BWCAMPUSNETZ

Zukunftsfähige Konzepte für die Campusnetze
an Universitäten und Hochschulen

Einsatz von Open Source in virtualisierten Netzen

Federführung bei der Erstellung dieses Dokuments:
Universität Stuttgart, Karlsruher Institut für Technologie
Kontakt: team@bwcampusnetz.de

Inhalt

Abbildungsverzeichnis

1 Einleitung

EVPN hat sich als moderne Overlay-Netzwerktechnologie etabliert, um skalierbare, flexible und herstellerunabhängige Netzwerke zu realisieren. In Service-Provider-Umgebungen wird EVPN für die Koordination und MPLS für den Datentransport eingesetzt. Besonders in Rechenzentren und Campus-Netzwerken wird EVPN-VXLAN zunehmend als Standard für Layer-2- und Layer-3-Virtualisierung eingesetzt, wobei die Daten über die VXLAN-Tunneltechnologie transportiert werden.

In diesem Dokument werden die Open-Source EVPN-VXLAN Implementierungen FRRouting (FRR) und GoBGP sowie die Interop-Konfiguration mit bzw. zwischen unterschiedlichen Herstellern vorgestellt. Im ersten Teil werden Open-Source EVPN Route-Reflektor Lösungen präsentiert. Trotz des Open-Source Schwerpunkts wird zum Vergleich gezeigt, wie eine entsprechende Konfiguration unter Cisco Catalyst/Nexus sowie Arista aussieht, um die Unterschiede und Gemeinsamkeiten zu verdeutlichen.

Im zweiten Teil des Dokuments wird die Implementierung einer Anycast Gateway (AGW) sowie Centralized Gateway (CGW) Lösung mit FRR beschrieben.

Ziel dieser Dokumentation ist es, eine praxisnahe Einführung in Open-Source EVPN-VXLAN zu geben. Grundkenntnisse der zugrundeliegenden Konzepte und Terminologie werden dabei vorausgesetzt.

2 Route-Reflektor

Die folgende Abbildung zeigt die Topologie bestehend aus zwei Leafs sowie einem Route Reflektor. Dabei peeren die beiden Leafs 1 und 2 per internal BGP (iBGP) innerhalb AS 64512 mit einem Route Reflektor. Vereinfacht wird hier ein CGW angenommen, bei welchem Leaf 2 das Centralized Gateway darstellt. An die beiden Leafs sind die Clients 1 und 2 in den VLANs 50 und 51 angeschlossen, wobei deren Adressen an den Route Reflektor geadvertised werden.

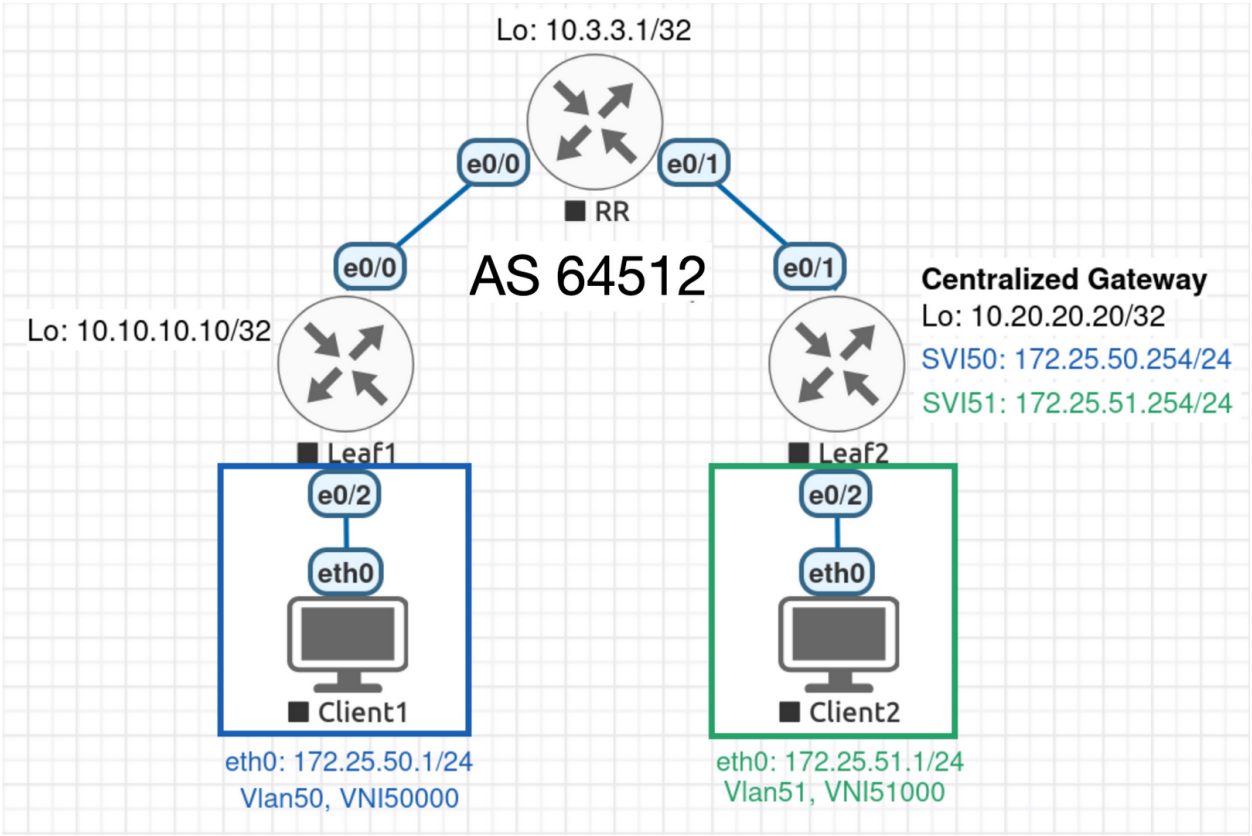

Abb. 2.1: Route Reflektor Topologie

Gerät	Interface	Adresse
Client1	eth0	172.25.50.1/24
Client2	eth0	172.25.51.1/24
Leaf1	Lo	10.10.10.10/32
Leaf2	Lo	10.20.20.20/32
Leaf2	SVI50	172.25.50.254/24
Leaf2	SVI51	172.25.51.254/24
RR	Lo	10.3.3.1/32

Tab. 2.1: Route Reflektor: Interfacekonfiguration

Vlan	VNI	L2VNI/L3VNI	BUM Replication Type
50	50000	L2VNI	Ingress Replication
51	51000	L2VNI	Ingress Replication

Tab. 2.2: Route Reflektor: VLANs

2.1 FRRouting

2.1.1 Konfiguration

In der aktuellen FRRouting Konfiguration wird eine Kombination aus VRF-basiertem Routing, OSPF und BGP EVPN für das Overlay-Netzwerk genutzt.

Die Datei `/etc/frr/daemons` ist die zentrale Konfigurationsdatei für FRRouting, in der festgelegt wird, welche FRR-Dienste (Daemons) beim Start des Systems aktiviert werden und mit welchen Optionen sie laufen. Diese Datei bestimmt, welche Routing-Protokolle aktiv sind und wie sich FRR verhält.

Im Folgenden werden dabei der BGP und OSPFv2 Daemon aktiviert. Darüber hinaus wird mittels `vtysh_enable` eine einheitliche Kommandozeilen Schnittstelle aktiviert. So kann nach der Eingabe von `vtysh` im Configure-Mode, welchen man mittels `configure terminal` erreicht, die Konfiguration vorgenommen werden.

Zebra ist dabei das zentrale Modul, welches die Verwaltung der Routing-Tabellen sowie die Kommunikation zwischen den verschiedenen Routing-Daemons und dem Betriebssystem-Kernel übernimmt. Dabei fungiert Zebra als Routing-Manager, welcher die von BGP, OSPF und anderen Protokollen generierten Routen verarbeitet und an das Betriebssystem weitergibt.

```
# /etc/frr/daemons

bgpd=yes
ospfd=yes
...
vtysh_enable=yes
zebra_options="  -A 127.0.0.1 -s 90000000"
mgmtd_options="  -A 127.0.0.1"
bgpd_options="   -A 127.0.0.1"
ospfd_options="  -A 127.0.0.1"
...
```

Damit der Spine die Daten weiterleitet, muss explizit das Forwarding eingeschaltet werden. Dies kann über die integrierte CLI innerhalb von FRR im Configuration Mode konfiguriert werden mittels:

```
ip forwarding
ipv6 forwarding
```

Daraufhin werden die folgenden `sysctl` Kernelparameter konfiguriert:

```
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
```

Ebenfalls über `vtysh` wird die Interfacekonfiguration sowie die Konfiguration von OSPF und BGP vorgenommen:

```
interface e0/0
 ip address <ip-addr>
 ip ospf area 0
 ip ospf network point-to-point
exit
!
interface e0/1
 ip address <ip-addr>
 ip ospf area 0
 ip ospf network point-to-point
exit
!
interface lo
 ip address <ip-addr>
 ip ospf area 0
exit
!
router ospf
 ospf router-id 10.3.3.1
exit
```

Das Peering zwischen dem Route Reflektor sowie den Leafs geschieht über die Loopback-adressen. Als (Subsequent) Addressfamilie ((S)AFI) wird L2VPN EVPN konfiguriert:

```
router bgp 64512
 bgp router-id 10.3.3.1
 neighbor 10.10.10.10 remote-as 64512
 neighbor 10.10.10.10 update-source lo
 neighbor 10.20.20.20 remote-as 64512
 neighbor 10.20.20.20 update-source lo
 !
 address-family l2vpn evpn
  neighbor 10.10.10.10 activate
  neighbor 10.10.10.10 route-reflector-client
  neighbor 10.20.20.20 activate
  neighbor 10.20.20.20 route-reflector-client
 exit-address-family
exit
```

2.1.2 Verifikation

Die L2VPN EVPN BGP Tabelle kann ausgegeben werden mittels des Befehls `show bgp l2vpn evpn`:

```
frr-rr# show bgp l2vpn evpn
...
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 10.10.10.10:50
 *>i[2]:[0]:[48]:[4a:b2:0e:ba:db:b7]
                    10.10.10.10              0    100      0 ?
                    RT:64512:50000 ET:8
 *>i[2]:[0]:[48]:[4a:b2:0e:ba:db:b7]:[32]:[172.25.50.1]
                    10.10.10.10              0    100      0 ?
                    RT:64512:50000 ET:8
...
```

2.2 GoBGP

2.2.1 Konfiguration

GoBGP kann als reiner BGP Route-Reflektor verwendet werden. Im Gegensatz zu FRR, welches eine breitere Unterstützung für verschiedene Routing-Protokolle, wie OSPF bietet, konzentriert sich GoBGP in diesem Szenario ausschließlich auf die Funktion als BGP Route-Reflektor. Wenn neben BGP auch andere Protokolle wie OSPF benötigt werden, kann FRR ergänzend eingesetzt werden.

Im Folgenden wird die Konfiguration für GoBGP dargestellt. Diese befindet sich in der Datei `/etc/gobgp/gobgpd.conf`.

```
[global.config]
  as = 64512
  router-id = "10.3.3.1"

[[neighbors]]
  [neighbors.config]
    neighbor-address = "10.10.10.10"
    peer-as = 64512
  [neighbors.route-reflector.config]
    route-reflector-client = true
    route-reflector-cluster-id = "10.3.3.1"
  [[neighbors.afi-safis]]
    [neighbors.afi-safis.config]
      afi-safi-name = "l2vpn-evpn"

[[neighbors]]
  [neighbors.config]
    neighbor-address = "10.20.20.20"
    peer-as = 64512
  [neighbors.route-reflector.config]
    route-reflector-client = true
    route-reflector-cluster-id = "10.3.3.1"
  [[neighbors.afi-safis]]
    [neighbors.afi-safis.config]
      afi-safi-name = "l2vpn-evpn"
```

2.2.2 Verifikation

Die BGP Sessions lassen sich wie folgt anzeigen:

```
root@go-rr:~# gobgp neighbor -a evpn
Peer           AS   Up/Down  State        |      #Received Accepted
10.10.10.10 64512  06:19:21 Establ        |        12         12
10.20.20.20 64512  06:19:25 Establ        |        17         17
```

Im Folgenden kann die L2VPN EVPN BGP Tabelle wie folgt ausgegeben werden:

```
root@go-rr:~# gobgp global rib -a evpn
*> [type:macadv][rd:10.10.10.10:50][etag:0][mac:4a:b2:0e:ba:db:b7]
   [ip:172.25.50.1] ...
*> [type:macadv][rd:10.10.10.10:50][etag:0][mac:4a:b2:0e:ba:db:b7]
   [ip:<nil>] ...
```

2.3 Closed Source

Des Weiteren wird im Folgenden beispielhaft die Konfiguration geschlossener (Closed-Source) Lösungen der Hersteller Cisco (Catalyst, Nexus) sowie Arista (EOS) dargestellt. Dies dient dem Vergleich und der besseren Einordnung der Open-Source-Alternativen im praktischen Einsatz.

2.3.1 Cisco Catalyst

```
router bgp 64512
  bgp router-id 10.3.3.1
  neighbor 10.10.10.10 remote-as 64512
  neighbor 10.10.10.10 update-source Loopback1
  neighbor 10.20.20.20 remote-as 64512
  neighbor 10.20.20.20 update-source Loopback1
  !
  address-family l2vpn evpn
   neighbor 10.10.10.10 activate
   neighbor 10.10.10.10 send-community both
   neighbor 10.10.10.10 route-reflector-client
   neighbor 10.20.20.20 activate
   neighbor 10.20.20.20 send-community both
   neighbor 10.20.20.20 route-reflector-client
 exit-address-family
```

2.3.2 Cisco Nexus

```
router bgp 64512
  router-id 10.3.3.1
  address-family l2vpn evpn
  neighbor 10.10.10.10
    remote-as 64512
    update-source loopback1
    address-family l2vpn evpn
      send-community
      send-community extended
      route-reflector-client
  neighbor 10.20.20.20
    remote-as 64512
    update-source loopback1
    address-family l2vpn evpn
      send-community
      send-community extended
      route-reflector-client
```

2.3.3 Arista

```
router bgp 64512
   router-id 10.3.3.1
   no bgp default ipv4-unicast
   rd auto
   neighbor 10.10.10.10 remote-as 64512
   neighbor 10.10.10.10 update-source Loopback1
   neighbor 10.10.10.10 send-community standard large
   neighbor 10.10.10.10 route-reflector-client
   neighbor 10.20.20.20 remote-as 64512
   neighbor 10.20.20.20 update-source Loopback1
   neighbor 10.20.20.20 send-community standard large
   neighbor 10.20.20.20 route-reflector-client
   !
   address-family evpn
      neighbor 10.10.10.10 activate
      neighbor 10.20.20.20 activate
```

3 Inter-Op EVPN

Im Folgenden wird sowohl eine Anycast Gateway, wie auch vereinfacht eine Centralized Gateway Lösung, vorgestellt. Beide Ansätze verfolgen hierbei das Ziel, Host-zu-Host-Kommunikation über eine EVPN-Fabric bereitzustellen.

3.1 Anycast Default Gateway

Eine EVPN Anycast Gateway Lösung ist ein Ansatz, der in modernen Rechenzentrums- sowie Campusnetzwerken auf Basis von VXLAN-EVPN genutzt wird. Dabei stellen mehrere Leaf-Switches dieselbe Default-Gateway IP-Adresse und MAC-Adresse bereit, sodass jedes angeschlossene Endgerät sein Gateway immer direkt am lokalen Leaf erreicht.

Im Folgenden wird dabei die Konfiguration von FRRouting unter Debian 12 sowie auf einem EdgeCore Switch mit SONiC als Betriebssystem erläutert. Darüber hinaus wird die damit sowie untereinander kompatible BGP-EVPN Implementierung der folgenden Hersteller gelistet:

- Arista (EOS 4.33.2F)
- Aruba (ArubaOS-CX 10.13)
- Cisco Catalyst (IOS-XE 17.12)
- Cisco Nexus (NX-OS 10.5(2))
- Huawei (YunShanOS V600R023C00SPC500)

Die Topologie sowie die vergebenen IP-Adressen lassen sich dabei dem Schaubild 3.1 entnehmen.

Als Tenant-VLAN und damit ebenfalls als Anycast-Gateway wird das (Interface) VLAN 2 innerhalb des VRF `green` verwendet. Auf allen Leafs ist hierfür ein SVI konfiguriert, das die IPv4-Adresse `10.0.2.1/24` sowie die IPv6-Adresse `2001:db8:0:2::1/64` bereitstellt. Als

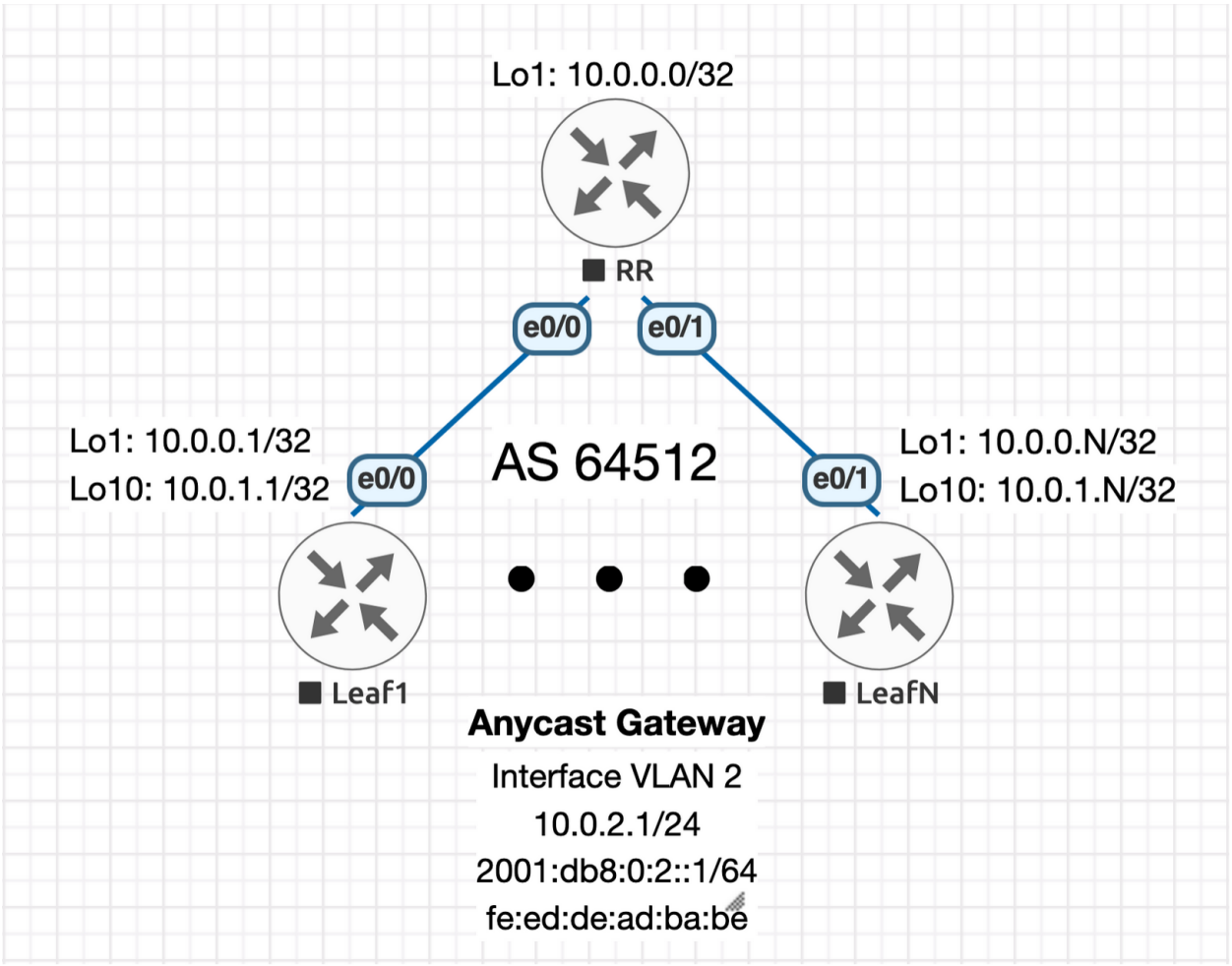

Abb. 3.1: Anycast Gateway Topologie

Gateway-MAC-Adresse dient `fe:ed:de:ad:ba:be`, die auf allen Leafs identisch verwendet wird.

Das Interface Loopback1 befindet sich im Underlay und dessen Adresse fungiert auf den Leafs als VTEP-Adresse. Gleichzeitig wird es für das BGP-Peering mit dem Route Reflektor in der L2VPN EVPN Adressfamilie eingesetzt.

Innerhalb des VRF `green` ist zusätzlich das Interface Loopback10 konfiguriert. Dieses dient als Tenant-Loopback und wird ausschließlich innerhalb des VRF verwendet.

Die BGP-Verbindungen zwischen Leafs und Spines sind als iBGP-Sessions realisiert und verwenden die AS-Nummer 64512.

3.1.1 FRRouting

3.1.1.1 Debian

Für den Betrieb muss zunächst die FRR-Daemon-Konfiguration angepasst werden. In der Datei `/etc/frr/daemons` muss Folgendes aktiviert werden:

```
#/etc/frr/daemons
bgpd=yes
ospfd=yes
...
vtysh_enable=yes
zebra_options="  -A 127.0.0.1 -s 90000000"
mgmtd_options="  -A 127.0.0.1"
bgpd_options="   -A 127.0.0.1"
ospfd_options="  -A 127.0.0.1"
```

Das Interface loopback1 wird als Dummy-Interface angelegt und mit der Adresse 10.0.0.8/32 versehen. Dieses dient als VTEP-Adresse und wird im Underlay-Routing per OSPF propagiert und als Update-Source für BGP-Sessions verwendet.

```
ip link add loopback1 type dummy
ip addr add 10.0.0.8/32 dev loopback1
ip link set up dev loopback1
```

Die EVPN Konfiguration basiert auf folgender Dokumentation. Für die Trennung der Tenant-Routing-Instanzen werden in dieser Umgebung Linux VRFs verwendet. Das VRF `green` wird mit Routing-Tabelle 1100 angelegt. Die Bridge br3 wird erstellt, dem VRF zugeordnet und mit einer festen MAC-Adresse versehen. Anschließend wird vxlan10003 mit VNI 10003 erzeugt und als Bridge-Port an br3 angebunden. VRF und Interfaces werden aktiviert, womit das L3VNI für tenant-spezifisches Routing/IRB bereitsteht.

```
ip link add green type vrf table 1100
ip link set green up

ip link add br3 type bridge
ip link set br3 master green addrgenmode none
```

```
ip link set br3 addr aa:bb:cc:00:00:64

ip link add vxlan10003 type vxlan local 10.0.0.8 dstport 4789 id 10003
   ↪ nolearning
ip link set vxlan10003 master br3 addrgenmode none
ip link set vxlan10003 type bridge_slave neigh_suppress on learning off

ip link set up dev vxlan10003
ip link set up dev br3
```

Es wird die Bridge br2 (als Anycast Gateway) sowie das VXLAN-Interface vxlan10002 mit der ID 10002 (VNI 10002), UDP-Port 4789, entsprechender IP-Adresse und deaktiviertem MAC-Learning konfiguriert. Anschließend wird das Interface vxlan10002 der Bridge br2 hinzugefügt und beide Interfaces aktiviert.

```
ip link add br2 type bridge
ip link set br2 master green
ip link set br2 addr fe:ed:de:ad:ba:be
ip addr add 10.0.2.1/24 dev br2
ip addr add 2001:db8:0:2::1/64 dev br2
ip link add vxlan10002 type vxlan local 10.0.0.8 dstport 4789 id 10002
   ↪ nolearning
ip link set vxlan10002 master br2 addrgenmode none
ip link set vxlan10002 type bridge_slave neigh_suppress on learning off
ip link set vxlan10002 up
ip link set br2 up
```

Das Dummy-Interface loopback10 wird angelegt, aktiviert und dem VRF `green` zugeordnet.

```
ip link add loopback10 type dummy
ip addr ad 10.0.1.8/32 dev loopback10
ip link set up dev loopback10
ip link set master green dev loopback10
ip addr add 2001:db8:0:1::8/128 dev loopback10
```

3.1.1.2 Konfiguration

Im Folgenden wurden die für den Betrieb notwendigen Konfigurationsbestandteile unter FRR sowie der in Kapitel 3.1 gelisteten Herstellerplattformen, soweit möglich, unterteilt in die Abschnitte VRF, Interfaces, BGP und VLANs/L2VPN. Vorausgesetzt wird einerseits, dass die VLANs 2 und VLAN 3 erstellt wurden. Darüber hinaus existiert eine direkte Verbindung jeweils zwischen den Leafs sowie dem Route Reflektor, wobei OSPF im Underlay als IGP verwendet wird.

VRF

Dieser Block definiert die Tenant-Routing-Instanz `green` verknüpft sie mit dem L3VNI 10003.

```
vrf green
 vni 10003
```

Interfaces

Der Interface-Block konfiguriert loopback1 als Underlay-VTEP-Adresse und bindet es in OSPF (Area 0) ein.

```
interface loopback1
 ip ospf area 0.0.0.0
```

BGP

Der globale BGP Block startet den EVPN Control-Plane-Prozess (AS 64512) mit loopback1 als Router-ID und Update-Source. Der VNI-Abschnitt (VNI 10002) legt RD/RT für das L2VNI fest und `advertise-svi-ip` sorgt dafür, dass Anycast SVI Adressen bekannt gemacht werden. Der VRF-spezifische BGP-Abschnitt (`router bgp 64512 vrf green`) übernimmt Redistribution von connected/static Routen für IPv4/IPv6 in das VRF und konfiguriert in der EVPN Adressfamilie den RD sowie RT für VNI 10003.

```
router bgp 64512
 bgp router-id 10.0.0.8
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.0.0.0 remote-as 64512
 neighbor 10.0.0.0 update-source loopback1
 !
 address-family l2vpn evpn
  neighbor 10.0.0.0 activate
  advertise-all-vni
  vni 10002
   rd 10.0.0.8:10002
   route-target import 64512:10002
   route-target export 64512:10002
  exit-vni
  advertise-svi-ip
  route-target import auto
  route-target export auto
 exit-address-family
exit
!
router bgp 64512 vrf green
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute static
 exit-address-family
 !
 address-family ipv6 unicast
  redistribute connected
  redistribute static
 exit-address-family
 !
 address-family l2vpn evpn
  advertise ipv4 unicast
  advertise ipv6 unicast
  rd 10.0.0.8:10003
  route-target import 64512:10003
  route-target export 64512:10003
 exit-address-family
exit
```

3.1.2 Cisco Catalyst

VRF

Definiert die Tenant-Instanz `green` und hinterlegt die RTs für Import/Export. Der RD wird hierbei automatisch erzeugt.

```
vrf definition green
 rd-auto
 address-family ipv4
  route-target export 64512:10003
  route-target import 64512:10003
  route-target export 64512:10003 stitching
  route-target import 64512:10003 stitching
 exit-address-family
 !
 address-family ipv6
  route-target export 64512:10003
  route-target import 64512:10003
  route-target export 64512:10003 stitching
  route-target import 64512:10003 stitching
 exit-address-family
```

L2VPN

Aktiviert EVPN auf dem Switch und ordnet die VLANs den VNIs zu.

```
l2vpn evpn
 replication-type static
 router-id Loopback1
!
l2vpn evpn instance 10002 vlan-based
 encapsulation vxlan
 replication-type ingress
!
vlan configuration 2
 member evpn-instance 10002 vni 10002
vlan configuration 3
 member vni 10003
```

Interfaces

```
interface Loopback1
 ip address 10.0.0.5 255.255.255.255
 ip ospf 1 area 0
!
interface Loopback10
 vrf forwarding green
 ip address 10.0.1.5 255.255.255.255
 ipv6 address 2001:DB8:0:1::5/128
 ipv6 enable
!
interface Vlan2
 mac-address feed.dead.babe
 vrf forwarding green
 ip address 10.0.2.1 255.255.255.0
 ipv6 address 2001:DB8:0:2::1/64
 ipv6 enable
!
interface Vlan3
 description L3VNI
 vrf forwarding green
 ip unnumbered Loopback1
 ipv6 enable
 no autostate
!
interface nve1
 no ip address
 source-interface Loopback1
 host-reachability protocol bgp
 member vni 10002 ingress-replication
 member vni 10003 vrf green
```

BGP

Startet BGP mit Loopback1 als Router-ID und aktiviert EVPN zum RR. Statische sowie connected Routen werden redistributed.

```
router bgp 64512
 bgp router-id 10.0.0.5
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.0.0.0 remote-as 64512
 neighbor 10.0.0.0 update-source Loopback1
 !
 address-family ipv4
 exit-address-family
 !
 address-family ipv6
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor 10.0.0.0 activate
  neighbor 10.0.0.0 send-community both
 exit-address-family
 !
 address-family ipv4 vrf green
  advertise l2vpn evpn
  redistribute static
  redistribute connected
 exit-address-family
 !
 address-family ipv6 vrf green
  redistribute connected
  redistribute static
  advertise l2vpn evpn
 exit-address-family
```

3.1.3 Cisco Nexus

VLANs/L2VPN

Ordnet VLAN 2 dem L2VNI 10002 zu. RT und RD werden automatisch konfiguriert.

```
vlan 2
  vn-segment 10002
evpn
  vni 10002 l2
    rd auto
    route-target import auto
    route-target export auto
```

VRF

Legt den Tenant `green` an und verknüpft ihn mit dem L3VNI 10003. RT und RD werden automatisch konfiguriert.

```
vrf context green
  vni 10003 l3
  rd auto
  address-family ipv4 unicast
    route-target both auto evpn
  address-family ipv6 unicast
    route-target both auto evpn
```

Interfaces

```
fabric forwarding anycast-gateway-mac feed.dead.babe

interface loopback1
  ip address 10.0.0.3/32
  ip router ospf underlay area 0.0.0.0

interface loopback10
  vrf member green
  ip address 10.0.1.3/32
  ipv6 address 2001:db8:0:1::3/128

interface Vlan2
  no shutdown
  vrf member green
  ip address 10.0.2.1/24
  ipv6 address 2001:db8:0:2::1/64
  fabric forwarding mode anycast-gateway

interface nve1
  no shutdown
  host-reachability protocol bgp
  source-interface loopback1
  member vni 10002
    ingress-replication protocol bgp
  member vni 10003 associate-vrf
```

BGP

Im VRF `green` werden direkt verbundene und statische Routen redistributed.

```
route-map permit permit 0

router bgp 64512
  router-id 10.0.0.3
  log-neighbor-changes
  address-family l2vpn evpn
  neighbor 10.0.0.0
    remote-as 64512
    log-neighbor-changes
    update-source loopback1
    address-family l2vpn evpn
      send-community
      send-community extended
  vrf green
    address-family ipv4 unicast
      redistribute direct route-map permit
      redistribute static route-map permit
    address-family ipv6 unicast
      redistribute direct route-map permit
      redistribute static route-map permit
```

3.1.4 Arista

VRF/L2VPN

Deaktiviert Flooding von ARP-/Neighbor-Advertisements.

```
router l2-vpn
   virtual-router neighbor advertisement flooding disabled
   virtual-router arp advertisement flooding disabled
```

Interfaces

```
ip virtual-router mac-address fe:ed:de:ad:ba:be
!
interface Loopback1
   ip address 10.0.0.1/32
   ip ospf area 0.0.0.0
!
interface Loopback10
   vrf green
   ip address 10.0.1.1/32
   ipv6 address 2001:db8:0:1::1/128
!
interface Vlan2
   vrf green
   ipv6 enable
   ip address virtual 10.0.2.1/24
   ipv6 address virtual 2001:db8:0:2::1/64
!
interface Vxlan1
   vxlan source-interface Loopback1
   vxlan udp-port 4789
   vxlan vlan 2 vni 10002
   vxlan vrf green vni 10003
```

BGP

Im VRF `green` werden der RD/RT statisch gesetzt und die lokalen sowie statischen Routen redistributed.

```
router bgp 64512
   neighbor 10.0.0.0 remote-as 64512
   neighbor 10.0.0.0 update-source Loopback1
   neighbor 10.0.0.0 send-community extended
   !
   vlan 2
      rd 10.0.0.1:10002
      route-target both 64512:10002
      redistribute learned
   !
   address-family evpn
      neighbor 10.0.0.0 activate
   !
   address-family ipv4
      no neighbor 10.0.0.0 activate
   !
   address-family rt-membership
      neighbor 10.0.0.0 activate
   !
   vrf green
      rd 10.0.0.1:10003
      route-target import evpn 64512:10003
      route-target export evpn 64512:10003
      redistribute connected
      redistribute static
```

3.1.5 Aruba

VRF

Konfiguriert den Tenant `green` und setzt RT/RD statisch.

```
vrf green
    rd 10.0.0.4:10003
    route-target export 64512:10003 evpn
    route-target import 64512:10003 evpn
```

L2VPN

Konfiguriert RD sowie RT statisch.

```
evpn
    vlan 2
        rd 10.0.0.4:10002
        route-target export 64512:10002
        route-target import 64512:10002
        redistribute host-route
```

Interfaces

```
interface loopback 1
    ip address 10.0.0.4/32
    ip ospf 1 area 0.0.0.0
interface loopback 10
    vrf attach green
    ip address 10.0.1.4/32
    ipv6 address 2001:db8:0:1::4/128
interface vlan 2
    vrf attach green
    ip address 10.0.2.1/24
    active-gateway ip mac fe:ed:de:ad:ba:be
    active-gateway ip 10.0.2.1
    ipv6 address 2001:db8:0:2::1/64
```

```
    active-gateway ipv6 mac fe:ed:de:ad:ba:be
    active-gateway ipv6 2001:db8:0:2::1
interface vxlan 1
    source ip 10.0.0.4
    no shutdown
    vni 10002
        vlan 2
    vni 10003
        vrf green
        routing
```

BGP

Im VRF `green` werden lokale und statische Routen redistributed.

```
router bgp 64512
    bgp router-id 10.0.0.4
    bgp log-neighbor-changes
    neighbor 10.0.0.0 remote-as 64512
    neighbor 10.0.0.0 update-source loopback 1
    address-family l2vpn evpn
        neighbor 10.0.0.0 next-hop-unchanged
        neighbor 10.0.0.0 send-community both
        neighbor 10.0.0.0 activate
    exit-address-family
!
    vrf green
        address-family ipv4 unicast
            redistribute connected
            redistribute local loopback
            redistribute static
        exit-address-family
        address-family ipv6 unicast
            redistribute connected
            redistribute local loopback
            redistribute static
        exit-address-family
```

3.1.6 Huawei

VRF

Richtet den Tenant `green` ein und setzt RD/RT statisch.

```
ip vpn-instance green
 ipv4-family
  route-distinguisher 10.0.0.6:10003
  vpn-target 64512:10003 export-extcommunity evpn
  vpn-target 64512:10003 import-extcommunity evpn
 ipv6-family
  route-distinguisher 10.0.0.6:10003
  vpn-target 64512:10003 export-extcommunity evpn
  vpn-target 64512:10003 import-extcommunity evpn
 vxlan vni 10003
```

L2VPN

Bindet VLAN 2 über den Bridge-Domain 10002 an das L2VNI 10002 und konfiguriert den RD sowie den RT statisch.

```
bridge-domain 10002
 l2 binding vlan 2
 vxlan vni 10002
 #
 evpn
  route-distinguisher auto
  vpn-target 64512:10002 export-extcommunity
  vpn-target 64512:10002 import-extcommunity
```

Interfaces

```
interface LoopBack1
 ip address 10.0.0.6 255.255.255.255
 ospf enable 1 area 0.0.0.0
#
interface LoopBack10
 ip binding vpn-instance green
 ipv6 enable
 ip address 10.0.1.6 255.255.255.255
 ipv6 address 2001:DB8:0:1::6/128
#
interface Vbdif10002
 ip binding vpn-instance green
 ipv6 enable
 ip address 10.0.2.1 255.255.255.0
 ipv6 address 2001:DB8:0:2::1/64
 mac-address feed-dead-babe
 vxlan anycast-gateway enable
 arp collect host enable
#
interface Nve1
 source 10.0.0.6
 vni 10002 head-end peer-list protocol bgp
```

BGP

In der vpn-instance `green` (Synonym für VRF) werden direkte und statische Routen redistributed.

```
bgp 64512
 router-id 10.0.0.6
 private-4-byte-as disable
 peer 10.0.0.0 as-number 64512
 peer 10.0.0.0 connect-interface LoopBack1
 #
 ipv4-family unicast
  peer 10.0.0.0 enable
 #
 ipv4-family vpn-instance green
  import-route direct
  import-route static
  advertise l2vpn evpn
 #
 ipv6-family vpn-instance green
  import-route direct
  import-route static
  advertise l2vpn evpn
 #
 l2vpn-family evpn
  policy vpn-target
  peer 10.0.0.0 enable
```

3.2 Centralized Default Gateway

Die folgende Abbildung zeigt den Aufbau bestehend aus zwei Leafs sowie einem Route Reflektor, welcher gleichzeitig als Spine agiert und die Pakete zwischen den beiden Leafs und den daran angebundenen Clients weiterleitet.

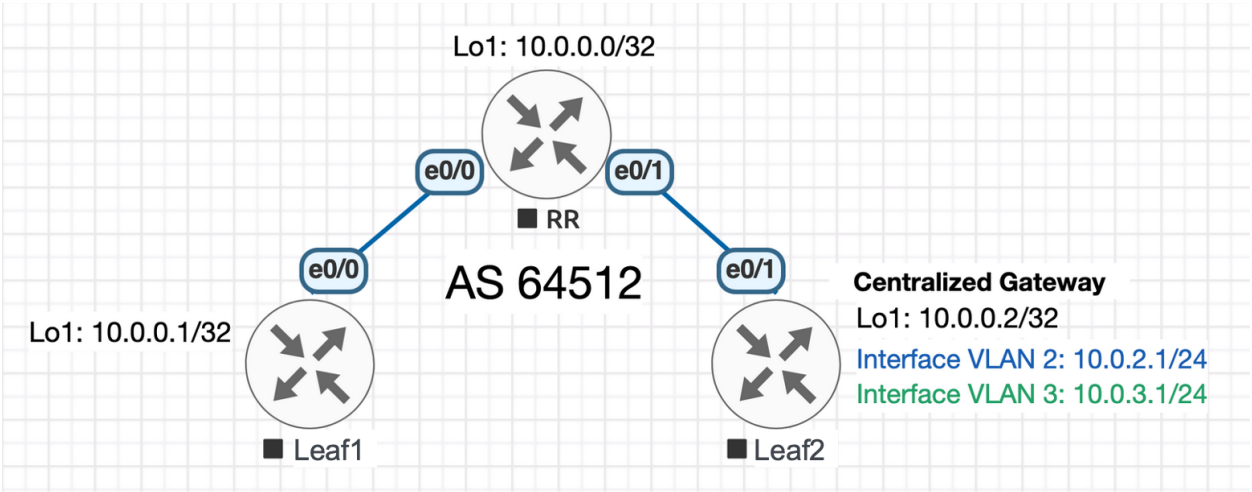

Abb. 3.2: Centralized Gateway Topologie

Bei Leaf 1 handelt es sich um einen Cisco Catalyst 9500 Switch mit der Software Version IOS-XE 17.12 und bei Leaf 2, welcher die Centralized Gateway Funktion übernimmt, um die FRRouting Instanz, welche unter Debian 12 läuft. In diesem Setup fungiert eine FRRouting-Instanz unter Debian 12 als Route-Reflektor und leitet gleichzeitig Datenpakete zwischen den beiden Leafs weiter. Auf dessen Konfiguration wird mit Verweis auf Kapitel 2.1 nicht weiter eingegangen.

Folgende Loopback Adressen werden dabei verwendet:

Gerät	Interface	Adresse
RR	Lo0	10.0.0.0/32
Leaf1	Lo0	10.0.0.1/32
Leaf2	Lo0	10.0.0.2/32

Tab. 3.1: Centralized Gateway: Loopback Interfaces

Vlan	VNI	L2VNI/L3VNI	BUM Replication Type
2	10002	L2VNI	Ingress Replication
3	10003	L2VNI	Ingress Replication

Tab. 3.2: Centralized Gateway: VLANs

3.2.1 Konfiguration

3.2.1.1 Cisco Catalyst

Die L2VPN EVPN Konfiguration des Catalyst Switches sieht dabei wie folgt aus:

```
l2vpn evpn
 logging peer state
 replication-type ingress
 route-target auto vni
!
l2vpn evpn instance 2 vlan-based
 encapsulation vxlan
l2vpn evpn instance 3 vlan-based
 encapsulation vxlan
!
vlan configuration 2
 member evpn-instance 2 vni 10002
vlan configuration 3
 member evpn-instance 3 vni 10003
!
interface nve1
 no ip address
 source-interface Loopback1
 host-reachability protocol bgp
 member vni 10002 ingress-replication
 member vni 10003 ingress-replication
end
```

Entsprechend der Tabelle in Kapitel 3.2 werden den beiden VLANs 2 und 3 die VNIs 10002 und 10003 zugeordnet. Ebenfalls wird für beide VNIs Ingress Replication als Replication Type konfiguriert.

```
router bgp 1
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 10.0.0.0 remote-as 1
 neighbor 10.0.0.0 update-source Loopback1
 !
 address-family l2vpn evpn
  neighbor 10.0.0.0 activate
  neighbor 10.0.0.0 send-community both
 exit-address-family
```

Da sich das CGW auf der FRR Instanz befindet und somit asymmetrisches Integrated Routing and Bridging stattfindet, werden keinerlei SVIs auf dem Leaf 1 konfiguriert. Eingehende Pakete der Clients 1 und 2 werden, insofern deren Ziel sich ausserhalb des eigenen Subnetzes befindet, immer zum CGW gebridged.

3.2.1.2 FRRouting

Network Namespaces

Zur Trennung von Overlay und Underlay Netzwerk werden exemplarisch, entgegen der Verwendung von Linux VRFs im Anycast Gateway Szenario, Linux Network Namespaces verwendet. Im Folgenden wird ein Namespace mit dem Namen `underlay` für das Underlay sowie ein Overlay bzw. Tenant Namespace mit dem Namen `campus`, verwendet.

```
ip netns add underlay
ip netns add campus
ip netns exec underlay ip link set lo up
ip netns exec campus ip link set lo up
```

Ebenfalls muss obiges Interface e0/1 in das `underlay` Namespace gesetzt und anschließend eine Adresse konfiguriert werden:

```
ip link set e0/1 netns underlay
ip netns exec underlay ip addr add <ip-addr> dev e0/1
ip netns exec underlay ip link set e0/1 up
```

Network Namespaces müssen in `/etc/frr/daemons` mittels der Zebra Option `-n` aktiviert werden. Ebenfalls kann watchfrr option `--netns=underlay` gesetzt werden, damit FRR im `underlay` Namespace startet.

```
# /etc/frr/daemons

bgpd=yes
ospfd=yes
...
vtysh_enable=yes
zebra_options="  -A 127.0.0.1 -s 90000000 -n"
mgmtd_options="  -A 127.0.0.1"
bgpd_options="   -A 127.0.0.1"
ospfd_options="  -A 127.0.0.1"
...
watchfrr_options="--netns=underlay"
```

Innerhalb der Konfigurationsdatei von FRR wird das `campus` Namespace eingebunden mittels:

```
vrf campus
 netns /run/netns/campus
exit-vrf
```

Interfaces

Im Namespace `underlay` werden zunächst die beiden L2VNIs konfiguriert und anschließend in den `campus` Namespace gesetzt. Als Namen für die Interfaces werden vni10002 und vni10003 jeweils mit den IDs 10002 und 10003 verwendet. Als Destination Port wird 4789 verwendet. Die "local" bzw. Source IP-Adresse des VXLAN Interfaces ist die des Loopback Interfaces im `underlay`. Als Adresse wird die des Loopback Interfaces verwendet.

```
ip netns exec underlay ip link add vni10002 type vxlan id 10002 dstport
   ↪ 4789 local 10.0.0.2 nolearning
ip netns exec underlay ip link add vni10003 type vxlan id 10003 dstport
   ↪ 4789 local 10.0.0.2 nolearning
ip netns exec underlay ip link set vni10002 netns campus
ip netns exec underlay ip link set vni10003 netns campus
```

Anschließend werden die Bridge Interfaces br10002 und br10003 erstellt und die VXLAN Interfaces den Bridge Interfaces zugeordnet. Den Bridges werden darüber hinaus die IP-Adressen der Default-Gateways zugewiesen.

```
ip netns exec campus ip link add name br10002 type bridge
ip netns exec campus ip link set vni10002 master br10002
ip netns exec campus ip addr add 10.0.2.1/24 dev br10002
ip netns exec campus ip link set br10002 up
ip netns exec campus ip link set vni10002 up

ip netns exec campus ip link add name br10003 type bridge
ip netns exec campus ip link set vni10003 master br10003
ip netns exec campus ip addr add 10.0.3.1/24 dev br10003
ip netns exec campus ip link set br10003 up
ip netns exec campus ip link set vni10003 up
```

Weitere (Client-)Interfaces im Namespace `campus` lassen sich den Bridge Interfaces wie folgt zuordnen:

```
ip netns exec campus ip link set eth0 master br10002
ip netns exec campus ip link set eth1 master br10003
```

Die Interface- sowie OSPF-Konfiguration sieht wie folgt aus:

```
interface e0/1
 ip address <ip-addr>
 ip ospf area 0
 ip ospf network point-to-point
exit
!
interface lo
 ip address <ip-addr>
 ip ospf area 0
exit
!
router ospf
 ospf router-id 10.0.0.2
exit
```

Um EVPN für eine BGP Instanz zu aktivieren, muss `advertise-all-vni` unter der EVPN Konfiguration vorgenommen werden. Genauso wird mittels `advertise-svi-ip` die Adresse des Default Gateways geadvertised.

```
router bgp 1
 bgp router-id 10.0.0.2
 neighbor 10.0.0.0 remote-as 1
 neighbor 10.0.0.0 update-source 10.0.0.0
 !
 address-family l2vpn evpn
  neighbor 10.0.0.0 activate
  advertise-all-vni
  advertise-svi-ip
 exit-address-family
exit
```

3.2.2 Verifikation

Die NVE Peers bzw. VNIs lassen sich auf den Leafs wie folgt anzeigen:

Cisco Catalyst

```
Leaf1#show nve peers
...
Interface  VNI      Type Peer-IP  ...   state
nve1       10002    L2CP 10.0.0.0 ...     UP
nve1       10003    L2CP 10.0.0.0 ...     UP
```

FRRouting

```
Leaf2# show evpn vni
VNI        Type VxLAN IF  ... Tenant VRF
10002       L2  vni10002  ...   campus
10003       L2  vni10003  ...   campus
```

4 BGP Software Bibliotheken

Eine leichtgewichtige Alternative zu FRR stellen BaGPipe und yabgp dar, wobei es sich hierbei um Open-Source BGP Softwarebibliotheken handelt, die Integration und Verwaltung von BGP-Routing ohne eine vollständige Router-Software. Bei yabgp handelt es sich um eine Python-basierte BGP-Bibliothek, die helfen soll, BGP-Sessions zu konfigurieren und Routen zu verwalten. Beide Bibliotheken bieten eine leichtgewichtige Lösung für die Arbeit mit BGP, jedoch ohne die vollständige Funktionalität eines traditionellen Routers der BGP spricht.

Darüber hinaus existiert mit BIRD eine weitere Open-Source Routing-Software, welche im Gegensatz zu BaGPipe und yabgp eine eigenständige Routing-Daemon Implementierung darstellt und neben BGP auch andere Routingprotokolle unterstützt. Allerdings befindet sich die Unterstützung für erweiterte Funktionen wie EVPN derzeit noch im Aufbau, sodass BIRD bislang nur eingeschränkt für komplexe EVPN-Szenarien einsetzbar ist. Damit bietet BIRD zwar eine leistungsfähige, aber im Bereich moderner Overlay-Technologien noch nicht vollständig ausgereifte Alternative zu umfassenden Routing-Suiten wie FRR.

5 Fazit

Das vorliegende Konzeptpapier soll einen Überblick über den aktuellen Stand von Open-Source und Closed-Source Lösungen für BGP EVPN geben. Damit erleichtert es Einsteigern den Zugang zum Thema und dient als Ausgangsbasis für weitere Evaluierungen und Proof-of-Concepts in unterschiedlichen Betriebsumgebungen.

6 Versionsverlauf

Version	Datum	Änderungen
1.0	17.10.2025	Initiale Veröffentlichung