6 address use-link-local-only
 no shut
!
evpn-instance myinstance
 rd auto
 route-target both auto ignore-as
 vlan add 101-108
 vlan add 121-128
 vlan add 141-148
 vlan add 161-168
 vlan add 120
 vlan add 140
 vlan add 160
 vlan add 180
!
router bgp
 local-as 3
 neighbor 2.6.0.0 remote-as 2
 neighbor 1000:2:6:: remote-as 2
 neighbor 3.6.0.0 remote-as 2
 neighbor 1000:3:6:: remote-as 2
 !
```

```
 address-family ipv4 unicast
  maximum-paths 8
  next-hop-recursion
  redistribute connected
  neighbor 2.6.0.0 allowas-in 1
  neighbor 3.6.0.0 allowas-in 1
 !
 address-family ipv6 unicast
  maximum-paths 8
  next-hop-recursion
  redistribute connected
  neighbor 1000:2:6:: activate
  neighbor 1000:3:6:: activate
  neighbor 1000:2:6:: allowas-in 1
  neighbor 1000:3:6:: allowas-in 1
 !
 address-family l2vpn evpn
  neighbor 2.6.0.0 activate
  neighbor 3.6.0.0 activate
  neighbor 2.6.0.0 allowas-in 1
  neighbor 3.6.0.0 allowas-in 1
  neighbor 2.6.0.0 next-hop-unchanged
  neighbor 3.6.0.0 next-hop-unchanged
 !
address-family ipv4 unicast vrf red
 redistribute connected
address-family ipv4 unicast vrf blue
 redistribute connected
address-family ipv4 unicast vrf green
 redistribute connected
address-family ipv4 unicast vrf yellow
 redistribute connected
address-family ipv6 unicast vrf red
 redistribute connected
address-family ipv6 unicast vrf blue
 redistribute connected
address-family ipv6 unicast vrf green
 redistribute connected
address-family ipv6 unicast vrf yellow
 redistribute connected
overlay-gateway Leaf6
 type layer2-extension
 ip interface Loopback 1
 attach rbridge-id add 1
 map vlan vni auto
 activate
int te 1/0/1
 ip address 2.6.0.1/31
 ipv6 address 1000:2:6::1/127
 no shut
!
int te 1/0/5
 ip address 3.6.0.1/31
 ipv6 address 1000:3:6::1/127
 no shut
!
int te 1/0/16
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 101-108
 switchport trunk allowed vlan add 121-128
 no shut
!
int te 1/0/17
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 141-148
 switchport trunk allowed vlan add 161-168
 no shut
!
```

Spine

The following configuration file is Spine1.cfg.

```
rbridge-id 1
 switch-attributes host-name Spine1
!!!!!Loopback interface
int loopback 1
 ip address 2.0.0.2/32
 ipv6 address 1000:2::2/128
 no shut
!
router bgp
 local-as 2
!BGP Sessions to the SuperSpine
 neighbor 1.2.1.0 remote-as 1
 neighbor 1000:1:2:1:: remote-as 1
!IPv4/IPv6 BGP Sessions to the leaf switches
 neighbor 2.6.0.1 remote-as 3
 neighbor 1000:2:6::1 remote-as 3
 neighbor 2.7.0.1 remote-as 3
 neighbor 1000:2:7::1 remote-as 3
 neighbor 2.8.0.1 remote-as 3
 neighbor 1000:2:8::1 remote-as 3
 neighbor 2.9.0.1 remote-as 3
 neighbor 1000:2:9::1 remote-as 3
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
 neighbor 1.2.1.0 allowas-in 1
address-family ipv6 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
 neighbor 1000:1:2:1:: activate
 neighbor 1000:1:2:1:: allowas-in 1
 neighbor 1000:2:6::1 activate
 neighbor 1000:2:7::1 activate
 neighbor 1000:2:8::1 activate
 neighbor 1000:2:9::1 activate
address-family l2vpn evpn
 retain route-target all
 neighbor 1.2.1.0 activate
 neighbor 1.2.1.0 allowas-in 1
 neighbor 1.2.1.0 next-hop-unchanged
 neighbor 2.6.0.1 activate
 neighbor 2.6.0.1 next-hop-unchanged
 neighbor 2.7.0.1 activate
 neighbor 2.7.0.1 next-hop-unchanged
 neighbor 2.8.0.1 activate
 neighbor 2.8.0.1 next-hop-unchanged
 neighbor 2.9.0.1 activate
 neighbor 2.9.0.1 next-hop-unchanged
!
int te 1/1/33
 ip address 1.2.1.1/31
 ipv6 address 1000:1:2:1::1/127
 no shutdown
 !
int te 1/1/34
 ip address 1.2.2.1/31
 ipv6 address 1000:1:2:2::1/127
 no shutdown
 !
int te 1/1/35
 ip address 1.2.3.1/31
 ipv6 address 1000:1:2:3::1/127
 no shutdown
 !
int te 1/1/36
 ip address 1.2.4.1/31
 ipv6 address 1000:1:2:4::1/127
 no shutdown
 !
```

```
int te 1/1/37
 ip address 1.2.5.1/31
 ipv6 address 1000:1:2:5::1/127
 no shutdown
 !
int te 1/1/1
 ip address 2.6.0.0/31
 ipv6 address 1000:2:6::0/127
 no shutdown
 !
int te 1/1/4
 ip address 2.7.0.0/31
 ipv6 address 1000:2:7::0/127
 no shutdown
 !
int te 1/1/3
 ip address 2.8.0.0/31
 ipv6 address 1000:2:8::0/127
 no shutdown
 !
int te 1/1/2
 ip address 2.9.0.0/31
 ipv6 address 1000:2:9::0/127
 no shutdown
 !
```

SuperSpine

The following configuration file is SuperSpine.cfg.

```
rbridge-id 1
 switch-attributes host-name SuperSpine
!!!!!Loopback interface
int loopback 1
 ip address 1.0.0.1/32
 ipv6 address 1000:1::1/128
 no shut
!
router bgp
 local-as 1
! IPv4/IPv6 BGP Sessions to Spine switches
 neighbor 1.2.1.1 remote-as 2
 neighbor 1.3.1.1 remote-as 2
 neighbor 1000:1:2:1::1 remote-as 2
 neighbor 1000:1:3:1::1 remote-as 2
address-family ipv4 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
address-family ipv6 unicast
 maximum-paths 8
 next-hop-recursion
 redistribute connected
 neighbor 1000:1:2:1::1 activate
 neighbor 1000:1:3:1::1 activate
address-family l2vpn evpn
 retain route-target all
! Activate EVPN Session to Spine switches
 neighbor 1.2.1.1 activate
 neighbor 1.3.1.1 activate
 neighbor 1.2.1.1 next-hop-unchanged
 neighbor 1.3.1.1 next-hop-unchanged
int te 1/1/1
 ip address 1.2.1.0/31
 ipv6 address 1000:1:2:1::/127
 no shutdown
 !
int te 1/5/1
 ip address 1.3.1.0/31
 ipv6 address 1000:1:3:1::/127
 no shutdown
 !
```