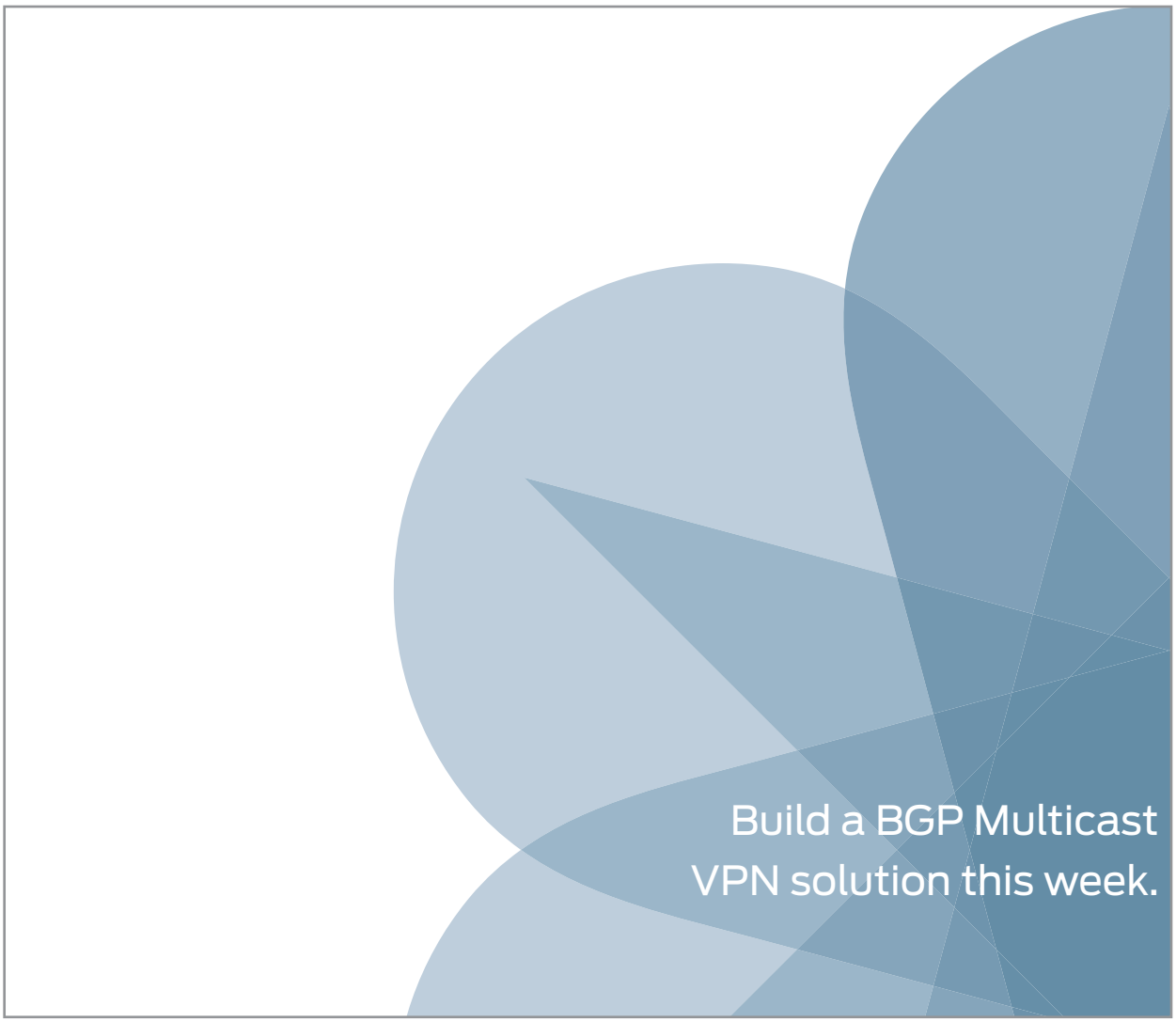
Junos® Networking Technologies

THIS WEEK: DEPLOYING BGP MULTICAST VPNs, 2ND EDITION



By Antonio Sánchez Monge

THIS WEEK: DEPLOYING BGP MULTICAST VPNs, 2ND EDITION

The networking industry has been looking for the best way to offer Multicast VPN services while leveraging the strength and scalability of the existing BGP/MPLS VPN technology. The result of several years of effort is Multiprotocol BGP Multicast VPN, often referred to as BGP MVPN. This next generation technology has received a warm welcome in the market and is already deployed in many production networks, from Tier-1 service providers to financial and trading companies.

This Week: Deploying BGP Multicast VPNs assumes the reader has at least some experience with IP/MPLS architectures, including Multiprotocol BGP and IGPs. You are not expected to be an expert in multicast as the basic concepts are revisited in the book, but if you are already familiar with BGP/MPLS IP VPNs, you will find this book easier to read.

Whatever you bring to this seminar in a book will only be amplified by its clear explanations, explicit examples, and attention to detail. The author walks you step-by-step through an advanced technology, thoroughly exploring a design that can be stood up in a week. Roll up your sleeves and let's get down to work.

"This excellent book is an ideal way to bring you up to speed on BGP Multicast VPN technology. The book is a very practical guide, clearly describing the theory while showing the reader exactly how to configure the network using Junos. It's highly recommended!"

Julian Lucek, Distinguished Systems Engineer, Juniper Networks

LEARN SOMETHING NEW ABOUT JUNOS THIS WEEK:

- Build a BGP Multicast VPN working solution from scratch.
- Configure and operate dynamic Point-to-Multipoint MPLS Label Switched Paths, with and without Traffic Engineering features.
- Describe the integration of Customer PIM instances with VPNs, both in Any-Source Multicast (ASM) and Source-Specific Multicast (SSM) scenarios.
- Design an optimal distribution of Inclusive and Selective tunnels. Find the right balance between control and forwarding plane efficiency.
- Use Route Target policies to achieve partial mesh topologies of MVPN sites.
- Understand the clean decoupling of control and forwarding planes in BGP MVPN.

Published by Juniper Networks Books
www.juniper.net/books

ISBN 9367792220

9 789367 792223 52000

07500209

Junos® Networking Technologies

This Week: Deploying BGP Multicast VPNs, 2ND Edition

By Antonio Sánchez Monge

JUNIPER® NETWORKS

Published by Juniper Networks Books
Writer: Antonio Sánchez Monge
Technical Review: Julian Lucek, Yakov Rekhter, Miguel Barreiros, Efrain Gonzalez
Editor in Chief: Patrick Ames
Copyeditor and Proofreader: Nancy Koerbel
J-Net Community Management: Julie Wider

This book is available in a variety of formats at: www.juniper.net/dayone.

Send your suggestions, comments, and critiques by email to: dayone@juniper.net.

Version History: First Edition, May 2011
Version History: Second Edition, May 2012

ISBN: 978-1-936779-22-2 (print)
Printed in the USA by Vervante Corporation.

ISBN: 978-1-936779-23-9 (ebook)

Juniper Networks Books are singularly focused on network productivity and efficiency. Peruse the complete library at www.juniper.net/books.

3 4 5 6 7 8 9 10 #7500209-en

About the Author
Antonio "Ato" Sanchez Monge (JNCIE-M #222 and CCIE #13098) holds a MS in Physics and a BA in Mathematics from the Universidad Autonoma de Madrid (UAM). He joined HP back in 1999 to specialize in networking support, then moved to Juniper Networks in 2004, where he is currently working at the Professional Services team. During the last decade, Ato has been involved with projects for different ISPs in Europe.

Author's Acknowledgments
I would like to thank: Patrick Ames, *my* great editor, for his continuous encouragement and positive attitude throughout the nine-month cycle that this book took to be born; Julian Lucek, whose technical review uncovered key implementation details that I was not aware of; Yakov Rekhter for spotting some mistakes and motivating the second version of this book; Miguel Barreiros for sharing his insight on how to structure a technical book targeted to a diverse audience; Efrain Gonzalez, whose thorough review definitely helped to make this text clearer; Anton Bernal for introducing me to the *Day One* program and for his restless knowledge-sharing passion; all the readers of the first edition who provided feedback, from Lorenzo Murillo who patiently built the lab scenarios in Junosphere and in real routers, to my father who did his best to understand the first pages; and, all my colleagues at Juniper, as well as our customers, for bringing something new to learn every day. Last but not least, I would have never written this book without the support of my family and friends, especially Eva, Manuel, and Lucas who supported my long writing hours "deep in the cave."

Welcome to *This Week*

This Week books are an outgrowth of the extremely popular *Day One* book series published by Juniper Networks Books. *Day One* books focus on providing just the right amount of information that you can execute, or absorb, in a day. *This Week* books, on the other hand, explore networking technologies and practices that in a classroom setting might take several days to absorb or complete. Both libraries are available to readers in multiple formats:

- Download a free PDF edition at http://www.juniper.net/dayone.
- Get the ebook edition for iPhones and iPads at the iTunes Store>Books. Search for *Juniper Networks Books*.
- Get the ebook edition for any device that runs the Kindle app (Android, Kindle, iPad, PC, or Mac) by opening your device's Kindle app and going to the Kindle Store. Search for *Juniper Networks Books*.
- Purchase the paper edition at either Vervante Corporation (www.vervante.com) or Amazon (www.amazon.com) for prices between $12-$28 U.S., depending on page length.
- Note that Nook, iPad, and various Android apps can also view PDF files.
- If your device or ebook app uses .epub files, but isn't an Apple product, open iTunes and download the .epub file from the iTunes Store. You can now drag and drop the file out of iTunes onto your desktop and sync with your .epub device.

What You Need to Know Before Reading

- You will need a basic understanding of Junos and the Junos CLI, including configuration changes using edit mode. See the *Day One* books at www.juniper.net/books for a variety of books at all skill levels.
- This book assumes that you have the ability to configure basic IP connectivity, including interface addressing and static routes, and to troubleshoot a simple network as needed.
- You should have at least some experience with IP/MPLS architectures, including Multiprotocol BGP and IGPs.
- If you are already familiar with BGP/MPLS VPN technology, you will find this book easier to read.
- You are not expected to be an expert in Multicast. The basic concepts are revisited in the book.
- The practical sections include hands-on tasks. You are strongly encouraged to find a stable lab to practice.

After Reading This Book You'll Be Able To

- Build a BGP Multicast VPN working solution from scratch. Comprehensively choose among the available design options for Customer multicast routing and Provider transport.
- Configure and operate dynamic Point-to-Multipoint MPLS Label Switched Paths, with and without Traffic Engineering features. Understand the details of Provider Tunnel Auto-Discovery.
- Describe the integration of Customer PIM instances with VPNs. Explain the signaling involved in Any-Source Multicast (ASM) and Source-Specific Multicast (SSM) scenarios.
- Design an optimal distribution of Inclusive and Selective tunnels. Find the right balance between control and forwarding plane efficiency.
- Use Route Target policies to achieve partial mesh topologies between Multicast VPN sites.
- Find an optimal placement of Customer Rendez-Vous Points in the ASM model. Explain the differences between SPT-only and RPT-SPT modes.
- List the different BGP Route Types supported for Multicast VPN. Draw flow charts with the signaling steps.
- Operate and troubleshoot the BGP MVPN solution effectively.
- Explain in detail the differences between draft-rosen and BGP Multicast VPN. Understand the clean decoupling of control and forwarding planes in the BGP-based solution.

MORE? An excellent source of information for how to deploy MPLS is *This Week: Deploying MPLS*, available at www.juniper.net/dayone.

MORE? An excellent source of advanced MPLS topics can be found in *MPLS-Enabled Applications, Third Edition*, by Ina Minei and Julian Lucek (2011, Wiley & Sons Publishers). You can get more information on this best-selling MPLS book at www.juniper.net/books.

MORE? Another excellent source for further reading is *Deploying Next Generation Multicast-enabled Applications: Label Switched Multicast for MPLS VPNs, VPLS, and Wholesale Ethernet*, by Vinod Joseph and Srinivas Mulugu (2011, Morgan Kaufmann). For more information see www.juniper.net/books.

Chapter 1

Introducing BGP Multicast VPNs

Following upon the huge success of Unicast BGP/MPLS VPN solutions, the networking industry has been looking for the best way to offer Multicast VPN services while leveraging the strength and scalability of the existing unicast technology. The result of several years effort is Multiprotocol BGP Multicast VPN, often referred to as BGP MVPN. This technology has received a warm welcome in the market and is already deployed in many production networks, ranging from Tier-1 service providers to financial and trading companies. The first chapter of this book introduces this state-of-the-art solution and explains how it differs from the other Multicast VPN flavors.

NOTE This book does not assume you to be an expert in both IP Multicast *and* BGP/MPLS, so the chapter begins with a basic concept refresher. Even if you are already familiar with both topics, you should read the introductory sections to understand the differences between draft-rosen and BGP Multicast VPN.

After the basic theoretical introduction, a comprehensive description of the different Multicast VPN solutions is provided. The flexibility of the Junos BGP Multicast VPN implementation makes it difficult to thoroughly cover all the configuration alternatives in any single book, so among the existing transport options, RSVP P2MP LSPs are used to illustrate the technology throughout the practical sections.

One last item. This book focuses on IPv4 BGP Multicast VPN, but the concepts and technology are fully portable to IPv6 BGP Multicast VPN, also supported by the Junos operating system.

IP Multicast Refresher

Multicast traffic flows from a given source (S) to a group (G) of receivers, as compared to unicast traffic, which is destined to a single receiver. The forwarding path used to transport multicast is typically modeled as a tree, with the source being the root and the receivers sitting at the leaves. Traffic is replicated by the routers at the branching points of the tree, so the structure is often referred to as a *multicast distribution tree*, or more simply, *multicast tree*. The tree is represented upside down, with the sources on top and the receivers at the bottom, so the traffic flows down, much like water in a river. With this picture in mind, the terms *upstream* and *downstream* are equivalent to towards the source and towards the receivers, respectively.

Multicast routing protocols are required to transport multicast traffic in networks where not all receivers sit in the same network segment as the source. When this is the case, the router directly connected to the source is called the *first-hop router*, while the *last-hop routers* are directly connected to the receivers. If the source and a receiver are both directly connected to the same router in different network segments, the first-hop router is also a last-hop router.

Multicast packets have an unicast source address and a multicast destination address. With IPv4 addressing in mind, multicast IP addresses are encompassed by the 224/4 prefix, namely addresses from 224.0.0.1 up to 239.255.255.255. An IP Multicast group G is associated to a unique IP Multicast address.

Not all these addresses are routable, particularly 224.0.0.1 up to 224.0.0.255, which are link-local addresses typically used by several protocols for local signaling. For example, OSPF hello packets have destination IP addresses 224.0.0.5 or 224.0.0.6.

Internet Group Management Protocol (IGMP)

IGMP is the protocol used by the final receivers to report multicast group membership to their directly-attached gateways, which become last-hop routers from the perspective of the multicast tree. An IGMP group membership message expresses the desire to receive traffic destined to a multicast group. A single receiver can subscribe to one, or several, multicast groups, and has the option to specify the sources that it is expecting the traffic to arrive from.

The routers send periodic IGMP Queries to the network segment, and the receiving hosts answer with IGMP Reports describing their group membership. In case there are several IGMP routers in a segment, one of them is elected as the Querier (by default, the lowest IP address wins). A receiver can spontaneously send a Query or Leave packet when it first subscribes to, or unsubscribes from, a multicast group. There are three IGMP versions:

- IGMPv1: This is a legacy version, most applications do not use it anymore.
- IGMPv2: The most commonly used version, IGMPv2 supports group-specific Queries and receiver-initiated Leaves.
- IGMPv3: This version also supports source-specific Queries, Reports, and Leaves.

As an alternative to learning group membership dynamically from receiving hosts, Junos allows for the configuration of static IGMP reports in downstream interfaces, if required or if needed.

Any Source Multicast (ASM) and Source Specific Multicast (SSM)

When an IGMPv1 or IGMPv2 receiver host joins a multicast group, it sends a (*, G) Report. The G stands for a multicast group, or IP Multicast address, while the asterisk (*) is the wildcard sign, and refers to any source sending traffic to group G. This model is known as Any Source Multicast (ASM). ASM is quite simple for the receiver applications – they just need to subscribe to a group address. The complexity is in the IP network, as here the last-hop routers need to find a mechanism to join the multicast distribution tree without really knowing in advance what sources are active for a given group. In other words, the network is responsible for converting (*, G) reports into (S, G) states.

A frequent question for those new to multicast is how the IP network handles the case where two different sources (S1 and S2) send traffic to the same group. The answer is the network treats the (S1, G) flow independently from the (S2, G) flow. Both arrive to the receiver, and it is up to the receiving application to consider each flow as independent (for example, two different video streams) or redundant copies of the same channel. This latter case is the most common when a (*, G) state is created by the receiver.

In the Source Specific Multicast (SSM) model, the receiver application has a process to know in advance the source IP addresses, for example, via a web portal. The receiver then sends an IGMPv3 source-specific (S, G) report, and the last-hop router knows what sources to pull traffic from – it's a model that's simpler for the network but more complex for the end-user application.

MORE? Refer to RFC 2236 and RFC 3376 for more information on IGMPv2 and IGMPv3, respectively. Reading RFC 3569 is also recommended as it describes the SSM model from a general standpoint. All can be viewed at https://datatracker.ietf.org/.

Protocol Independent Multicast (PIM)

PIMv2 is the industry de facto IPv4 Multicast routing protocol. PIM is responsible for building multicast distribution trees connecting sources to receivers, and needs to be enabled on all the router interfaces involved in multicast routing.

TIP If it helps, you can think of IGMP as the protocol between routers and hosts, and PIM as the protocol between routers.

All the PIM packets (except the Registers) are sent with the destination IP address 224.0.0.13, hence they are processed by all the PIM-enabled neighbors. PIM adjacencies are established with the exchange of hello packets at the interface. Once two routers are PIM neighbors on a link, they can exchange other PIM control packets. The most important PIM control packet type is the Join/Prune, which contains a Join list and a Prune list. These are lists of multicast source and group addresses. PIM Join/Prune packets with an empty Prune list are commonly referred to as *Joins*, whereas those with an empty Join list are called *Prunes*.

When a router has downstream (S, G) receivers and is not connected to the (S, G) distribution tree, it typically, if running in sparse mode, sends a (S, G) PIM Join in the direction to the source S. The upstream PIM neighbor first updates its outgoing interface list for (S, G) traffic, including the link at which the Join was received. Then if it is not part of the (S, G) distribution tree yet, sends the (S, G) PIM Join to the next upstream router. This process is repeated until the PIM Join reaches the distribution tree and/or the first-hop router. If the downstream (S, G) state is lost, say, due to loss of IGMP membership at the last-hop routers, a (S, G) PIM Prune is sent upstream to cut the branch off the multicast distribution tree.

An important aspect of the PIM protocol is its soft-state nature, meaning that the multicast forwarding state is maintained by periodically sending the relevant PIM messages. Downstream routers keep a set of refresh timers controlling the periodic generation of PIM Join and Prune messages upstream. This refresh process keeps the multicast distribution tree active in steady state. In this sense, PIM behaves unlike the highly-scalable unicast routing protocols such as OSPF, IS-IS, or BGP, which have a reliable information exchange mechanism and do not require periodic control traffic flooding.

Multicast forwarding is more complex than unicast because it follows a tree structure with branching points and in order to avoid loops, it uses the RPF (Reverse Path Forwarding) mechanism. Before forwarding an IP Multicast packet, the router checks whether the packet was received at the closest interface to the source (or, to the Rendezvous Point in certain cases). The process can be seen in Figure 1.1, where router C discards multicast traffic sourced from S if it is received via router B. An IP unicast lookup is performed on the source address S, and if the next-hop points to the incoming interface, the packet is allowed for multicast forwarding. This is the case for multicast packets sourced from S and arriving to router C via the direct link to router A. The RPF check mechanism relies on IP unicast routing protocols – like static, RIP, OSPF, ISIS, or BGP – to populate the unicast routing table with routes towards all the potential multicast sources. When PIM is used, the choice of unicast routing protocols is irrelevant, hence the independent in the Protocol Independent Multicast name.

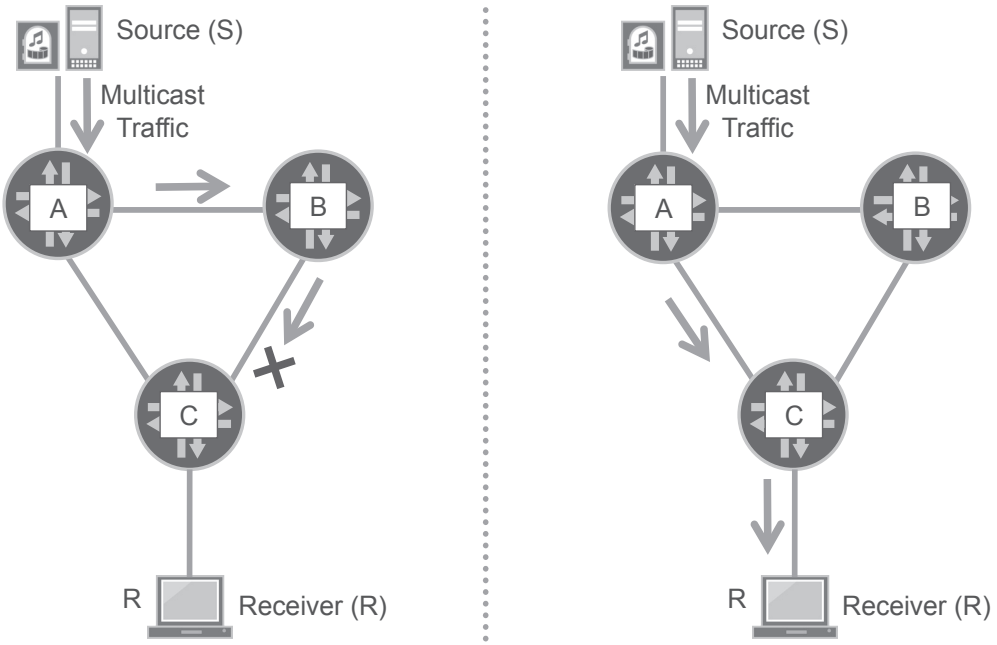


Figure 1.1 Reverse Path Forwarding (RPF) – Assuming All Links Have the Same Metric

Dense Mode and Sparse Mode

There are two common flavors of PIM covered in this book, dense mode and sparse mode PIM. (BiDir is not covered in this book.)

- Dense Mode (DM): Multicast traffic passing the RPF-check is flooded to all the PIM neighbors. All the links are potentially considered downstream interfaces. Once the traffic is flooded throughout the network, the downstream routers send PIM Prunes upstream if they have no interested receivers in order to stop the unnecessary flooding. This model is commonly described as *flood-and-prune*. Due to its bandwidth and signaling inefficiencies, it is rarely used in medium- to large-sized networks, although it is fully supported in Junos, including the BGP Multicast VPN feature set. It is not covered in this book.
- Sparse Mode (SM): This book focuses on sparse mode because it is a vastly more efficient mode in which the distribution tree is built to deliver multicast traffic only to the interested receivers. The goal is to create the necessary branches for the data distribution, without flooding the traffic all over the place. With PIM SM multicast (S, G) traffic is only forwarded down to interfaces with local receivers – having notified G membership via IGMP – or with downstream PIM Join state. The latter is generated if a downstream neighboring PIM router has sent a (S, G) or (*, G) PIM Join up to the local router.

PIM SM is supported in both the SSM and the ASM multicast models. The signaling in the SSM case is quite simple: first, the receiver sends a (S, G) IGMPv3 membership report to the last-hop router. This in turn triggers a PIM (S, G) Join upstream towards the Multicast source. The Join is processed hop-by-hop until it reaches the

first-hop router, and the process is repeated for all last-hop routers (each connected to a different set of receivers), finally building all the branches of the multicast tree rooted at the first-hop router, as shown in Figure 1.2.

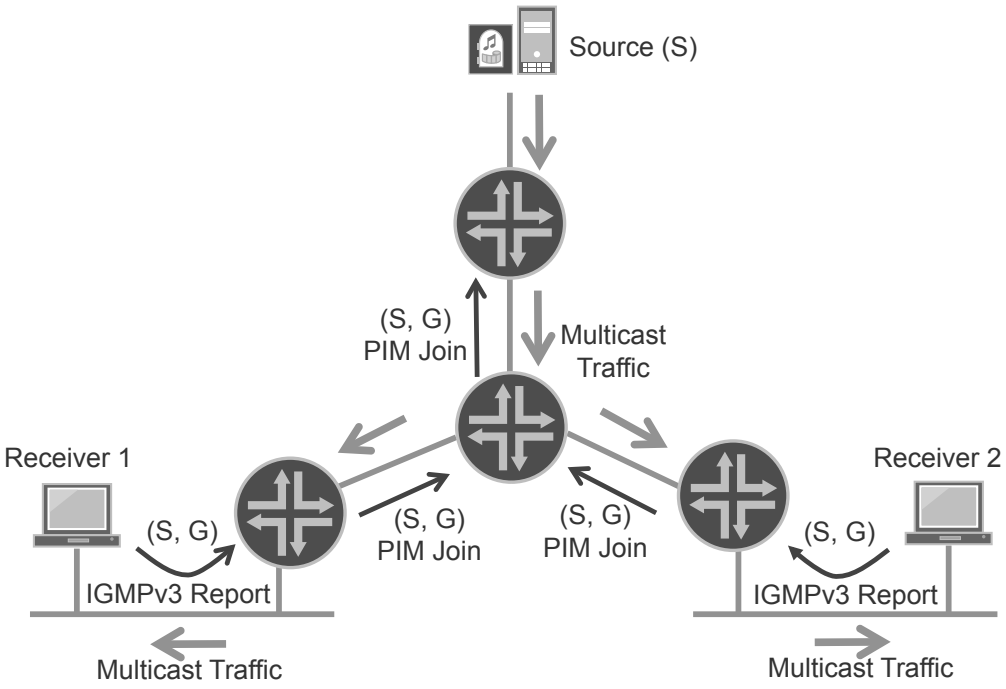


Figure 1.2 Multicast Tree Signaling with PIM SM in the SSM Model

The ASM model is a little more complex. The last-hop router has a (*, G) state learned from the receivers, typically via IGMPv2. It knows what multicast groups the receivers are interested in, but it has no knowledge about what sources are sending traffic to the group G destination address. In the same manner, the first-hop router does not know where to forward the multicast traffic generated by the local S source. A special router called Rendezvous Point (RP) is in charge of connecting sources and receivers together – rendezvous, of course, is French for *meeting*. First-hop routers register multicast sources with the RP, which learns about all the active (S, G) flows of the network and then connects downstream (*, G) branches with the sources. Figure 1.3 shows a typical ASM scenario.

In order for the ASM model to work, it is necessary that the RP can connect the sources and the receivers together. And this raises a redundancy concern: if all the routers in the network point to the same RP address, what happens if the RP fails?

Well, there are two general methods to solve this redundancy problem: either by relying on an agent that keeps track of the RP liveliness and informs the routers of the current active RP, or, by keeping several active RPs sharing a secondary virtual RP-address and exchanging information about multicast sources through a dedicated control inter-RP session. The second approach, generally known as *Anycast*, is the preferred option in best practice deployments because it provides better convergence times while being more scalable and robust.

MORE? There are two flavors of Anycast, one based in PIM Registers (RFC 4610) and another one in Multicast Source Discovery Protocol or MSDP (covered in RFC 4611). Have a look at the RFCs at https://datatracker.ietf.org/, and understand the magic of Anycast.

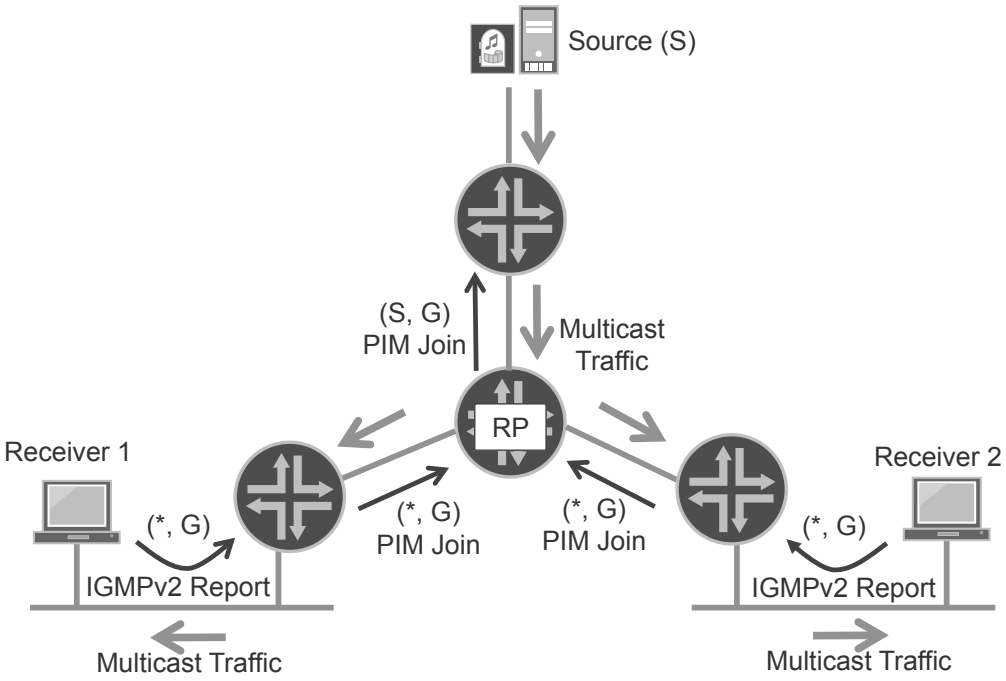


Figure 1.3 Multicast Tree Signaling with PIM SM in the ASM Model

Rendezvous Point Tree (RPT) and Shortest Path Tree (SPT)

In Figure 1.3 you saw that the shortest path between the source and the receivers went through the RP. This is not always the case. In fact in most topologies, transiting the RP is not the most efficient way to transport multicast traffic.

In the ASM model, the data packets start flowing through the Rendezvous Point Tree (RPT), also known as *Shared Tree*. Depending on its configuration, the last-hop router may decide to initiate a switchover to the Shortest Path Tree (SPT), also known as *Source Tree*, once the transit (S, G) multicast traffic allows it to learn the source address (S). Figure 1.4 illustrates the switchover process.

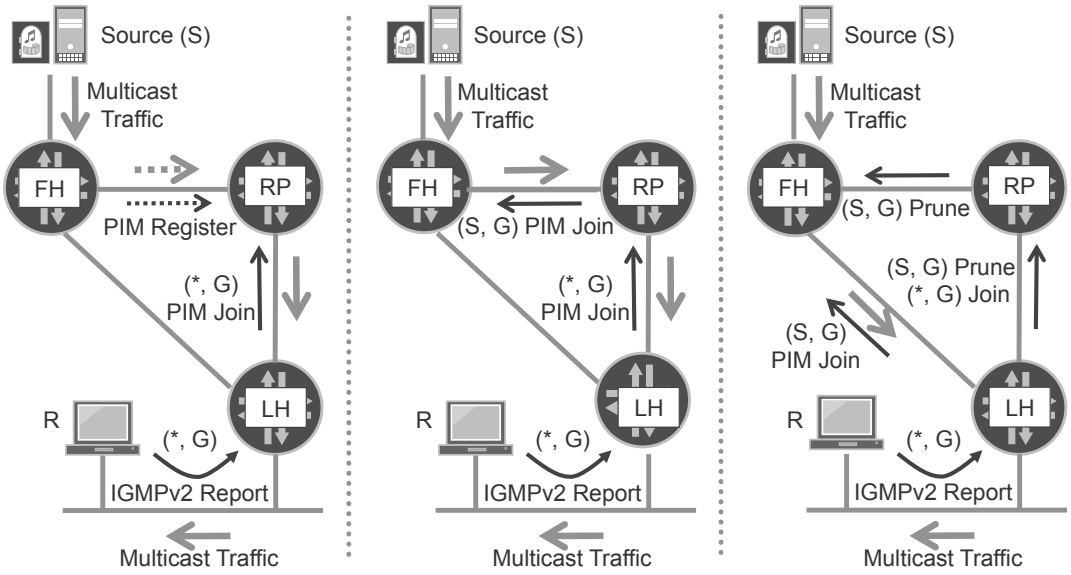


Figure 1.4 Switchover from Shared Tree to Shortest Path Tree

The full sequence of events in Figure 1.4 is shown in three panels, and proceeds from left to right as:

1. The receiver sends an IGMP (*, G) Report.

2. The last-hop router (LH) sends a PIM (*, G) Join towards the RP.

3. The source S starts to send multicast traffic to group G.

4. The first-hop router (FH) encapsulates the multicast traffic into unicast packets called PIM Register-Start, or simply *Registers*. The Registers are sent unicast to the RP.

5. The RP decapsulates the Registers and sends the native multicast traffic down the Shared Tree to LH.

6. The RP sends a PIM (S, G) Join towards FH.

7. FH forwards the multicast traffic both natively and encapsulated in Registers to the RP.

8. The RP sends a PIM Register-Stop to FH, so as to stop the Registers flow.

9. LH switches over to the Shortest Path Tree by first sending a PIM (S, G) Join towards FH.

10. FH sends the multicast traffic natively to the RP and LH.

11. LH sends a PIM (S, G) Prune to the RP.

12. The RP sends a PIM (S, G) Prune to FH.

13. FH sends the multicast traffic natively to LH only.

14. FH periodically sends a Register-Start packet to RP, which replies with a Register-Stop.

The key step in this process is Step 9, the convergence to the Shortest Path Tree (SPT) initiated by the last-hop router. By default , the last-hop router triggers the convergence to SPT in Junos as soon as the first packet from a (S, G) flow is received. Optionally, the `spt-threshold infinity` keywords can be configured at the last-hop router to prevent SPT switchover from being initiated.

Even after SPT switchover is complete, the PIM (*, G) Join towards the RP is maintained in order to receive traffic from other sources that may start sending traffic to the same G address.

Try It Yourself: Test Your PIM Expertise

It's time for you to test your PIM workings. Using a scenario similar to Figure 1.4, plot out the PIM signaling and the final forwarding path followed by multicast traffic, BUT where the first-hop router (FH) and the RP are NOT directly connected to each other. Go ahead and assume that the last-hop router (LH) IS configured with spt-threshold infinity. (Answer at the end of this chapter.)

PIM in a LAN

PIM usage in broadcast media like Ethernet requires an additional set of relatively complex mechanisms to ensure that there is only one copy of the final multicast stream being forwarded in the segment. The various mechanisms required include:

- Designated Router (DR) Election: In multi-access networks with at least two PIM-enabled routers, one of them is elected as a DR (based on the higher configured priority and numerical IP address value as a tie-breaker). The DR has two (and only two) functions. As a last-hop router, the DR takes care of processing downstream IGMP reports and brings the multicast data streams to the locally connected receivers. And as a first-hop router, the DR of the segment handles the multicast data packets originated by locally connected sources, encapsulating them in PIM Register-Start packets and sending them via unicast to the RP. The DR has no special role in intermediate core links with no end sources or receivers.
- Unicast Upstream Neighbor: PIM Join/Prune packets have the destination IP address 224.0.0.13, so they are processed by all PIM routers in a LAN. Downstream routers need to specify which upstream router a given PIM Join/Prune packet is targeted to, and they do so by setting a specific field called *Unicast Upstream Neighbor* within the PIM header. All the neighbors keep processing the messages, but only the selected router converts them into local Join/Prune state.
- Assert Mechanism: There are several situations where the DR mechanism is not enough to avoid the existence of duplicate multicast data flows. One example is depicted in Figure 1.5, with two upstream routers (U1 and U2) and two downstream routers (D1 and D2). As D1 and D2 choose a different upstream neighbor, both U1 and U2 initially inject the multicast flows into the LAN, which results in undesired traffic duplication. In order to fix this situation, U1 and U2 initiate a competition with PIM Assert packets, whose winner (based on a lower IGP metric to the source, and a higher numerical IP address value as a tie-breaker) keeps forwarding the multicast traffic. D1 and D2 listen to the PIM Assert packets and send further PIM Join/Prune messages to the Assert winner only.

- Join Suppression: In order to reduce state, D1 and D2 in Figure 1.5 have a random timer mechanism to ensure that no duplicate PIM Join/Prune refresh is sent upstream. This results, over time, into a load sharing mechanism, where D1 and D2 send each ~50% of the refresh packets to the upstream Assert winner.
- Prune Delay: If Receiver 2 in Figure 1.5 disconnects or leaves group G, D2 sends a PIM Prune to the upstream Assert winner. Before stopping the multicast forwarding state towards the LAN, the upstream router starts a timer to allow for other downstream routers like D1 to send a PIM Join overriding the previous Prune. This is possible since PIM Join/Prune packets are sent to 224.0.0.13 multicast address, hence D1 can process and react to the PIM Prune packet sent by D2.

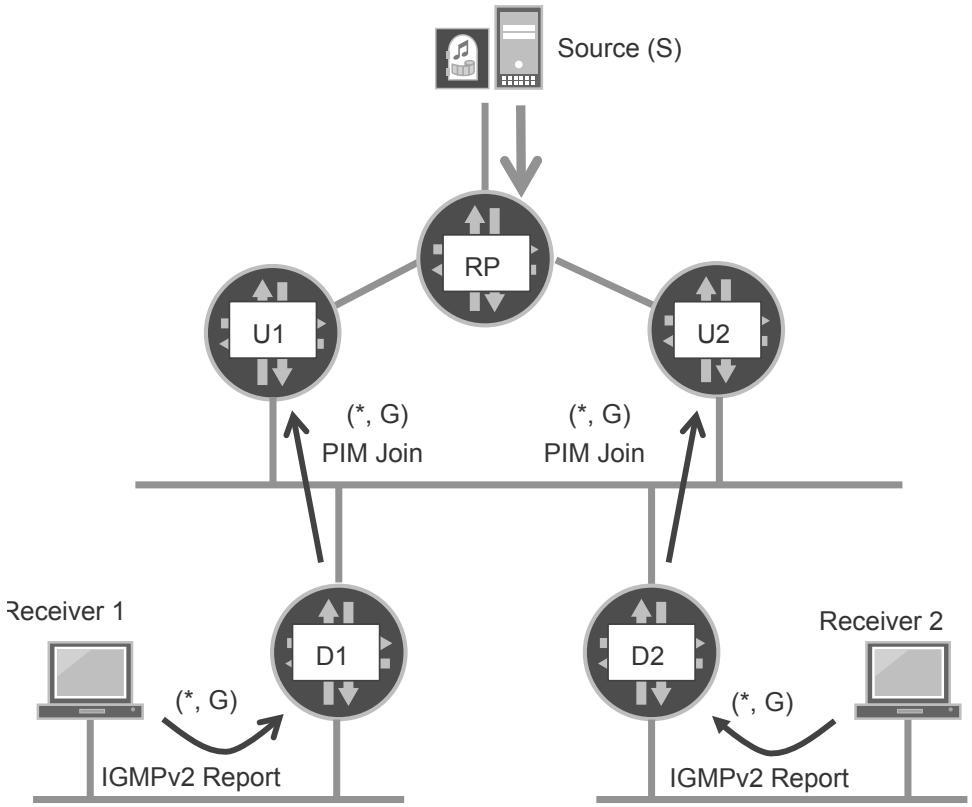


Figure 1.5 Duplicate Traffic Scenario Addressed by the PIM Assert Mechanism

NOTE You may be wondering why these complex soft-state mechanisms are worth noting, as this kind of topology is rare in a service provider network. Actually, the draft-rosen Multicast VPN solution is based precisely on a model where PE routers have per-VPN PIM adjacencies through a multi-access virtual segment.

MORE? Read RFC 4602 for a very detailed explanation of how PIM Sparse Mode works and all the different scenarios that it covers. This is a thick specification; do not forget to grab a cup of coffee first!

Interdomain Multicast

PIM has several limitations that make it unsuitable to be used as a standalone peering protocol for IP Multicast services across different Autonomous Systems (AS) in the Internet. Multicast Source Discovery Protocol (MSDP) is a TCP-based protocol enabling the exchange of information about active (S, G) flows, encoded in simple Source Active messages. Thanks to MSDP, a Rendezvous Point (RP) in a given AS can learn the active multicast sources present in all the peering AS's, hence allowing the local RP to generate PIM (S, G) Joins toward sources in remote AS's. In order for the PIM Join to be generated, RPF towards S needs to succeed, which raises a requirement about inter-AS unicast prefix reachability in core routers.

NOTE With BGP Multicast VPN implementation, Junos can help to bypass this requirement in a pure Internet (non-VPN) scenario. Note that this application is *not* discussed in this book.

Multiprotocol BGP with AFI=1 and SAFI=2 (address family `inet multicast`) can also be used in the Interdomain IPv4 Multicast context. Despite its name, this address family does not include any multicast information. Only unicast prefixes are exchanged, and this information is installed in an auxiliary unicast routing table or Routing Information Base (RIB). This RIB is only used for RPF checks towards the sources. In this manner, one AS can influence the PIM Joins generated by the peering AS (for example using MED for link selection) independently from the standard IP unicast routing. The equivalent functionality for IPv6 is achieved with AFI=2, SAFI=2 (address family `inet6 multicast`).

MORE? RFC 4611 provides a comprehensive description on how MSDP is currently being used in the real world. MSDP can also be used within an AS or enterprise network as the peering protocol among RPs in the Anycast model. For more specific details, RFC 3618 is the MSDP protocol specification. Find either at https://datatracker.ietf.org/.

BGP/MPLS VPN Refresher

BGP/MPLS VPN, also known as BGP/MPLS IP VPN or simply L3 VPN, stands for Multiprotocol BGP Virtual Private Network. As stated by RFC 4364 and its predecessor RFC 2547, this technology provides:

"*a method by which a Service Provider may use an IP backbone to provide IP Virtual Private Networks (VPNs) for its customers. This method uses a peer model, in which the Customer's edge routers (CE routers) send their routes to the Service Provider's edge routers (PE routers). CE routers at different sites do not peer with each other. Data packets are tunneled through the backbone, so that the core Provider routers (P routers) do not need to know the VPN routes. The primary goal of this method is to support the outsourcing of IP backbone services for enterprise networks. It does so in a manner which is simple for the enterprise, while still scalable and flexible for the Service Provider, and while allowing the Service Provider to add value.*"

The BGP/MPLS VPN architecture differentiates three roles that routers can play in the overall solution:

- CE (Customer Edge): IP device (host or router) connected to a single customer network in a single location, which can be managed by the Service Provider or

by the customer. A CE needs no visibility of the Service Provider network core. It typically exchanges customer routes with the attached PE(s) using regular IP protocols like RIP, OSPF, or BGP; or it may just have static routes pointing to the adjacent PE.

- PE (Provider Edge): Router managed by the Service Provider (SP) and connected to a set of CEs, each potentially servicing a different customer network. PEs exchange customer routes with each other using Multiprotocol BGP extensions. They also signal and maintain a mesh of transport tunnels able to carry traffic from one PE to another PE.
- P (Provider): Router managed by the Service Provider (SP), whose links are all internal to the SP backbone. P-routers provide the routing infrastructure to interconnect PE-routers with each other, acting as transit points of the transport tunnels. They keep no end customer routing state at all.

In this section, this book uses the terms *VPN* and *BGP/MPLS VPN* interchangeably. You can see in Figure 1.6 two Virtual Private Networks (VPNs), one black and one white, each corresponding to a different customer of the Service Provider. In order to interconnect the remote sites of the black customer network, PE1 and PE2 keep a dedicated VPN Routing and Forwarding table (VRF) associated to VPN black. The table contains all the routes advertised by the black CEs, namely CE1 and CE2. In this way, both PE1 and PE2 have complete visibility of the prefixes required to interconnect the black VPN sites.

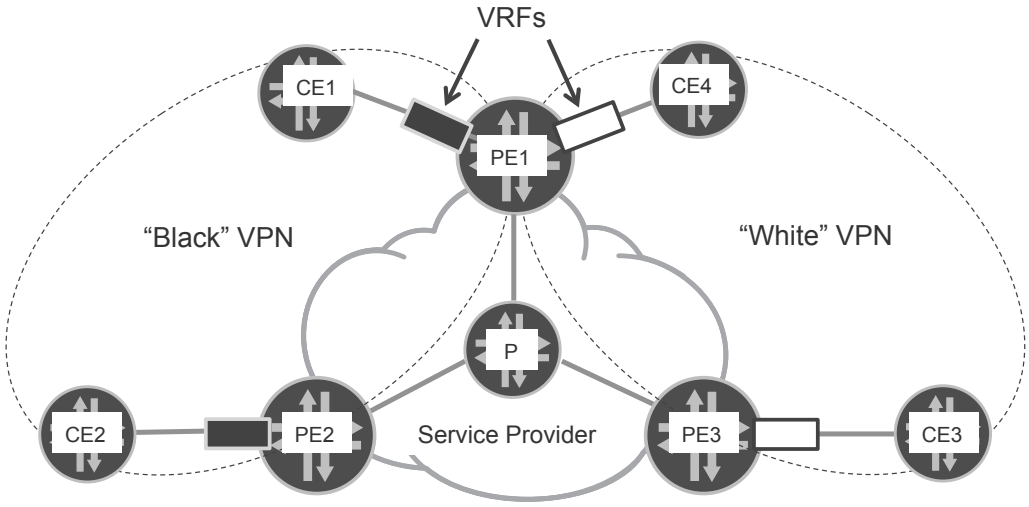


Figure 1.6 BGP/MPLS VPN Architecture

On the other hand, PE1 has two VRFs (black and white) with an independent set of routes. A single IP prefix like 192.168.1/24 can be present in both VRFs and point to completely unrelated next-hops, but this is not an issue as VPNs are private by definition. There is no association between the routes in VRFs black and white. One exception to this general rule is the extranet scenario, where two customers may decide to interconnect their private networks and leak certain prefixes between VRFs in one or several PEs. In Junos, a VRF can have multiple routing tables, like `black.inet.0` and `black.inet6.0` for IPv4 and IPv6 unicast prefixes, respectively.

From the perspective of a BGP/MPLS VPN customer, the Service Provider network (PEs + Ps) behaves as a big centralized router providing IP connectivity among its geographically distant network sites. Using a similar analogy, for a L2VPN or VPLS customer, the Service Provider network is like a big centralized switch (these technologies are not covered in this book).

MORE? *MPLS-Enabled Applications, Third Edition*, by Ina Minei and Julian Lucek (2011, Wiley & Sons Publishers) is an excellent source on MPLS advanced topics. For more information see www.juniper.net/books.

BGP/MPLS VPN Routes

PEs exchange Unicast VPN routes among themselves using Multiprotocol Border Gateway Protocol (MBGP), simply called BGP in this book. When a PE receives a BGP/MPLS VPN route from another PE, it needs to know which VPN the route belongs to.

In the scenario depicted in Figure 1.7, it could well be that PE2 advertises the route 192.168.1/24 for VPN white, while at the same time, PE3 announces the same prefix for VPN black. The receiving PE (PE1) needs to interpret each announcement in the context of the appropriate VPN. This requirement is addressed by Multiprotocol BGP Extensions, extending the idea of prefix/route to the more general concept of NLRI (Network Layer Reachability Information). A NLRI representing a BGP/MPLS VPN Route has three meaningful components: a Route Distinguisher (RD), a VPN label, and the prefix itself.

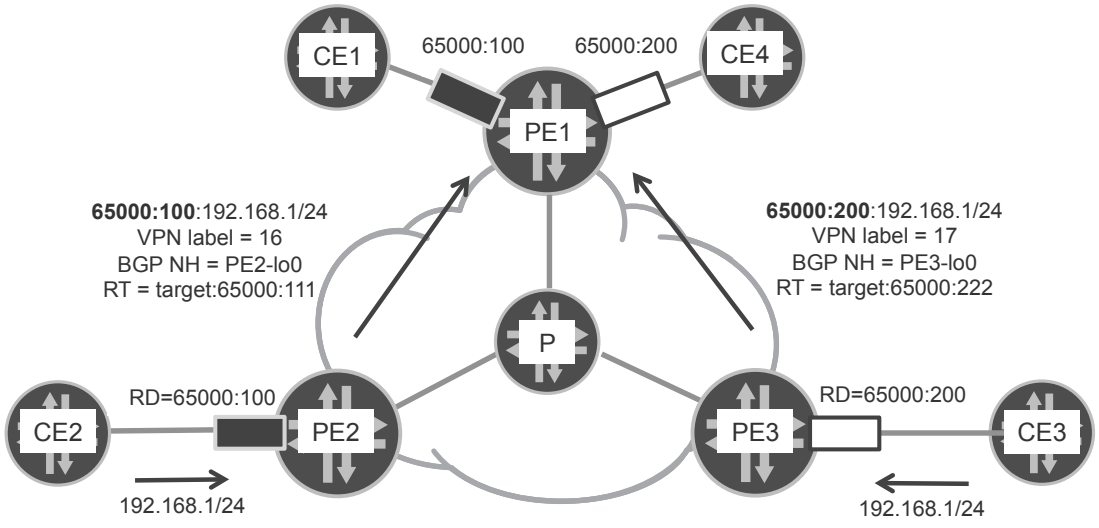


Figure 1.7 Route Distinguisher and VPN Label

With Multiprotocol BGP, each NLRI has an AFI (Address Family Identifier) and a SAFI (Subsequent Address Family Identifier), which are interpreted as the *route type*. BGP/MPLS VPN Routes have [AFI = 1, SAFI = 128] for IPv4, and [AFI = 2, SAFI = 128] for IPv6. In Junos these address families are called `inet-vpn unicast` and `inet6-vpn unicast`.

NOTE This book assumes IPv4 addressing as it is currently the most common choice, additionally the BGP/MPLS VPN technologies described are supported for IPv6 as well.

Each VRF has a different RD configured within a PE. In Figure 1.7, VRF black has RD = 65000:100, and VRF white has RD = 65000:200. When PE2 and PE3 advertise the 192.168.1/24 prefix in the context of VPN black and white respectively, they actually announce 65000:100:192.168.1/24 and 65000:200:192.168.1/24. This encoding makes the two prefixes different, even if the IPv4 address is identical, so the receiving PE routers do not run BGP best path selection between them. On the other hand, a BGP Route Reflector in the network would reflect both announcements. A common practice is to assign the same RD to all the VRFs of a given VPN, as depicted in Figure 1.7: 65000:100 for white and 65000:200 for black. Some service providers prefer to assign globally unique RDs per (PE, VRF) pair, but this book stays with the one global RD per VPN convention for simplicity.

A locally-significant VPN label is included in the NLRI as well. When PE2 advertises label 16, it is basically telling the rest of the PEs in the network: 'if you want to send me a packet targeted to 192.168.1/24 in the context of RD 65000:100, make sure I get it with MPLS label 16.' So BGP/MPLS VPN and MPLS are always related, regardless of the transport technology used. Even though different label values (16 and 17) are used by PE2 and PE3 to advertise the BGP/MPLS VPN prefixes, the values could have been identical since the VPN label is locally assigned by the advertising PE.

BGP/MPLS VPN routes have two key BGP attributes: the *BGP next hop* and a set of one or more *Route Targets* (RTs). The BGP next hop is normally set to the global loopback address of the advertising PE, and is key for the PE receiving the route to know where to tunnel the VPN traffic. The RTs are extended communities controlling the distribution of the BGP/MPLS VPN routes in the Service Provider. In simple VPN full-mesh topologies, with all VPN sites connected to each other in a non-hierarchical fashion, there is typically one RT per VPN. In the example, target:65000:111 could be used for VPN black, and target:65000:222 for VPN white. When a PE has to announce customer routes from its black VRF to other PEs, its `vrf-export` policies add the community target:65000:111 to the advertised prefix. On the other hand, configured `vrf-import` policies in receiving PEs install routes carrying RT target:65000:111 into VRF black, provided that the VRF is defined locally (not the case for PE3). More complex VRF import/export policies involving several route targets can be used for hub-and-spoke or for extranet scenarios.

NOTE SAFI 129 is used in the Junos operating system for address families `inet-vpn multicast` or `inet6-vpn multicast`, depending on the AFI value (1 and 2, respectively). This is the VPN equivalent to SAFI=2, described in the previous section, *Interdomain Multicast*, in this chapter. It allows population of an additional per-VRF RIB (e.g. `black.inet.2`) with unicast prefixes used for RPF only.

MORE? RFC 4364 is the BGP/MPLS VPN specification and contains a very comprehensive description of the solution. Find it at https://datatracker.ietf.org/.

Tunneling Technologies for BGP/MPLS VPN

One of the beauties of the BGP/MPLS VPN technology is the clear separation between the control and forwarding planes. Even though the Unicast VPN data

packets are always MPLS-tagged as they traverse the backbone (with at least the VPN MPLS label), the transport mechanism can either be based on MPLS or on other supported tunneling technologies, like GRE. The general architecture for BGP/MPLS VPN is illustrated in Figure 1.8.

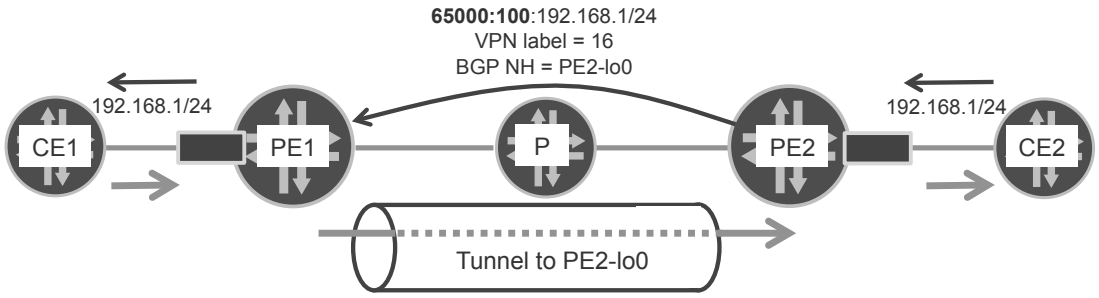


Figure 1.8 Transport Tunnels in BGP/MPLS VPN

In Figure 1.8 you can see that PE1 is the ingress PE (also called *head-end*) and PE2 is the egress PE (also called *tail-end*). They represent the points where the transit packets enter and exit the backbone, respectively. From the perspective of PE1, a data packet received from CE1 in VRF black with a destination IP in 192.168.1/24 will be forwarded to a tunnel next hop. This tunnel takes the packet through the backbone to PE2, whose loopback address is precisely the BGP next hop of the 65000:100:192.168.1/24 route. The transport tunnels are unidirectional, so the concepts of upstream (towards the ingress PE) and downstream (towards the egress PE) are applicable here in the same manner as in IP Multicast.

MPLS is the most common and flexible tunneling technology capable of transporting Unicast VPN data packets between two PEs. The tunnels based on MPLS are usually called Label Switched Paths (LSPs), and share similar switching principles as ATM or Frame Relay, in the sense that a MPLS label is a local identifier pretty much like an ATM VPI/VCI or a FR DLCI. There are several standard protocols to signal transport labels across the backbone. The most common ones in an intra-AS scenario are Label Distribution Protocol (LDP) and Reservation Protocol (RSVP). LDP is more plug-and-play, and RSVP is more feature rich, supporting things like Traffic Engineering, Link Protection, or Fast Reroute. RFCs and technical literature often refer to RSVP-TE (RSVP-Traffic Engineering) as the extension of RSVP used for MPLS LSP signaling. Figure 1.9 shows the establishment of a RSVP-based LSP, and the resulting forwarding path operations applied to end user data traffic.

As shown in Figure 1.9, PE1 and PE2 are, respectively, the ingress and egress PE. PE1 pushes two MPLS headers, prepended to the customer IP data packet. The inner MPLS label, or VPN label, is transported untouched throughout the backbone and its value (16) matches the label embedded in the BGP/MPLS VPN NLRI advertised by PE2. The outer MPLS label or transport label changes hop-by-hop as it traverses the P-routers in the backbone. Label value 3 has a special meaning, and implies a label pop operation. This behavior is called *Penultimate Hop Popping* (PHP). The last P router in the path (or penultimate hop if you count the egress PE) pops the transport label, so that the VPN label is exposed when the packet reaches the egress PE. The latter forwards the packet in the context of the VRF to which the VPN label is associated.

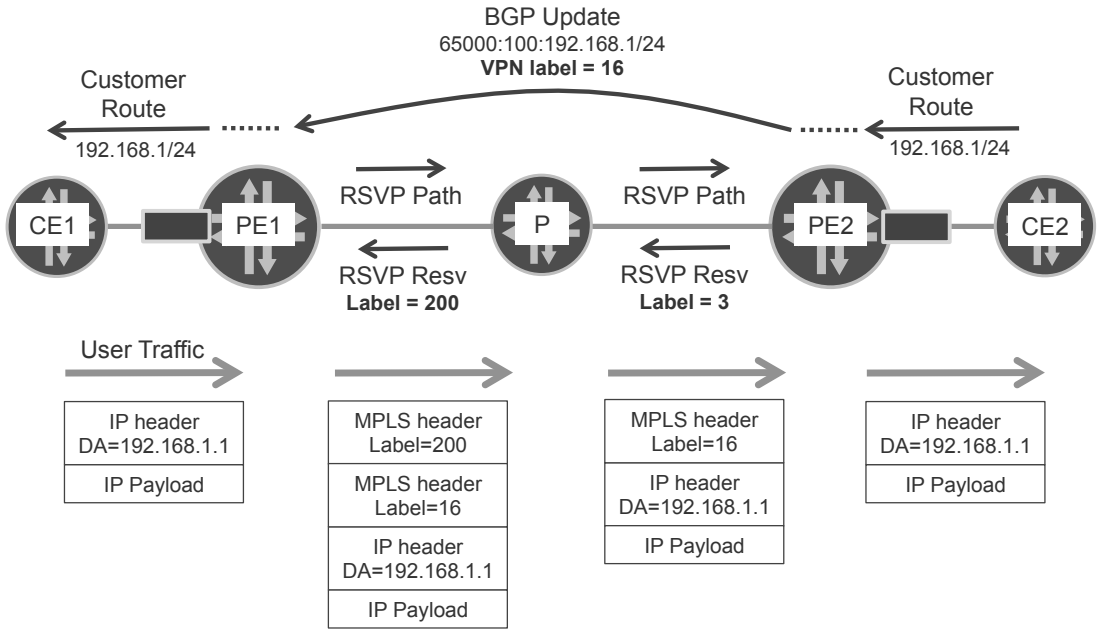


Figure 1.9 RSVP Tunnel Signaling (Path & Resv Messages) and MPLS Forwarding

The use of a two-label stack allows for a single tunnel to transport traffic belonging to many different VPNs between two PEs. For example, PE1 can encapsulate any Unicast VPN packet directed to PE2 by pushing outer MPLS label 200 and sending the packet to P. The inner MPLS label varies from one VPN to another and is used as a VPN identifier.

MPLS is not the only available tunneling technology for Unicast VPN data transport in the backbone. It is possible to encapsulate user packets with a single MPLS label (the VPN label) into GRE or L2TP at the ingress PE. In this way, the MPLS transport label is replaced by a GRE (RFC 4023) or L2TP (RFC 4817) header, achieving similar end-to-end results but with far fewer control plane features.

MORE? RFC 3209 explains RSVP-TE in detail, and is highly recommended reading. One of the fancy advantages of RSVP-TE is the possibility of having sub-second failure recovery with the use of Fast Reroute extensions (RFC 4090). Find them at https://datatracker.ietf.org/.

Try It Yourself: Different Tunneling Technologies

Draw a diagram similar to Figure 1.9, but this time use GRE *instead* of MPLS as the transport technology.

Past, Present and Future in Multicast VPN

A Multicast VPN (also known as MVPN) is simply a BGP/MPLS VPN with multicast services enabled. It typically requires unicast connectivity between sources and receivers: otherwise, RPF would never succeed. The Unicast VPN traffic is routed and forwarded by the same procedures previously described: BGP to signal customer routes, and point-to-point tunnels to transport unicast traffic through the backbone.

NOTE Throughout the rest of this book, the terms *Multicast VPN*, *MVPN*, and *VPN* are used interchangeably.

Multicast VPN technologies allow for multicast traffic to flow between different sites of the same VPN, traversing the service provider backbone in transport tunnels adapted to the point-to-multipoint nature of multicast. There is a wide range of available options to signal and transport multicast traffic. The MVPN technology chosen needs to cover two aspects: signaling customer multicast state between VPN sites and building the transport tunnels. Although both topics are related to each other, it is key to differentiate them in order to fully understand the MVPN solutions.

The prefixes C- and P- are widely used to differentiate the Customer and the Provider contexts, respectively. Applied to the BGP/MPLS VPN solution already described, C-Unicast routes are exchanged between PEs and CEs using any available C-Unicast routing protocol, and PEs peer with each other using BGP to exchange C-Unicast routes. Note that point-to-point P-Tunnels (usually MPLS-based) are built in the backbone to transport C-Unicast packets.

Moving to the multicast world, the multicast-enabled VRFs have a C-PIM instance running and establish C-PIM adjacencies with the directly attached CEs. Depending on the Multicast VPN flavor, C-Multicast routes are exchanged between PEs using BGP or C-PIM. The P-Tunnels used in MVPN to transport C-Multicast traffic across the backbone typically have a point-to-multipoint nature, and can be based on MPLS or GRE - although not all flavors support MPLS.

In some Multicast VPN models, there is also a P-PIM instance running in the service provider core. Note that C-PIM and P-PIM are not different protocols, they just represent different contexts of usage for PIM: Customer and Provider.

CAUTION It is a common mistake to think that anything with the P- prefix relates to a P-router. Although they both stand for Provider, some P- concepts have more to do with PE-routers than with P-routers. For example, in the P-Tunnels associated to Multicast VPNs, a set of PEs act as roots and leaves, while P's just act as transit and replication points.

Draft-rosen

Draft-rosen is the historical term given to the first model that was developed to solve the challenge of interconnecting IP multicast C-Sources and C-Receivers across a service provider backbone.

The idea behind draft-rosen is quite intuitive: make the backbone look like a LAN from the C-Multicast perspective. The C-PIM instance running in a VRF establishes PIM adjacencies with locally attached CEs and with VRFs of remote PEs. Figure

1.10 shows three PEs sharing Multicast VPNs black and white. The C-PIM instances of VRF black at the PEs are C-PIM neighbors of each other through the backbone, and the same applies to VRF white. The C-PIM virtual LAN interface associated to a Multicast VPN is called default MDT, where MDT stands for Multicast Distribution Tree. The terms MDT and P-Tunnel are used interchangeably in this section. There is a distinct default MDT for each Multicast VPN, one for black and another one for white. In this example, all PEs are attached to the same Multicast VPNs so the black and white MDTs have exactly the same branches, but they are instantiated as two independent P-Tunnels.

In draft-rosen, P-Tunnels are signaled using a master instance of PIM called P-PIM. You can assume for the moment that P-PIM is using the ASM model. Each Multicast VPN has a default MDT, typically associated to a unique P-Multicast group (P-G). The role of P-PIM is to build the distribution tree required to deliver P-G traffic to all the PEs attached to the MVPN. The PEs send a (*, P-G) P-PIM Join towards the P-RP (Provider Rendezvous Point) in order to join the default MDT. In this way, when a PE sends a packet with destination IP address equal to P-G, this packet reaches all the other PEs in the MVPN. The black and white MVPNs have different P-G addresses so they have independent default MDTs.

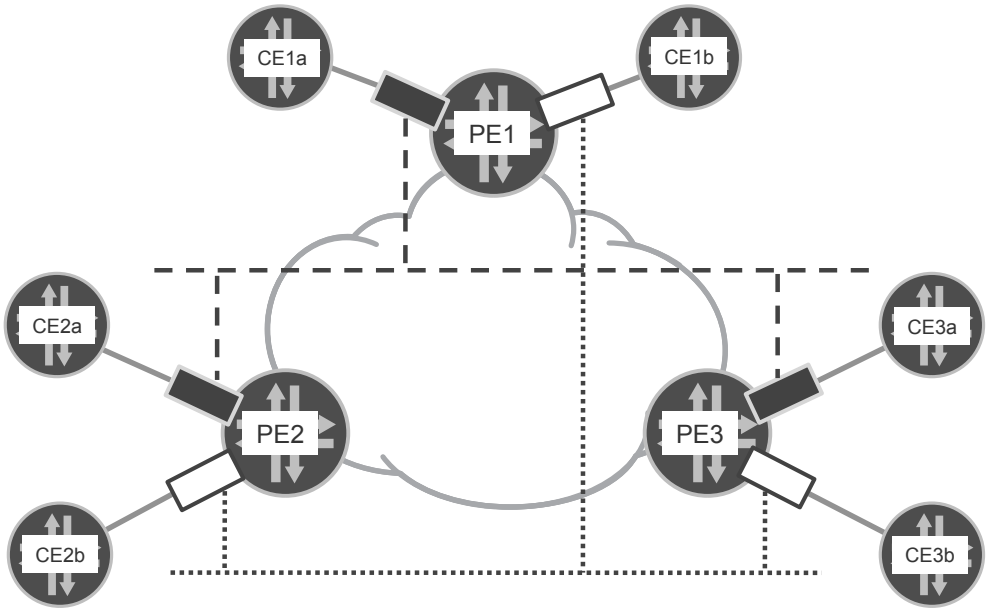


Figure 1.10 Multicast Distribution Tree in Draft-rosen

You can now see in Figure 1.11 the details of the forwarding and control planes in a branch of the default MDT connecting the two PEs. All the C-Multicast packets traversing the backbone in the context of black VPN are encapsulated in GRE. The external IP header has a multicast destination address: the black P-G.

One of the key aspects of the default MDT is that it not only carries user C-Multicast traffic, but also all the C-PIM control packets addressed to the well-known PIM multicast C-G address 224.0.0.13. This includes C-Hellos, C-Joins and C-Prunes, which travel encapsulated in GRE exactly in the same way as (C-S, C-G) end user traffic. C-Hellos dynamically trigger P-Registers and the creation of P-Tunnels by P-PIM ASM mechanisms.

On the other hand, C-Register packets are unicast so they traverse the backbone using the same set of point-to-point P-Tunnels as the rest of the Unicast VPN traffic. As for the P-PIM control packets, they are needed to build the default MDT and travel as plain IP within the backbone. Not all the signaling is included in Figure 1.11, for example, the following packets are omitted for simplicity: C-Hellos, P-Hellos, P-Registers (sent by PEs), and (P-S, P-G) P-Joins (sent by the P-RP).

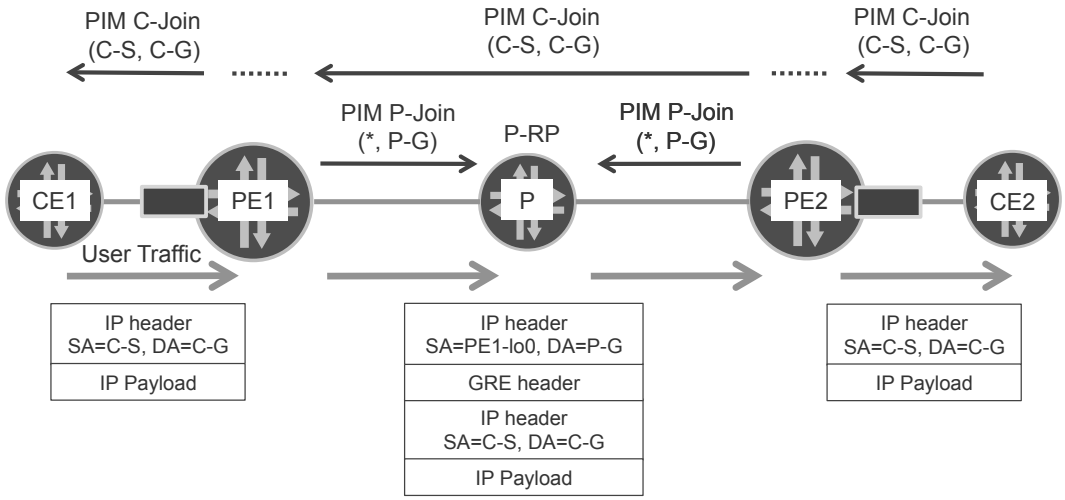


Figure 1.11 GRE Encapsulation in Draft-rosen

Draft-rosen introduces the concept of data MDTs, which overcome a limitation of the default MDT. By default, all the C-Multicast (C-S, C-G) packets are replicated by the default MDT to all the PEs attached to the MVPN. This happens even if a PE does not have downstream receivers for C-G. Each PE receives the multicast flows it has requested, as well as those requested by other PEs in the same VPN. This default behavior results in a suboptimal use of the bandwidth in the backbone. The alternative is signaling an additional P-Tunnel called data MDT with a different P-G address. Data MDTs carry certain (C-S, C-G) or (*, C-G) flows only, and the receiver PEs have the option to join them or not.

Finally, if P-PIM runs in SSM mode, Multiprotocol BGP with [AFI = 1, SAFI = 66] is used for (P-S, P-G) Auto-Discovery. This is the main function of BGP in draft-rosen. It covers just a piece of the whole solution.

NOTE Draft-rosen has recently become Historic RFC 6037: *Cisco Systems' Solution for Multicast in MPLS/BGP IP VPNs*. What is a Historic RFC? The answer is in RFC 2026: *A specification that has been superseded by a more recent specification or is for any other reason considered to be obsolete is assigned to the Historic level.* Although strictly speaking it is no longer a draft, most people in the industry (and in this book) still call it *draft-rosen*. The solution described can coexist with existing Unicast MPLS/BGP IP VPNs, however the multicast model it proposes is not based on MPLS and most edge signaling is achieved with PIM rather than BGP. Note that Junos supports this technology for backwards compatibility with existing draft-rosen deployments.

Assessment of Draft-rosen

Draft-rosen is not fully consistent with the standard BGP/MPLS VPN model: it does not rely on MPLS forwarding plane, and the inter-PE C-Multicast control plane is mainly based on C-PIM rather than BGP. This model does not reuse a protocol stack that has proven its value in terms of scaling and stability in networks across the world. Instead, it requires the validation and implementation of another completely different set of control core and edge protocols (PIM and GRE). Therefore, it cannot leverage the strength and flexibility of BGP/MPLS VPN technology.

In draft-rosen, PIM is the control protocol chosen both to instantiate the transport P-Tunnels and to signal C-Multicast routing state. Two instances of PIM (P-PIM and C-PIM) take care of the signaling that in BGP/MPLS VPN is performed by two different protocols (typically LDP/RSVP and BGP).

Due to the limitations of PIM, draft-rosen brings two new functional extensions based on other protocols. First, in order to support PIM SSM instead of ASM for the P-Tunnels, an auxiliary Multiprotocol BGP address family (MDT SAFI = 66) is defined with the specific purpose of default MDT Auto-Discovery. Second, a TLV-based protocol based on UDP is used for data MDT Auto-Discovery. So draft-rosen proposes three different protocols (PIM, BGP and TLV-based UDP) to accomplish the whole C-Multicast signaling functionality.

The most significant characteristic of draft-rosen described here is the use of PIM as the protocol to convey C-Multicast state. The default MDT is functionally equivalent to a LAN from the point of view of C-PIM. This means that every single C-Hello, C-Join, and C-Prune, with destination C-Multicast address 224.0.0.13, is received and processed by all the PEs attached to the Multicast VPN. Since PIM is a soft state protocol, all these messages are periodically resent and reprocessed, which brings scaling concerns at the PE control plane as the number of Multicast VPNs, sites per VPN, and customer flows per site increase.

There is one separate set of PIM adjacencies per VPN, unlike BGP/MPLS VPN where one single BGP session can carry C-Routes of different VPNs.

Additionally, PIM is a cumbersome protocol in a LAN. It relies on a series of extra mechanisms (Designated Router election, Unicast Upstream Neighbor, Asserts, Join Suppression, Prune Delay) that increase significantly the complexity of the operation and troubleshooting of this solution. This becomes more evident as the number of C-PIM neighbors in the Virtual LAN (or, in other words, PEs in a Multicast VPN) increases. The extra overhead or noise becomes proportionally more relevant than the useful part of the C-PIM signaling. The lack of efficiency in the control plane is not only a concern from a pure scaling perspective (how many sites or flows fit in a PE) but also from the point of view of service convergence speed upon failure.

Last but not least, draft-rosen only supports the use of one tunneling technology (P-PIM/GRE) to transport C-Multicast traffic through the backbone, as compared to BGP/MPLS VPN where the use of BGP to signal customer routing state allows for a complete decoupling between the control and forwarding planes, bringing the support of multiple tunneling technologies (MPLS, GRE, etc.). All of which brings us to the topic of this book.

BGP Multicast VPN

BGP Multicast VPN, formerly known as Next-Generation Multicast VPN, uses BGP control plane and offers a wide variety of data planes. It is a flexible solution that

leverages the MPLS/BGP technology and cleanly addresses all the technical limitations of draft-rosen.

NOTE Most topics mentioned in following paragraphs will be illustrated during the upcoming chapters, so do not worry too much if there is a concept or definition that remains difficult to understand.

RFC 6513 (Multicast in MPLS/BGP IP VPNs) is the result of a multivendor effort to achieve a common specification for Multicast VPN in the industry. It is quite extensive and includes all the possible Multicast VPN alternatives, with a sensible weighing of pros and cons. This standard defines a generic terminology that applies both to the draft-rosen model and to BGP Multicast VPN. It classifies the PEs depending on their role in each Multicast VPN:

- Sender Sites set: PEs in the Sender Sites set can send C-Multicast traffic to other PEs using P-Tunnels. The terms *Sender PE* and *Ingress PE* are used in this book for the sake of brevity.
- Receiver Sites set: PEs in the Receiver Sites set can receive C-Multicast traffic from P-Tunnels rooted on other (Sender) PEs. The terms *Receiver PE* and *Egress PE* are used in this book, again for the sake of brevity.

One PE can be both Sender and Receiver in the same VPN. Every time you read the words Sender, Receiver, Ingress, or Egress, keep in mind that they are used in the context of one specific VPN and even one C-Multicast flow. It is perfectly possible for one PE to be Sender for VPN black, Receiver for VPN white, and both Sender and Receiver for VPN grey.

RFC 6513 also defines the general concept of PMSI (P-Multicast Service Interface) as the virtual interface that a Sender PE uses to put C-Multicast traffic into a P-Tunnel. The P-Tunnel is point-to-multipoint in nature and takes the traffic to a set of Receiver PEs. It is very common to name the P-Tunnel as a *Tree*, where the Sender PE is the *root* and the Receiver PEs are the *leaves*.

Every Sender PE is the root of at least one P-Tunnel. In other words, an Ingress PE requires at least one PMSI in the VRF to send C-Multicast traffic to other sites. There are two types of PMSIs: *Inclusive* (I-PMSI), and *Selective* (S-PMSI). In this book you will see the terms Inclusive Tree and Selective Tree very often. In the language of draft-rosen, the Inclusive Tree is the default MDT, and a Selective Tree is a data MDT. A Sender PE can have up to one I-PMSI, and an unlimited number of S-PMSI's, in a given Multicast VPN.

When a Sender PE tunnels a C-Multicast packet (C-S, C-G) into the core, it uses by default the I-PMSI. The Inclusive Tree takes the packets to all the Receiver PEs in the Multicast VPN. The Egress PEs in turn forward the traffic only to the attached CEs having signaled (C-S, C-G) or (*, C-G) state – either through a C-PIM join or via IGMP. If a Receiver PE has no such downstream CE, it silently discards the packet.

The Inclusive Tree may result into a waste of bandwidth resources. This is especially true if certain C-Multicast flows have high bandwidth requirements and a small number of C-Receivers. Specific (C-S, C-G) or (*, C-G) flows can be optionally mapped to a S-PMSI. Selective Trees connect the Sender PE to the subset of Egress PEs with interested downstream C-Receivers for the transported flows.

By default, there is a one-to-one mapping between a PMSI and the P-Tunnel it points to. On the other hand, a PMSI is dedicated to a VRF. Putting these two facts together, a P-Tunnel by default only carries traffic from one Multicast VPN. The concept of *Aggregate Trees*, explained at the end of this chapter, changes the default behavior.

BGP Control Plane in a Nutshell

RFC 6514 (BGP Encodings and Procedures for Multicast in MPLS/IP VPNs) is a multivendor specification for the BGP signaling required in BGP Multicast VPN. This document introduces a new address family (MCAST-VPN SAFI = 5). This new NLRI includes a set of route types covering the whole functionality of MVPN edge signaling. The knowledge investment required to learn this new route set is largely compensated for by the flexibility and operational strength of the solution.

There are three major features that any Multicast VPN solution needs to address:

- Site Auto-Discovery: In BGP/MPLS VPN, there is no specific Auto-Discovery mechanism – a PE gets to know remote sites as it receives C-Unicast routes from other PEs. On the other hand, Multicast VPN requires prior knowledge of the Sender and Receiver Sites sets before starting to exchange C-Multicast routing state. BGP Multicast VPN always relies on BGP for site Auto-Discovery. In most draft-rosen implementations, it is initiated either by C-PIM Hellos or by BGP, depending on the P-Tunnel type (P-PIM ASM or P-PIM SSM, respectively).
- C-Multicast Routing State Signaling: BGP Multicast VPN uses BGP to signal C-Join and C-Prune states between sites. Draft-rosen uses C-PIM adjacencies between PEs.
- P-Tunnel Signaling: MPLS-based P-Tunnels are supported in BGP Multicast VPN. This is the preferred option, since it is a feature-rich transport technology. It is also possible to signal P-Tunnels with P-PIM and rely on GRE Multicast for transport, like in draft-rosen.

As you can see in Table 1.1, BGP Multicast VPN perfectly aligns with the classical BGP/MPLS VPN model in terms of protocols and technologies supported for each of these functions.

Table 1.1 Multicast VPN Technology Support Matrix

Technology	Site Autodiscovery		C-Route Signaling		Tunneling Supported	
	BGP	C-PIM	BGP	C-PIM	MPLS	GRE
BGP/MPLS VPN	Yes	No	Yes	No	Yes	Yes
BGP Multicast VPN						
Draft-rosen	Yes	Yes	No	Yes	No	Yes

There are seven different route types within MCAST-VPN NLRI, each of them used to signal a different kind of state within the Multicast VPN context. These route types can be classified in two major functional groups:

- Provider Routes (P-Routes): The P-Routes are needed to determine what PEs are members of each Multicast VPN, and provide information about the

P-Tunnels that need to be signaled. Basically, it's what the roots and the leaves of each tree are.

- Customer Routes (C-Routes): These routes signal C-Multicast routing state, functionally equivalent to C-Joins or C-Registers, depending on the route type.

Although the definition of P-Route and C-Route functional groups is not included in the existing drafts, we use the terminology in this book for educational purposes only. Both P-Routes and C-Routes are exchanged among PEs using MCAST-VPN NLRI. Table 1.2 lists all the existing route types. You may have noticed that the types 3 and 4 are considered both P-Routes and C-routes, as they contain information about both provider and customer contexts. The structure and usage of each type is fully explained throughout the different chapters of this book.

Table 1.2 MCAST-VPN Route Types ('A-D' Stands for Auto-Discovery)

Route Type	Route Name	C-PIM Equivalence	Functional Group
1	Intra-AS I-PMSI A-D Route	C-Hello	P-Route
2	Inter-AS I-PMSI A-D Route		
3	S-PMSI A-D Route		P-Route & C-Route
4	Leaf A-D Route		
5	Source Active A-D Route	(C-S, C-G) C-Register	C-Route
6	C-Multicast route – Shared Tree Join	(*, C-G) C-Join	
7	C-Multicast route – Source Tree Join	(C-S, C-G) C-Join	

PEs in the Sender Sites set include a PMSI Tunnel attribute in the type 1, 2, or 3 routes they generate. This attribute specifies the type and identifier of a P-Tunnel rooted at the advertising PE. There is a wide range of P-Tunnel technologies compatible with BGP Multicast VPN. Table 1.3 lists the technologies defined in RFC 6514, as well as the minimum Junos operating system version that supports each of them.

Table 1.3 MCAST-VPN Tunnel Types

Tunnel Type	Tunnel Type Name	Description	Junos Support
1	RSVP-TE P2MP LSP	Point-to-Multipoint Label-Switched-Path signaled with RSVP-TE	8.5R1
2	mLDP P2MP LSP	Point-to-Multipoint Label-Switched-Path signaled with LDP	11.2R1
3	PIM-SSM Tree	GRE Multicast transport, PIM SSM signaling (P-S, P-G)	10.0R2
4	PIM-SM Tree	GRE Multicast transport, PIM ASM signaling (*, P-G)	8.4R1
5	PIM-Bidir Tree	GRE Multicast transport, PIM BiDir signaling	No
6	Ingress Replication	Set of Point-to-Point Label-Switched-Paths, LDP or RSVP	10.4R1 (RSVP only)
7	mLDP MP2MP LSP	Multipoint-to-Multipoint Label-Switched-Path with LDP	No

MORE? RFCs 6513 and 6514 can be viewed at https://datatracker.ietf.org/.

BGP Multicast VPN Assessment

The Junos operating system provides unmatched flexibility in the choice of the P-Tunnel technology. The multipoint-to-multipoint solutions (PIM-Bidir, MP2MP LSP) are not supported to date for one reason: lack of customer demand due to the asymmetric nature of most real-world multicast deployments. One of the beauties of BGP MVPN is the complete decoupling between the propagation of the C-Multicast routing state between PEs (always based on BGP) and the signaling of the transport P-Tunnel (wide range of options). A simple BGP attribute, the PMSI, does all the magic as the Sender PE has complete freedom to specify the P-Tunnel it uses to inject C-Multicast traffic in the backbone. This decision is local to the PE, and it is perfectly possible to have half of the Sender PEs using PIM-SSM P-Tunnels, and half of them using RSVP-TE P2MP LSPs. The Receiver PEs would pull C-Multicast traffic from two types of P-Tunnels, depending on which Sender PE the traffic is coming from. This is completely supported by the specification as well as by Junos OS implementation, and it greatly simplifies the tunneling technology migration tasks . The only exception to this rule is Ingress Replication, which requires that all PEs of a given VPN agree on the P-Tunnel transport mechanism.

One of the key advantages of BGP MVPN over the draft-rosen model is the way the PEs peer with each other. First, the PEs no longer establish and maintain per-VPN full-mesh adjacencies. Each BGP session carries information about multiple VPNs. There is just one session with each BGP peer, as compared to C-PIM where a PE needs to maintain separate adjacencies for each Multicast VPN. Suppose a PE has 100 Multicast VPNs, and peers with 100 PEs on each VPN. With draft-rosen, it needs to keep 10,000 adjacencies. With BGP, it needs to keep just 100 sessions, which can be used to exchange routes of other address families as well. This number can be further reduced if BGP Route Reflectors come into play.

The BGP packets are typically transported as internal IP traffic in the backbone so the control and forwarding planes are nicely separated, unlike draft-rosen where C-PIM packets are encapsulated by default in the same P-Tunnel (default MDT) as the C-Multicast user traffic.

Since BGP is based on TCP, transport reliability is ensured and there is no need to keep soft C-Multicast state anymore. In other words, the periodic flooding of C-Joins and C-Prunes is no longer needed. Just get the corresponding BGP route announced once to all the BGP peers (a few RRs) and the job is done.

Last but not least, the use of special BGP communities and route targets allow for Source Tree Join and Shared Tree Join BGP routes (functionally equivalent to C-PIM Joins) to be imported only by the chosen upstream PE. This is quite different from C-PIM in draft-rosen, where all C-Join and C-Prune packets are flooded to 224.0.0.13 and hence processed by all the PEs in the Multicast VPN. Not to mention all the complex mechanisms (DR election, assert, join suppression, prune delay) that are cleanly avoided by using BGP.

In summary, BGP Multicast VPN brings adjacency state reduction, control and forwarding plane decoupling, P-tunnel choice flexibility, and removes the need for C-Multicast soft state refresh, as well as unnecessary C-Join and C-Prune flooding and processing. For all these reasons, it is a more scalable model and justified in its ranking as a Next Generation solution since the time it was released.

After this technical comparison, it is useful to check out what the service provider community is currently recommending. RFC 6513 is quite agnostic and describes all the possible Multicast VPN solutions. This brings the debate of choosing the minimal feature set that should be expected from a vendor claiming Multicast VPN support. In that sense, RFC 6517 (Mandatory Features in a Layer 3 Multicast BGP/MPLS VPN Solution) is a useful reference as it provides the pulse of several major worldwide service providers regarding the requirements for modern MVPN solutions. Here are some key paragraphs of this vendor-agnostic draft.

The recommendation is that implementation of the BGP-based auto-discovery is mandated and should be supported by all Multicast VPN implementations.

It is the recommendation of the authors that BGP is the preferred solution for S-PMSI switching signaling and should be supported by all implementations.

Although draft-rosen supports BGP-based Auto-Discovery via MDT SAFI, it defines a TLV-based UDP protocol (and not BGP) for the S-PMSI signaling. To date, the only technology meeting both requirements is BGP MVPN.

Deployment Options for BGP Multicast VPN

Since the BGP MVPN technology cleanly decouples the control and data planes, you can choose C-Multicast and P-Tunnel flavors independently of each other. Let's first choose a technology for the P-Tunnel, then move on to the C-Multicast part.

Among the different tunneling technologies, the P-PIM ASM/SSM approach is common in multivendor backbones with incomplete support of Point-to-Multipoint MPLS technology. It is also used as an interim solution when migrating from a draft-rosen deployment to BGP Multicast VPN. Although P-PIM is fully supported in Junos BGP MVPN implementation, the PIM protocol itself is far from being carrier-class in terms of robustness, flexibility, predictability, and operability. Choosing P-PIM also requires GRE encapsulation in the backbone, which might sometimes be an issue in terms of MTU, fragmentation, and reassembly. On the other hand, MPLS is the de-facto core transport technology in the industry. Since BGP MVPN introduces the support of tunneling C-Multicast packets in MPLS Label Switched Paths, it makes sense to illustrate it in this document. A complete reference would provide examples of both P-PIM and MPLS, but it simply is not feasible to cover all the options here.

As for the particular flavor of MPLS LSPs to be used, RSVP-TE (or simply RSVP) has the clear advantage of supporting Traffic Engineering (TE) extensions, which bring the possibility of: performing bandwidth reservations, signaling bypass LSPs to achieve sub-second service restoration, choosing the paths according to link colors, keeping per-LSP statistics, and a wide range of options that make it the most flexible approach. The other MPLS alternatives are Ingress Replication and LDP. The latter does not have TE extensions, while Ingress Replication relies on a set of Point-to-Point (P2P) LSPs.

With Ingress Replication (IR), the Sender PE uses a downstream allocated MPLS label to tag Multicast VPN packets and send them through the same P2P LSPs that carry Unicast VPN traffic. IR is completely transparent to the P-routers since no extra tunneling state needs to be created to transport C-Multicast traffic. On the other hand, the Sender PE needs to send one different copy of each packet to each Receiver PE, which in certain cases results in a loss of bandwidth efficiency, as compared to all the other P-Tunnel technologies.

RSVP P2MP is the chosen P-Tunnel flavor in this book. BGP MVPN fully supports dynamic signaling, so it is not mandatory to statically configure the endpoints of the P2MP LSPs. Static LSPs are still an option, but this book focuses on the dynamic ones, which better illustrates the concept of Auto-Discovery.

Once the P-Tunnel technology has been chosen, let's focus on C-Multicast. Due to a high variety of customer deployments, it makes sense to illustrate both the C-PIM SSM and ASM architectures. Chapters 2 and 3 focus on C-PIM SSM and ASM, respectively. The SSM model is quite straightforward in terms of signaling, on the other hand ASM has a wider range of implementation options. In the next two chapters the P-Tunnel technology is RSVP-TE, so PIM (C-PIM) only plays a role in the PE-CE links.

Most of the examples in this book use Inclusive P-Tunnels, but Chapter 4 focuses on an example of Selective P-Tunnels for completeness.

The concept of aggregation is not covered in the rest of the book. Aggregate P-Tunnels use a MPLS label allocated by the Ingress PE to carry traffic from different Multicast VPNs. With aggregate P-Tunnels, several PMSIs are bound to the same P-Tunnel. This approach has the advantage of reducing the amount of P-Tunnel signaling and state, but its efficiency in terms of traffic flooding strongly depends on the receiver site distribution. A downstream PE would get traffic from all Multicast VPNs mapped to a given Aggregate P-Tunnel, even if it only has receiver sites for one MVPN. Junos OS already supports upstream label allocation infrastructure, but aggregate P-Tunnels are not implemented yet in shipping code. Hence, you can safely assume that P-Tunnels are non-aggregate and there is a 1:1 mapping between PMSI and P-Tunnel, and this actually simplifies the interpretation of the PMSI concept.

NOTE Inter-AS mechanisms are not discussed in this book.

MORE? The upstream label allocation mechanism is fully described in RFC 5331, and can be viewed at at https://datatracker.ietf.org/.

Answers to Try It Yourself Sections of Chapter 1

Try It Yourself: Test Your PIM Expertise

Most people believe that the RPT (RP-tree) always traverses the RP, which is a very reasonable assumption. This is not always the case, though. In this scenario, the RP is indirectly triggering a SPT-switchover when it sends the PIM (S, G) Join upstream. Figure 1.12 depicts the signaling and steps involved.

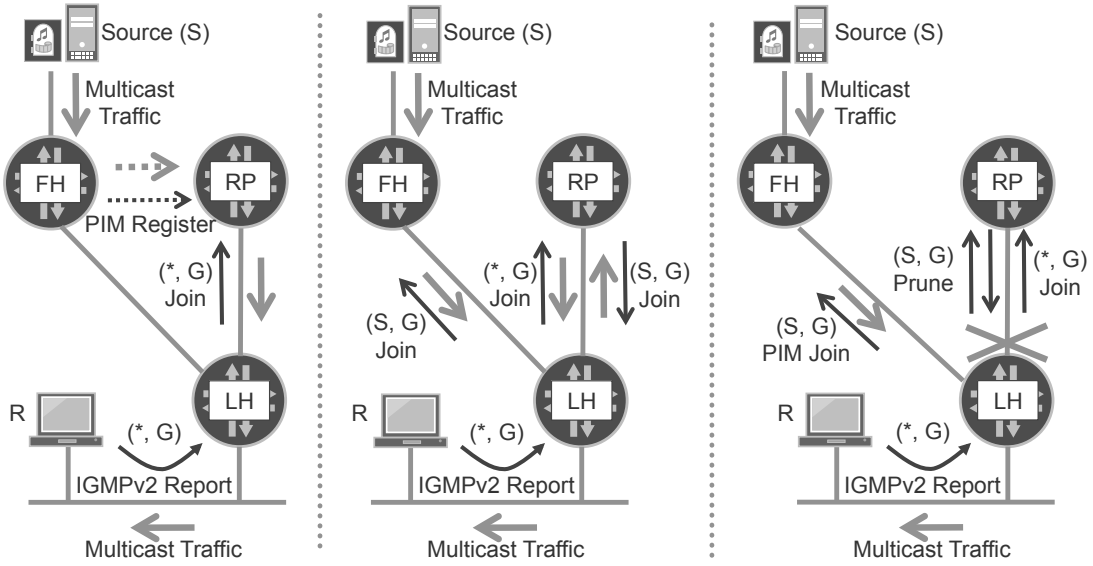


Figure 1.12 Rendezvous Point in a Stick

The full sequence of events is:

1. The receiver sends an IGMP (*, G) Report.

2. The last-hop router (LH) sends a PIM (*, G) Join towards the RP.

3. The source S starts to send multicast traffic to group G.

4. The first-hop router (FH) encapsulates the multicast traffic into unicast packets called PIM Register-Start, or simply Registers. The Registers are sent unicast to the RP.

5. The RP decapsulates the Registers and sends the native multicast traffic down the Shared Tree to LH.

6. The RP sends a PIM (S, G) Join towards FH, which is first processed by LH.

7. LH forwards the PIM (S, G) Join towards FH.

8. FH forwards the multicast traffic both natively and encapsulated in Registers to the RP.

9. The RP sends a PIM Register-Stop to FH, so as to stop the Registers flow.

10. LH sends a PIM (S, G) Prune to the RP, as it is already receiving the flow through the SPT.

11. The RP no longer has downstream (S, G) Join state so it sends a PIM (S, G) Prune upstream to LH.

12. The bidirectional (S, G) Prune state between LH and RP effectively stops the (S, G) data traffic in that link.

13. FH sends the multicast traffic natively to LH only.

14. LH only sends the traffic to its local receivers.

15. FH periodically sends a Register-Start packet to RP, which replies with a Register-Stop.

Try It Yourself: Different Tunneling Technologies

Does your diagram look like Figure 1.13? If so, well done!

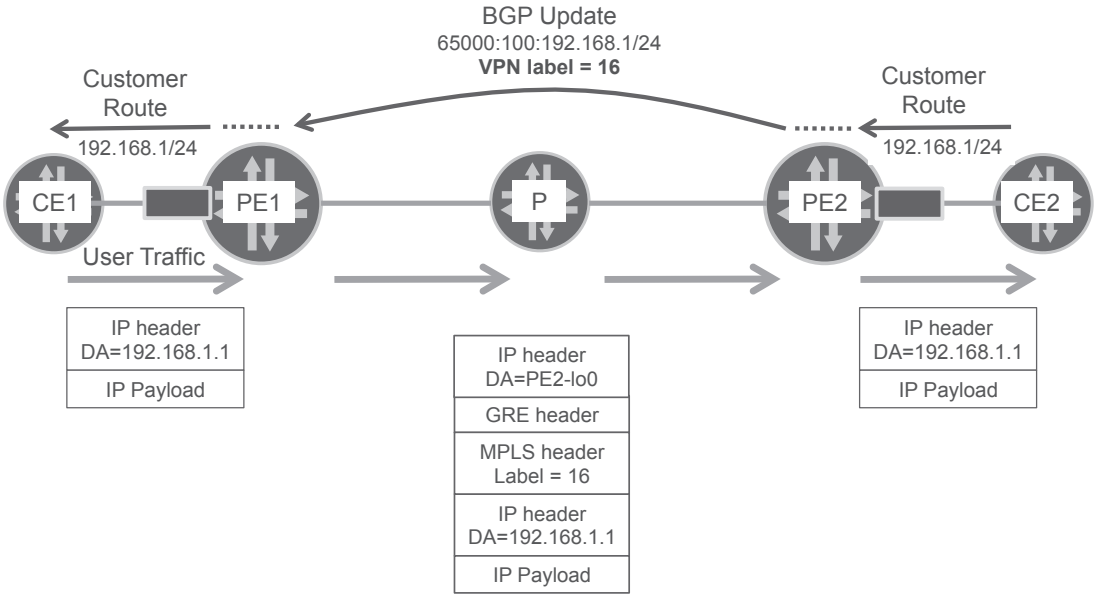


Figure 1.13 Unicast VPN Traffic transported with GRE P-Tunnels

Chapter 2

BGP Multicast VPN with PIM SSM as PE-CE Protocol

In this chapter you are going to build a fully working BGP Multicast VPN scenario. As explained in the Chapter 1, the P-Tunnels that you configure in this book are based on Point-to-Multipoint (P2MP) LSPs signaled with RSVP-TE. Since the CEs interact with the PEs using PIM (C-PIM), two different models are possible: Any Source Multicast (ASM) and Source Specific Multicast (SSM). The latter model is simpler in terms of signaling since the PIM C-Joins already contain the IP addresses of the sources (C-S), hence the router receiving the C-Join does not need to rely on any source discovery process. Due to its simplicity, let's start with the SSM model, and leave ASM for Chapter 3.

Building the Baseline Scenario

In order to experiment with the BGP Multicast VPN feature set, you first need a working BGP/MPLS VPN test bed. This requires some infrastructure investment and some time to build the physical and logical topology. Figure 2.1 shows the topology and IP addressing scheme used throughout this book.

Refer to the Appendix, where you can find the initial configuration of CE1, PE1 and P. The configurations of the rest of the elements (CE2, CE3, CE4, PE2, PE3, PE4) are similar, and can be built by just modifying IP addresses and interface names according to Figure 2.1.

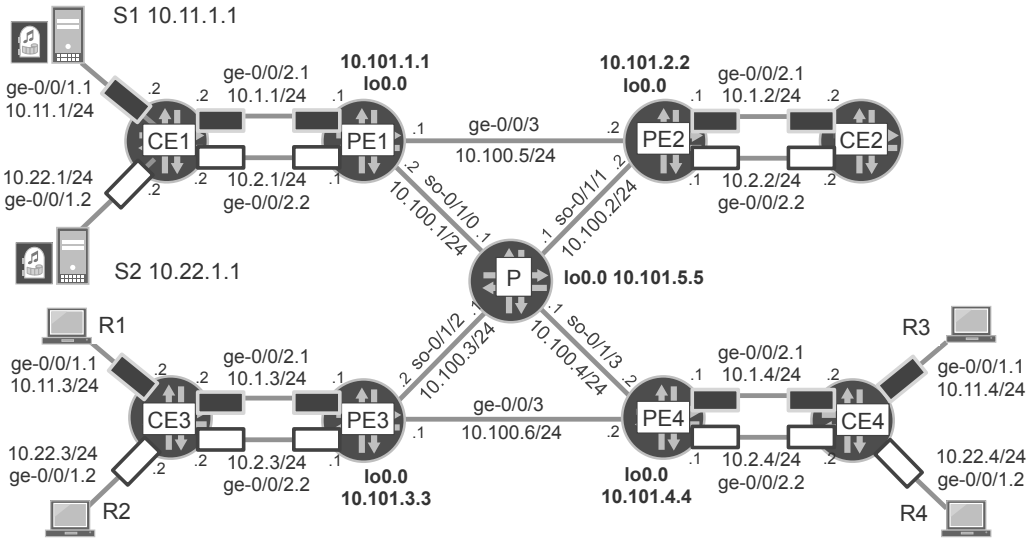


Figure 2.1 Physical Connectivity and IP Addressing of the Test Bed

NOTE The hardware paths of the interfaces (like FPC and PIC slot numbers) are *not* important for the test bed. The usage of other types of access or core interfaces is also allowed, as long as the core links support `vrf-table-label`.

As an alternative to `vrf-table-label`, vt- (virtual tunnel) interfaces can be used. This requires tunnel-capable hardware, for example any MX-series Dense Port Concentrator (DPC) or Modular Port Concentrator (MPC) can be configured to enable Tunnel Services.

TIP If you are on a low budget and plan to use Logical Systems (LS) to simulate each PE/P router, make sure you use vt- interfaces and not `vrf-table-label`. The author did not verify whether LS cover the whole feature set included in this book or not.

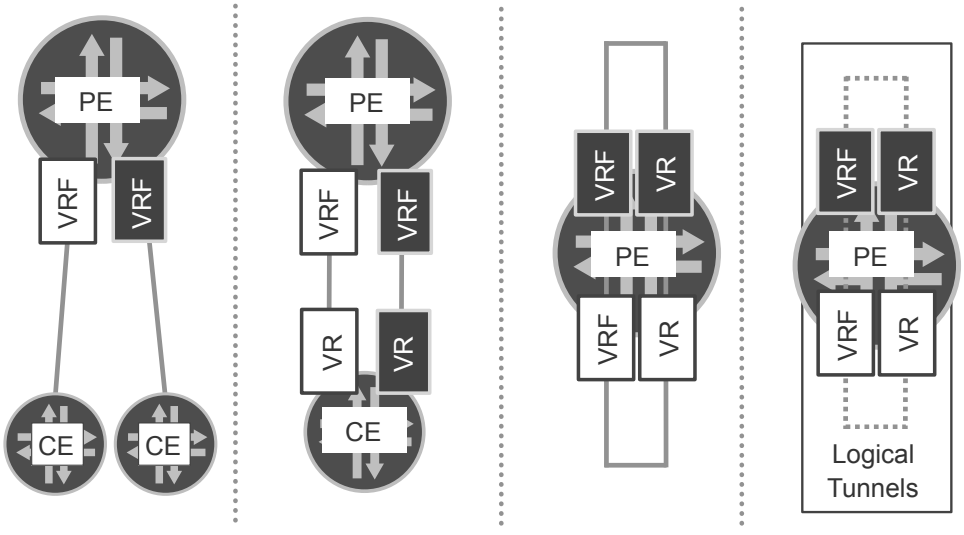
You'll need five routers running Junos to make up the backbone: four routers performing a PE role, and one dedicated to the P function (and BGP Route Reflector). The test bed used to prepare this book consisted of five M-series routers running Junos OS 10.4R9.2. Note that the technology described in this chapter has been supported since Junos OS 8.5.

CAUTION

If you are planning to run Junos OS versions older than 9.2 in some P/PE routers, and 9.2 or above in other P/PE routers, make sure you configure `protocols rsvp no-p2mp-sublsp` in those with newer versions. This knob is only required for backwards compatibility. Do not use it if all the P/PE routers are running 9.2 or above.

As for the CEs, there are several alternatives illustrated in Figure 2.2. Choose one of the following options:

- Connect an independent physical CE to each VRF at the neighboring PE. With this option, you need eight CE routers.
- Connect one single CE to each PE. The CEs have two VRs (Virtual Routers), black and white. The black VR peers with the black VRF at the neighboring PE, and the same applies to white VR and VRF. This is the model used in the examples of this book, but you can use any other option by adapting the configurations accordingly. With this option, you need four CE physical routers.
- Configure VRs at the PE, and interconnect VRs and VRFs by using back-to-back cables linking two different ports of the PE. With this option, you need no CE physical router, just extra ports at the PE.
- Configure VRs at the PE, and interconnect VRs and VRFs by using a lt- (logical tunnel) interface. With this option, you need no CE physical router, but additional hardware may be required (for example, a Tunnel PIC).



One VR (Virtual Router) is equivalent to an independent CE.

Figure 2.2 Several Options to Deploy or Simulate CE Routers

MORE? VRF and VR are just two types of routing instances. A VRF is configured with `instance-type vrf`, and requires route distinguishers and route targets in its definition. A VR is configured with `instance-type virtual-router` and cannot be used in the context of BGP/MPLS VPN services, because it does not interact with Multiprotocol BGP. Each VR instance behaves virtually like a separate router.

Unicast Protocols

The VRs have just a default static route pointing to the neighboring VRF at the PE, for example VR black at CE1 has the route 0.0.0.0 next-hop 10.1.1.1. The VRFs also have a specific static route pointing to the connected CE/VR, for example VRF black at PE1 has the route 10.11.1/24 next-hop 10.1.1.2.

The Interior Gateway Protocol (IGP) configured within the backbone (PEs and P) is IS-IS level 2. It is worthwhile to note that OSPF is also a supported option.

As for MPLS, the label control protocol used here is LDP. Why not RSVP? Actually, using RSVP to signal both point-to-point (P2P) and point-to-multipoint (P2MP) tunnels is a fully-supported and valid option. The reason the author chooses LDP here is to stress the fact that if a particular network has LDP already running, there is absolutely no need to migrate the Unicast VPN services from LDP to RSVP tunnels. It is perfectly possible to keep LDP for P2P/Unicast, while deploying RSVP for P2MP/Multicast. Each protocol would carry information about different Forwarding Equivalence Classes (FECs), so there is no functional overlap. LDP and RSVP are two parallel options that the router will use depending on whether the service is unicast or multicast.

The Unicast VPN prefixes are exchanged via Multiprotocol IBGP. Each PE has one single BGP session with the P router, which acts as a Route Reflector:

```
user@PE1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0           12         12          0          0          0          0
Peer                   AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.101.5.5            65000         70         58       0       1       24:00 Establ
  bgp.l3vpn.0: 12/12/12/0
  black.inet.0: 6/6/6/0
  white.inet.0: 6/6/6/0
```

There are six different BGP routes imported at each VRF, two from each remote PE. Every PE advertises both a direct and a static route (PE-CE link and CE-host link) for each VRF:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5

black.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
* 10.1.1.0/30             Self                          100        I
* 10.11.1.0/30            Self                          100        I

white.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
* 10.2.1.0/30             Self                          100        I
* 10.22.1.0/30            Self                          100        I
```

Follow the steps described in the Appendix, including all the end-to-end connectivity tests. It's important to have unicast working perfectly before moving on to multicast.

Configuring C-Multicast Islands

In this section, you will be configuring C-Multicast sources and receivers, and establishing C-PIM adjacencies between CEs and their local PEs. The integration of multicast into the existing VPN service will be done later. C-PIM signaling and C-Multicast traffic stays within the PE-CE local islands and does not flow end-to-end from C-Sources to C-Receivers yet.

Starting C-Multicast Traffic

Here you will need either a hardware-based traffic generator or a software program capable of generating IP Multicast traffic from a host (PC, workstation, or server). This multicast source is connected at CE1 ge-0/0/1, and should be capable of sending VLAN-tagged packets. If this is an issue, you can alternatively connect the sources to the CE using two different physical interfaces.

At the traffic generator, define the following flows:

- Source IP = 10.11.1.1. Destination IP = 239.1.1.1. Source MAC = Unicast MAC address. Destination MAC = 01:00:5e:01:01:01. VLAN ID = 101. Rate = 100 pps.
- Source IP = 10.11.1.1. Destination IP = 239.11.11.11. Source MAC = Unicast MAC address. Destination MAC = 01:00:5e:0b:0b:0b. VLAN ID = 101. Rate = 100 pps.
- Source IP = 10.22.1.1. Destination IP = 239.2.2.2. Source MAC = Unicast MAC address. Destination MAC = 01:00:5e:02:02:02. VLAN ID = 102. Rate = 100 pps.
- Source IP = 10.22.1.1. Destination IP = 239.22.22.22. Source MAC = Unicast MAC address. Destination MAC = 01:00:5e:16:16:16. VLAN ID = 102. Rate = 100 pps.

CAUTION The choice of source and destination MAC addresses is critical for the flows to be valid.

The destination MAC address must correspond to the destination IP address (or C-Group address). RFC 1112 states that an IP host group address is mapped to an Ethernet multicast address by placing the low-order 23-bits of the IP address into the low-order 23 bits of the Ethernet multicast address 01-00-5E-00-00-00 (hex).

Also, the source MAC addresses of the defined flows must be of type unicast. How can you differentiate a multicast MAC address from an unicast MAC address? By looking at the last bit of the first octet. Unicast MAC addresses have this bit set to 0. In other words, if the first octet of the MAC address is even, then it is unicast. It is worth mentioning that IPv4 Multicast is just one of the possible multicast services that can run over Ethernet. For example, IS-IS hellos over Ethernet have a multicast destination MAC address 01:80:c2:00:00:15.

Once the flows start, you can check the incoming packet rate at CE1. There should be around 200pps received at each logical interface:

```
user@CE1> show interfaces ge-0/0/1.1 statistics detail | match pps
     Input  packets:              7817833                  199 pps
     Output packets:                    0                    0 pps
```

```
user@CE1> show interfaces ge-0/0/1.2 statistics detail | match pps
     Input  packets:              7762586                  199 pps
     Output packets:                    0                    0 pps
```

TIP If you do not have an external multicast source, you can ping a multicast IP address from a router. The method is simple but not straightforward and will soon be the topic of a *Day One* book: *The Power of Ping - A Toolbox for Curious Testers and Troubleshooters*, to be published in 2012-13.

Configuring C-PIM Protocol

Once the BGP/MPLS VPN service is up and running, it's time to enable PIM at all the routing-instances in the CEs and the PEs. Execute the following commands at all PEs and CEs:

```
user@PE1> configure
user@PE1# set routing-instances black protocols pim interface all mode sparse
user@PE1# set routing-instances black routing-options multicast ssm-groups 239/14
user@PE1# set routing-instances white protocols pim interface all mode sparse
user@PE1# set routing-instances white routing-options multicast ssm-groups 239/8
user@PE1# commit and-quit
```

WARNING The `ssm-groups` ranges specified for each VRF are not the same.

In VRF white, the range 239/8 includes both 239.2.2.2 and 239.22.22.22, which get defined as SSM groups. On the other hand, in VRF black, the range 239/14 includes 239.1.1.1, but not 239.11.11.11. The latter is handled in the context of C-PIM ASM, and is discussed in Chapter 3. For the moment, you are working with 239.1.1.1 and 239.2.2.2 only, both configured as SSM groups.

Each CE is a C-PIM neighbor to its local PE. Execute the following commands at all PEs and CEs:

```
user@CE1> show pim neighbors instance black
Instance: PIM.black
B = Bidirectional Capable, G = Generation Identifier,
H = Hello Option Holdtime, L = Hello Option LAN Prune Delay,
P = Hello Option DR Priority

Interface           IP V Mode        Option      Uptime Neighbor addr
ge-0/0/2.1           4 2             HPLG      00:05:17 10.1.1.1

user@CE1> show pim neighbors instance white
Instance: PIM.black
B = Bidirectional Capable, G = Generation Identifier,
H = Hello Option Holdtime, L = Hello Option LAN Prune Delay,
P = Hello Option DR Priority

Interface           IP V Mode        Option      Uptime Neighbor addr
ge-0/0/2.2           4 2             HPLG      00:05:17 10.2.1.1
```

The key point you are seeing here is that there are no PIM adjacencies between PE and P-routers.

Starting C-Multicast Receivers

C-Multicast receivers are connected to CE3 and CE4. They can be configured to run IGMP version 3 and dynamically subscribe to the (C-S, C-G) flows. The default IGMP version is 2, so this needs to be changed. Execute the following commands at CE3 and CE4:

```
user@CE3> configure
user@CE3# edit protocols igmp
user@CE3# set interface ge-0/0/1.1 version 3
user@CE3# set interface ge-0/0/1.2 version 3
user@CE3# commit and-quit
```

If the receiving application does not support IGMP version 3, there are several alternatives. The simplest one is to configure static IGMP subscriptions directly at the CE interfaces. Execute the following commands at CE3 and CE4:

```
user@CE3> configure
user@CE3# edit protocols igmp
user@CE3# set interface ge-0/0/1.1 static group 239.1.1.1 source 10.11.1.1
user@CE3# set interface ge-0/0/1.2 static group 239.2.2.2 source 10.22.1.1
user@CE3# commit and-quit
```

NOTE Groups 239.11.11.11 and 239.22.22.22 are not configured at this stage yet. As discussed before, they will be used in later chapters.

Once the IGMP subscriptions are in place, both CE3 and CE4 send C-Join packets upstream to the neighboring PE, their unicast next-hop to reach the C-Sources. Execute the following commands at CE3 and CE4:

```
user@CE3> show pim join instance black
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.1.1.1
    Source: 10.11.1.1
    Flags: sparse,spt
    Upstream interface: ge-0/0/2.1

user@CE3> show pim join instance white
Instance: PIM.white Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.2.2.2
    Source: 10.22.1.1
    Flags: sparse,spt
    Upstream interface: ge-0/0/2.2
```

However, since C-Join state is not propagated across the backbone, CE1 does not forward the multicast traffic downstream. If you check the CE1 multicast routing table, it should not display any downstream interface for the C-Multicast flows:

```
user@CE1> show multicast route instance black group 239.1.1.1
Family: INET

Group: 239.1.1.1
    Source: 10.11.1.1/32
    Upstream interface: ge-0/0/1.1

user@CE1> show multicast route instance white group 239.2.2.2
Family: INET

Group: 239.2.2.2
    Source: 10.22.1.1/32
    Upstream interface: ge-0/0/1.2
```

In summary, there is one site (PE1-CE1) with active multicast sources, and two sites (PE3-CE3, PE4-CE4) with active receivers, but there is no signaling end to end. This prevents multicast traffic from reaching the receivers. You can see the current scenario in Figure 2.3 (PE4-CE4 omitted).

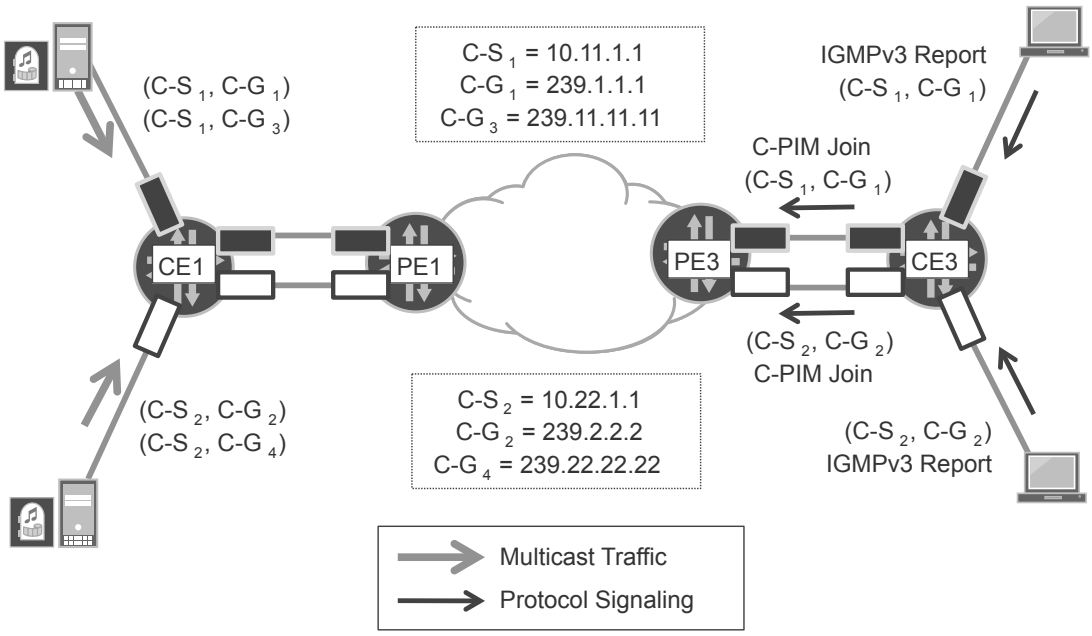


Figure 2.3 Traffic and Signaling in Baseline C-Multicast Scenario

Stopping C-Multicast Receivers

The next tasks are easier to follow if you temporarily disable the host-facing interfaces connected to the receivers. In this way, you can see MVPN Site Auto-Discovery and C-Multicast signaling as two separate steps. Everything would work fine if you decide not to disable, but let's temporarily stop the receivers. Execute the following commands on CE3 and CE4:

```
user@CE3> configure
user@CE3# set interface ge-0/0/1 disable
user@CE3# commit
```

Multicast VPN Site Auto-Discovery

In order to connect the C-PIM islands, it is necessary to integrate each VRF into a MVPN instance, and this in turn enables signaling C-Multicast state between PEs.

Enabling Multicast VPN Address Family

The first step is to let the PEs discover each other as members of black and white MVPN.

As you know from Chapter 1, MVPN routes are advertised via Multiprotocol BGP using a separate address family, so let's execute the following command at all PEs:

```
user@PE1> configure
user@PE1# set protocols bgp group RR family inet-mvpn signaling
user@PE1# commit and-quit
```

Execute the following command at the P-router:

```
user@P> configure
user@P# set protocols bgp group RR-CLIENTS family inet-mvpn signaling
user@P# commit and-quit
```

Once the `inet-mvpn` family is successfully negotiated, a new table called `bgp.mvpn.0` (empty for the moment) is listed in the command output shown below:

```
user@PE1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0           12         12          0          0          0          0
bgp.mvpn.0             0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.101.5.5            65000          8          5       0       0           1 Establ
  bgp.l3vpn.0: 12/12/12/0
  black.inet.0: 6/6/6/0
  white.inet.0: 6/6/6/0
  bgp.mvpn.0: 0/0/0/0
```

Full Mesh Multicast VPN at VRF black

The VRF black key configuration statement is `vrf-target target:65000:111`, as opposed to VRF white which has explicit `vrf-import` and `vrf-export` policies. In the context of unicast IPv4 routes, choosing `vrf-target` has two implications. First, every locally learned (in this case, direct *and* static) route at the VRF is exported to BGP with the specified route target (RT). Also, every received inet-vpn BGP route with that RT value is imported into VRF black. This has the advantage of a simpler configuration, and the drawback of less flexibility in selecting and modifying the exported/imported routes. It also implies that the VPN is full mesh and all the PEs get routes from each other, so complex configurations like hub-and-spoke, or extranet, are not feasible. If any of these features are required, then it is necessary to use `vrf-import` and `vrf-export` instead. Here is the complete VRF black configuration at PE1:

```
routing-instances {
   black {
      instance-type vrf;
      interface ge-0/0/2.1;
      route-distinguisher 65000:100;
      vrf-target target:65000:111;
      vrf-table-label;
      routing-options {
         static {
            route 10.11.1.0/30 next-hop 10.1.1.2;
         }
      }
   }
}
```

Moving to multicast, enabling MVPN with `vrf-target` is straightforward – add the following configuration to all PEs:

```
user@PE1> configure
user@PE1# set routing-instances black protocols mvpn
user@PE1# commit and-quit
```

Check that each local PE receives three routes, one per remote PE. These routes are installed in the `bgp.mvpn.0` and `black.mvpn.0` tables:

```
user@PE1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0           12         12          0          0          0          0
bgp.mvpn.0             3          3          0          0          0          0
Peer                    AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.101.5.5             65000         26         17       0       0        4:21 Establ
  bgp.l3vpn.0: 12/12/12/0
  white.inet.0: 6/6/6/0
  bgp.mvpn.0: 3/3/3/0
  black.mvpn.0: 3/3/3/0
  black.inet.0: 6/6/6/0
```

NOTE If `bgp.mvpn.0` stores inet-mvpn routes received for all the VRFs, then each VRF has a separate `<vrf-name>.mvpn.0` table to store the routes with a matching route target. Similarly, inet-vpn unicast routes are stored in `bgp.l3vpn.0` before being imported in the specific `<vrf-name>.inet.0` table. You can think of `bgp.mvpn.0` as the equivalent to `bgp.l3vpn.0` in the multicast world.

You can see the exchange of Type 1 Intra-AS I-PMSI Auto-Discovery routes in Figure 2.4 (Inter-AS scenarios are not covered in this book, so the Intra-AS term will be omitted for brevity). I-PMSI stands for *Inclusive PMSI* and the concept is explained in Chapter 1. The term Auto-Discovery is often shortened as just *A-D*.

The I-PMSI A-D routes perform a double function (see Figure 2.4):

- Site Auto-Discovery: The auto-discovery function is mandatory because it allows the PEs to discover each other's multicast VPN membership. Upon exchange of I-PMSI A-D routes, PE1, PE2, PE3 and PE4 become neighbors in MVPN black.
- Inclusive Tree Auto-Discovery: Later in this chapter, you will configure PE1 as the root of an Inclusive Tree for MVPN black. And at that time, PE1 adds an optional attribute called PMSI to the I-PMSI A-D route. This attribute carries information about the P-Tunnel type and identifier. PE1 considers all its neighbors in VPN black (PE2, PE3 and PE4) as leaves of the Inclusive Tree locally rooted at PE1. The tree itself is signaled with an independent mechanism, for example using RSVP-TE.

Examine the one inet-mvpn route advertised by each PE by executing the following commands at all PEs:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5 table black.mvpn

black.mvpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
   Prefix                  Nexthop        MED     Lclpref    AS path
*  1:65000:100:10.101.1.1/240  Self                    100        I

user@PE1> show route advertising-protocol bgp 10.101.5.5 table black.
mvpn extensive | match communities
     Communities: target:65000:111
```

NOTE You can safely ignore the network mask of these routes. It only expresses the number of significant bits of the prefix, which has a fixed length anyway (there is no subnetting here).

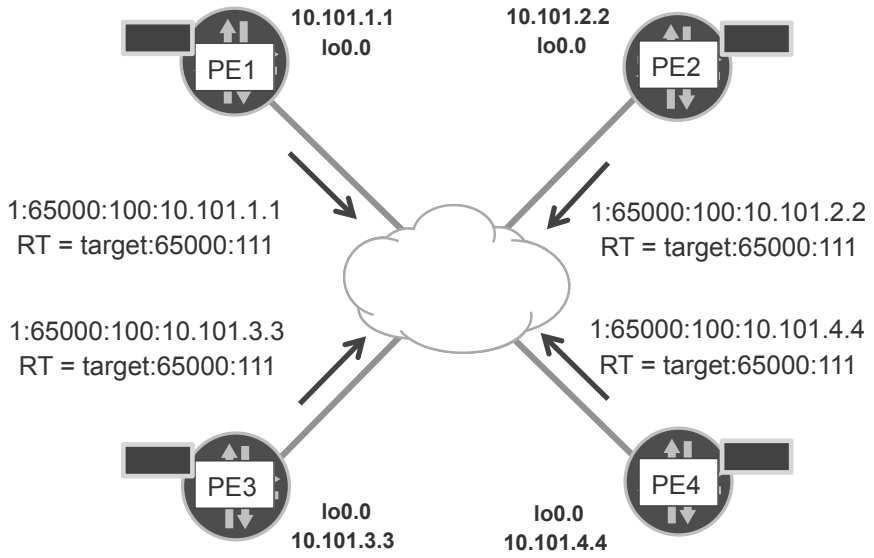


Figure 2.4 Type 1 I-PMSI Auto-Discovery Route Generation for MVPN black

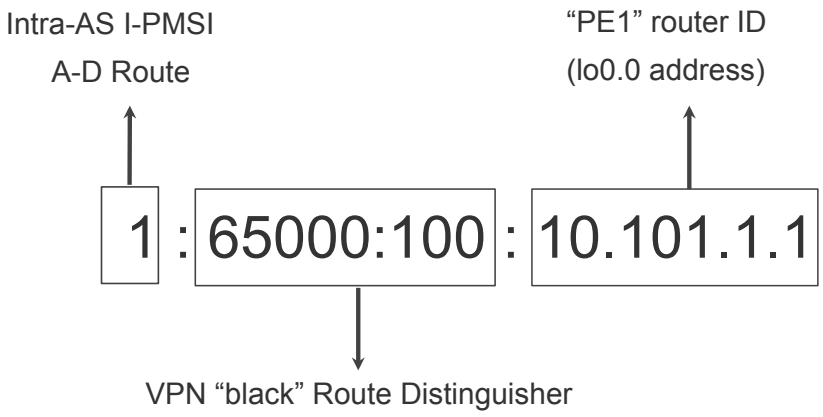


Figure 2.5 Format of a Type 1 Intra-AS Auto-Discovery Route

Examine the three inet-mvpn routes installed from remote PEs. These routes are received via the Route Reflector P. Execute the following commands at all PEs:

```
user@PE1> show route receive-protocol bgp 10.101.5.5 table black.mvpn

black.mvpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
   Prefix                  Nexthop        MED     Lclpref    AS path
*  1:65000:100:10.101.2.2/240  10.101.2.2              100        I
*  1:65000:100:10.101.3.3/240  10.101.3.3              100        I
*  1:65000:100:10.101.4.4/240  10.101.4.4              100        I

user@PE1> show route receive-protocol bgp 10.101.5.5 table black.
mvpn extensive | match communities
     Communities: target:65000:111
     Communities: target:65000:111
     Communities: target:65000:111
```

None of the PEs is a Sender PE for the moment, so there is no P-Tunnel being signaled yet. You will configure this later in this chapter.

WARNING When the `show route` command is applied to inet-mvpn routes it often displays a resolved next hop with push label operations. You should ignore the information since it has nothing to do with the labels used in the P-Tunnel. It comes from BGP Next Hop resolution. The BGP Next Hop is a mandatory attribute but is often (somehow) irrelevant for MVPN routes.

Partial Mesh Multicast VPN at VRF white

VRF white has explicit `vrf-import` and `vrf-export` policies. The initially configured policies have terms stating `from family inet` which apply only to unicast prefixes:

```
routing-instances {
    white {
        instance-type vrf;
        interface ge-0/0/2.2;
        route-distinguisher 65000:200;
        vrf-import white-imp;
        vrf-export white-exp;
        vrf-table-label;
        routing-options {
            static {
                route 10.22.1.0/30 next-hop 10.2.1.2;
            }
        }
    }
}
policy-options {
    policy-statement white-exp {
        term unicast {
            from family inet;
            then {
                community add white-target;
                accept;
            }
        }
    }
    policy-statement white-imp {
        term unicast {
            from {
                family inet;
                community white-target;
            }
            then accept;
        }
    }
    community white-target members target:65000:222;
}
```

In this example, the unicast part of the `vrf-import` and `vrf-export` policies do not perform any prefix selection or change. They just match and set the route target.

So, why not just use `vrf-target`? Let's assume that as an administrator of VPN white, you have been told that multicast sources are currently connected to CE1 and will be connected to CE2 in the near future, while receivers are local to CE3 and CE4. The multicast service requires connectivity between senders and receivers, but there is no need for two sites of the same type – two senders, or two receivers – to talk to each other in the SSM model (in Chapter 3 you will see how this differs in ASM). You decide to split the Multicast VPN white into two site groups: sender sites at PE1 and PE2, and receiver sites at PE3 and PE4. In this way, the P-Tunnel rooted

at PE1 will not have PE2 as destination, and vice versa.

This is a unique feature of BGP Multicast VPN, allowing you to save resources at the control and forwarding planes. In order to implement it, configure different route targets for each type of site, and modify policies accordingly.

NOTE Unicast topology for VPN white remains full mesh. The separation between sender and receiver sites only affects inet-mvpn prefixes. In fact, the route targets used for each address family are deliberately chosen to be different: multicast (target:65000:2 and target:65000:22) and unicast (target:65000:222).

Configure the following at PE1 and PE2:

```
user@PE1> configure
user@PE1# edit policy-options
user@PE1# set community white-sender-mvpn members target:65000:22
user@PE1# set policy-statement white-exp term mvpn from family inet-mvpn
user@PE1# set policy-statement white-exp term mvpn then community add white-sender-mvpn
user@PE1# set policy-statement white-exp term mvpn then accept
user@PE1# set community white-receiver-mvpn members target:65000:2
user@PE1# set policy-statement white-imp term mvpn from family inet-mvpn
user@PE1# set policy-statement white-imp term mvpn from community white-receiver-mvpn
user@PE1# set policy-statement white-imp term mvpn then accept
user@PE1# top
user@PE1# set routing-instances white protocols mvpn
user@PE1# commit and-quit
```

And configure the following at PE3 and PE4:

```
user@PE3> configure
user@PE3# edit policy-options
user@PE3# set community white-receiver-mvpn members target:65000:2
user@PE3# set policy-statement white-exp term mvpn from family inet-mvpn
user@PE3# set policy-statement white-exp term mvpn then community add white-receiver-mvpn
user@PE3# set policy-statement white-exp term mvpn then accept
user@PE3# set community white-sender-mvpn members target:65000:22
user@PE3# set policy-statement white-imp term mvpn from family inet-mvpn
user@PE3# set policy-statement white-imp term mvpn from community white-sender-mvpn
user@PE3# set policy-statement white-imp term mvpn then accept
user@PE3# top
user@PE3# set routing-instances white protocols mvpn
user@PE3# commit and-quit
```

Figure 2.6 illustrates the different RTs added to the I-PMSI Auto-Discovery routes advertised by Sender and Receiver PEs.

Examine the one inet-mvpn route advertised by each sender site by executing the following commands at PE1 and PE2:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5 table white.mvpn

white.mvpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
   Prefix                  Nexthop        MED     Lclpref    AS path
*  1:65000:200:10.101.1.1/240  Self                    100        I

user@PE1> show route advertising-protocol bgp 10.101.5.5 table white.
mvpn extensive | match communities
     Communities: target:65000:22
```

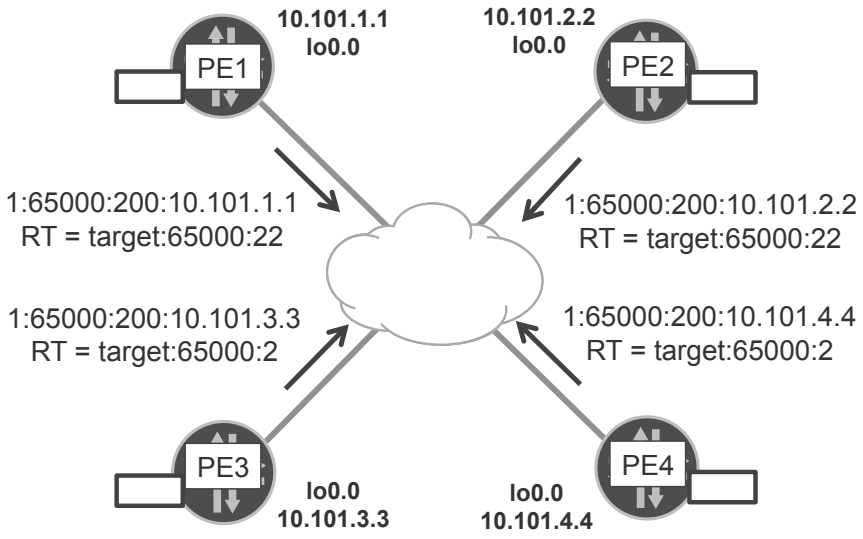


Figure 2.6 Type 1 Auto-Discovery Route Generation for Multicast VPN white

Now examine the one inet-mvpn route advertised by each receiver site by executing the following commands at PE3 and PE4:

```
user@PE3> show route advertising-protocol bgp 10.101.5.5 table white.mvpn

white.mvpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
   Prefix                      Nexthop        MED     Lclpref    AS path
*  1:65000:200:10.103.1.1/240  Self                    100        I

user@PE3> show route advertising-protocol bgp 10.101.5.5 table white.
mvpn extensive | match communities
     Communities: target:65000:2
```

Examine the two remote inet-mvpn routes received by each sender site by executing the following commands at PE1 and PE2:

```
user@PE1> show route receive-protocol bgp 10.101.5.5 table white.mvpn

white.mvpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
   Prefix                      Nexthop        MED     Lclpref    AS path
*  1:65000:200:10.101.3.3/240  10.101.3.3              100        I
*  1:65000:200:10.101.4.4/240  10.101.4.4              100        I

user@PE1> show route receive-protocol bgp 10.101.5.5 table white.
mvpn extensive | match communities
     Communities: target:65000:2
     Communities: target:65000:2
```

Now examine the two remote inet-mvpn routes received by each receiver site by executing the following commands at PE3 and PE4:

```
user@PE3> show route receive-protocol bgp 10.101.5.5 table white.mvpn

white.mvpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
   Prefix                      Nexthop        MED     Lclpref    AS path
*  1:65000:200:10.101.1.1/240  10.101.1.1              100        I
*  1:65000:200:10.101.2.2/240  10.101.2.2              100        I
```

```
user@PE3> show route receive-protocol bgp 10.101.5.5 table white.
mvpn extensive | match communities
     Communities: target:65000:22
     Communities: target:65000:22
```

Finally, to display the resulting Multicast VPN neighbors discovered for each VRF, execute the following command at *all* PEs:

```
user@PE1> show mvpn neighbor
[...]
Instance : black
  MVPN Mode : SPT-ONLY
  Neighbor                                I-P-tnl
  10.101.2.2
  10.101.3.3
  10.101.4.4
[...]
Instance : white
  MVPN Mode : SPT-ONLY
  Neighbor                                I-P-tnl
  10.101.3.3
  10.101.4.4
```

Signaling Inclusive Provider Tunnels

Although the PEs have discovered the other members of each MVPN, there is no P-Tunnel signaled to transport C-Multicast traffic over the core. Yet. You are about to configure per-VRF Inclusive P-Tunnels, signaled from the root to all the discovered MVPN neighbors. The chosen technology is RSVP Point-to-Multipoint (P2MP) LSPs.

Since the active C-Sources are local to CE1 for both VPNs black and white, it makes sense to configure the P-Tunnels rooted at PE1.

NOTE You could configure other PEs (PE2, PE3, PE4) as P-Tunnel roots as well. It's only needed if you plan to connect C-Sources to these sites in the future. But let's leave this optional task for the end of the book when you'll be more ready to see more routes and sessions.

Configure the following only at PE1:

```
user@PE1> configure
user@PE1# edit routing-instances black
user@PE1# set provider-tunnel rsvp-te label-switched-path-template default-template
user@PE1# top
user@PE1# edit routing-instances white
user@PE1# set provider-tunnel rsvp-te label-switched-path-template default-template
user@PE1# commit and-quit
```

One P2MP LSP is basically a set of point-to-point sub-LSPs. Each sub-LSP is a separate RSVP session signaled end-to-end. The transit routers need a way to know whether two sub-LSPs belong to the same parent P2MP LSP or not in order to build the point-to-multipoint forwarding state correctly. This is done with a special RSVP object called *P2MP LSP SESSION*, which has the same value for all the sub-LSPs of a given P2MP LSP.

MORE? Read RFC 4875 for details on RSVP-TE extensions for P2MP LSPs at https://datatracker.ietf.org/.

There are two P2MP LSPs signaled from PE1 (the root of the P-Tunnels), one for each VPN. Execute the following command at PE1:

```
user@PE1> show rsvp session p2mp
Ingress RSVP: 5 sessions
P2MP name: 65000:100:mvpn:black, P2MP branch count: 3
To              From            State   Rt Style Labelin Labelout LSPname
10.101.2.2      10.101.1.1      Up       0  1 SE       -   303376 10.101.2.2:65000:100:mvpn:black
10.101.3.3      10.101.1.1      Up       0  1 SE       -   303376 10.101.3.3:65000:100:mvpn:black
10.101.4.4      10.101.1.1      Up       0  1 SE       -   303376 10.101.4.4:65000:100:mvpn:black
P2MP name: 65000:200:mvpn:white, P2MP branch count: 2
To              From            State   Rt Style Labelin Labelout LSPname
10.101.3.3      10.101.1.1      Up       0  1 SE       -   303488 10.101.3.3:65000:200:mvpn:white
10.101.4.4      10.101.1.1      Up       0  1 SE       -   303488 10.101.4.4:65000:200:mvpn:white
Total 5 displayed, Up 5, Down 0

Egress RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0

Transit RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

As you can see, the `show rsvp session p2mp` command lists all the sub-LSPs for each P2MP LSP. They should all be Up, but can you figure out why `65000:100:mvpn:black` has 3 sub-LSPs, whereas `65000:200:mvpn:white` only has 2 sub-LSPs? The answer is that PE1 is not aware of PE2 membership to MVPN white due to the route target policies configured. It's intentional, and the goal is to save bandwidth and control plane resources. In a lab with just four PEs, the benefit may not be that self-evident, but as the network scales to hundreds or thousands of PEs, this flexibility in the policy framework becomes a key advantage of BGP Multicast VPN.

MORE? You need to use the extensive option if you want to see the next hop for each sub-LSP: look for PATH sentto pattern. Experiment with the different options of this command (name, ingress, egress, transit, etc...), in order to get familiar with it.

Let's examine the P2MP LSPs from the perspective of the P router:

```
user@P> show rsvp session p2mp
Ingress RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0

Egress RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0

Transit RSVP: 5 sessions
P2MP name: 65000:100:mvpn:black, P2MP branch count: 3
To              From            State   Rt Style Labelin Labelout LSPname
10.101.2.2      10.101.1.1      Up       0  1 SE  303376       16 10.101.2.2:65000:100:mvpn:black
10.101.3.3      10.101.1.1      Up       0  1 SE  303376       16 10.101.3.3:65000:100:mvpn:black
10.101.4.4      10.101.1.1      Up       0  1 SE  303376       17 10.101.4.4:65000:100:mvpn:black
P2MP name: 65000:200:mvpn:white, P2MP branch count: 2
To              From            State   Rt Style Labelin Labelout LSPname
10.101.3.3      10.101.1.1      Up       0  1 SE  303488       17 10.101.3.3:65000:200:mvpn:white
10.101.4.4      10.101.1.1      Up       0  1 SE  303488       18 10.101.4.4:65000:200:mvpn:white
```

You can see that the P router allocates the same MPLS label value for all the sub-LSPs of the same P2MP LSP (in other words, sharing the same P2MP LSP SESSION object). This means that PE1 only has to send one labeled copy of each packet to P, and P takes care of the replication. Figure 2.7 shows the forwarding path in both of the P2MP LSPs rooted at PE1, based on the label values in the output above.

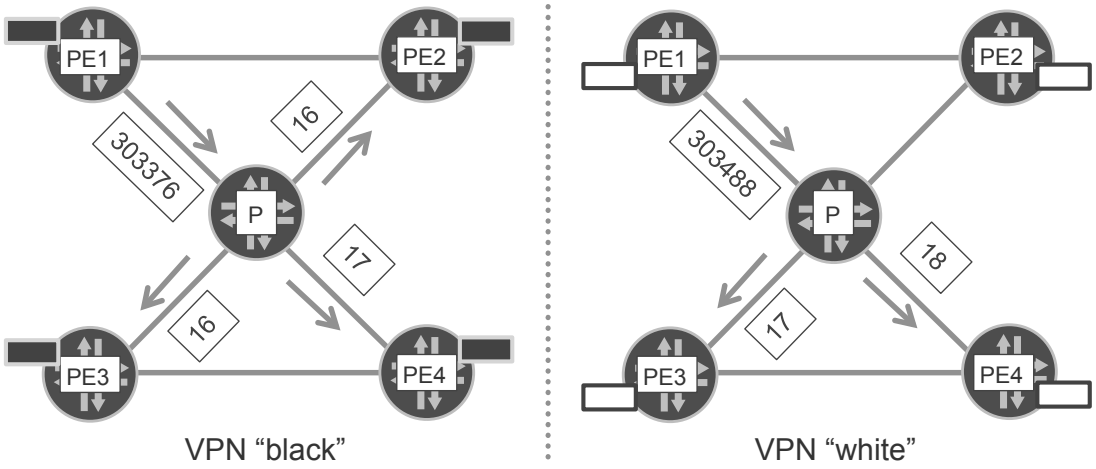


Figure 2.7 P2MP LSPs Rooted at PE1 for Multicast VPNs black and white

WARNING Do not expect the same label values in your setup as shown here because labels are assigned dynamically and in a non-deterministic way.

Now, let's examine the P2MP LSPs from the perspective of the egress PEs. Execute the following command at PE3, PE4 and PE2:

```
user@PE3> show rsvp session p2mp
Ingress RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0

Egress RSVP: 2 sessions
P2MP name: 65000:100:mvpn:black, P2MP branch count: 1
To           From          State   Rt Style Labelin Labelout LSPname
10.101.3.3     10.101.1.1     Up      0  1 SE     16       - 10.101.3.3:65000:100:mvpn:black
P2MP name: 65000:200:mvpn:white, P2MP branch count: 1
To           From          State   Rt Style Labelin Labelout LSPname
10.101.3.3     10.101.1.1     Up      0  1 SE     17       - 10.101.3.3:65000:200:mvpn:white
Total 2 displayed, Up 2, Down 0

Transit RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

What do the Labelin values at an egress PE represent? Execute the following commands at PE3, using the label values you got from the command above (not necessarily 16 and 17):

```
user@PE3> show route label 16

mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

16                 *[VPN/0] 08:15:12
                      to table black.inet.0, Pop

user@PE3> show route label 17

mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

17                 *[VPN/0] 08:16:15
                      to table white.inet.0, Pop
```

The PEs have `vrf-table-label` configured at the routing instances. In the book's test bed, PE3 assigns one MPLS label to each VRF: label 16 for black and 17 for white. Every MPLS packet arriving from the core to PE3 with label 16 or 17 is forwarded in the context of the black or white VRF, respectively. This applies to both types of traffic: unicast and multicast. The protocol used to advertise VPN labels is different in each case, though: BGP for unicast, and RSVP for multicast.

MORE? Imagine that P router is also an egress PE, with local VRFs and directly connected CEs. In that case, it would advertise to PE1 different MPLS labels for transit and egress branches of the same P2MP LSP. In VPN black at Figure 2.7, the three transit sub-LSPs (to PE2, PE3 and PE4) would have label 303376, whereas the egress sub-LSP (to P) would have the P-assigned black VRF label. This would result in traffic duplication at the PE1-P link, with no impact for end users but affecting the link bandwidth utilization. In order to improve this behavior, you can replace `vrf-table-label` by a vt- (virtual tunnel) unit assigned to the VRF; this unit just needs to have `family inet` configured. There are other advantages of using vt- interfaces, like MVPN extranet support. Modern hardware based on Trio chipset performs inline tunneling with no performance impact, so this option shall become a popular one.

If you remember, or if you look back in Figure 1.9, unicast traffic is transported using a label stacking method. The outer MPLS label is the transport label pushed by PE1 to tunnel unicast traffic towards the egress PE. The forwarding plane is identical, regardless of the protocol used to signal the transport label: RSVP as in Figure 1.9, or LDP as in our current unicast scenario. On the other hand, the inner (VPN) label is advertised by the egress PE as part of the BGP inet-vpn prefixes. Due to Penultimate Hop Popping (PHP) at the P router, the VPN label becomes the outer, and only, label when it reaches the Egress PE, which then selects the appropriate VRF to perform route resolution.

Back to multicast, our traffic is tunneled with just one MPLS label in the current scenario. Figure 2.8 shows the RSVP signaling involved. The last hop label is, in fact, the VPN label, so when the Egress PE (PE3) receives the packets it automatically maps them to the correct VRF.

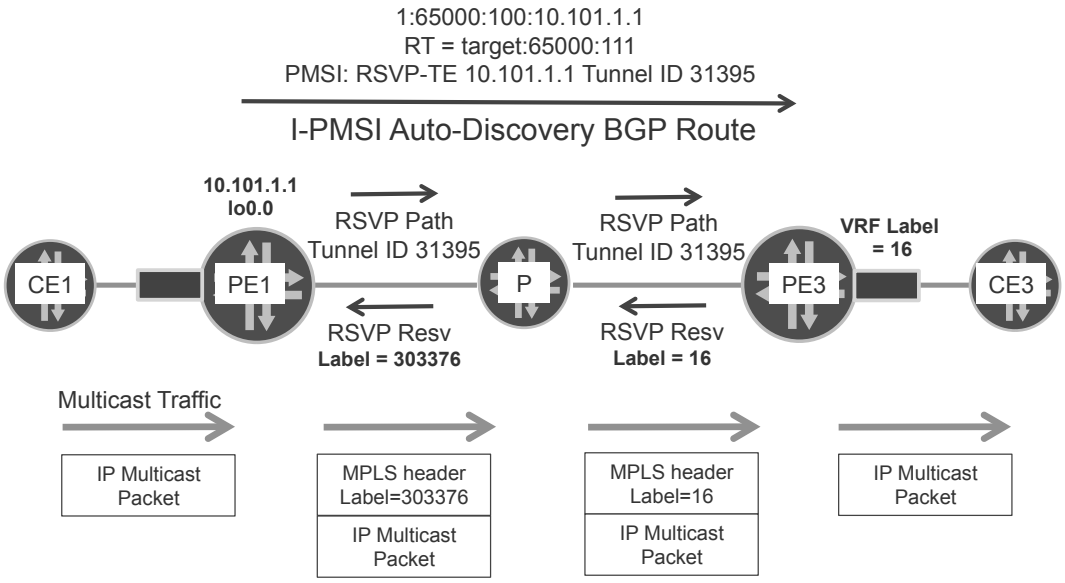


Figure 2.8 P2MP LSPs, Label Switching and VPN Labels

As you can see in Figure 2.8, PE3 receives a RSVP Path message for P2MP LSP `65000:100:mvpn:black`, and it answers with a RSVP Resv message whose label object matches the black VPN label, 16. The process means that PE3 somehow guessed that the P2MP LSP corresponds to MVPN black, otherwise it would have never signaled a label which is already bound to that VPN. This guessing process is fully deterministic, and here is how it works.

As soon as PE1 is configured with the `provider-tunnel` statement at VRF black and white, it becomes the root of two P-Tunnels. A new attribute called PMSI is added to the I-PMSI A-D routes sent by PE1 for VPNs black and white. Let's illustrate this by executing the following commands at PE1:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5 table black.mvpn extensive | match PMSI
     PMSI: Flags 0x0: Label[0:0:0]: RSVP-TE: Session_13[10.101.1.1:0:31395:10.101.1.1]

user@PE1> show route advertising-protocol bgp 10.101.5.5 table white.mvpn extensive | match PMSI
     PMSI: Flags 0x0: Label[0:0:0]: RSVP-TE: Session_13[10.101.1.1:0:31396:10.101.1.1]
```

You can see that the PSMI attrribute includes `Session_13[<unique_identifier>]`. The unique identifier includes the router ID of PE1 and a locally generated Tunnel ID, in this case: 31395 for black and 31396 for white. This same Tunnel ID is also part of the P2MP LSP SESSION object of the RSVP session, illustrated earlier in Figure 2.8.

Let's execute the following command at PE3:

```
user@PE3> show rsvp session p2mp extensive | match "p2mp|port"

P2MP name: 65000:100:mvpn:black, P2MP branch count: 1
  P2MP LSPname: 65000:100:mvpn:black
  Port number: sender 1 receiver 31395 protocol 0
P2MP name: 65000:200:mvpn:white, P2MP branch count: 1
  P2MP LSPname: 65000:200:mvpn:white
  Port number: sender 1 receiver 31396 protocol 0
```

Can you spot the Tunnel IDs 31395 and 31396 in the output?

This is the magic behind P2MP LSP to VRF mapping. First, PE1 sends an I-PMSI A-D BGP route. According to the route targets, this route is installed into VRF black at PE3 (table `black.mvpn.0`). Later, PE3 gets a RSVP Path message whose Tunnel ID matches the PMSI attribute of the I-PMSI A-D route originated by PE1. This closes the loop and maps the P2MP LSP to VRF black at PE3.

What if the I-PMSI A-D route is received after the RSVP Path message? PE3 temporarily assigns label 3 (implicit null) to the LSP, then upon reception of PE1's A-D route, sends a new RSVP Resv message with the VPN black label value.

There is one command that puts all this information together, here executed at PE1 and PE3:

```
user@PE1> show mvpn instance
[...]
Legend for provider tunnel
I-P-tnl -- inclusive provider tunnel S-P-tnl -- selective provider tunnel

Instance : black
  MVPN Mode : SPT-ONLY
  Provider tunnel: I-P-tnl:RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1
  Neighbor                                I-P-tnl
  10.101.2.2
  10.101.3.3
  10.101.4.4
```

```
Instance : white
  MVPN Mode : SPT-ONLY
  Provider tunnel: I-P-tnl:RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1
  Neighbor                              I-P-tnl
  10.101.3.3
  10.101.4.4

user@PE3> show mvpn instance
[...]
Legend for provider tunnel
I-P-tnl -- inclusive provider tunnel S-P-tnl -- selective provider tunnel

Instance : black
  MVPN Mode : SPT-ONLY
  Provider tunnel: I-P-tnl:invalid:
  Neighbor                              I-P-tnl
  10.101.1.1                            RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1
  10.101.2.2
  10.101.4.4

Instance : white
  MVPN Mode : SPT-ONLY
  Provider tunnel: I-P-tnl:invalid:
  Neighbor                              I-P-tnl
  10.101.1.1                            RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1
  10.101.2.2
```

The neighbor list should match the route target policies. The I-P-tnl listed (with explicit Tunnel IDs) corresponds to the PEs with `provider-tunnel` explicitly configured, in this case, PE1 only.

Try It Yourself: Traffic Engineering of P2MP LSP's

One of the benefits of using RSVP to signal P2MP LSPs is the possibility to use Traffic Engineering (TE) extensions. Instead of using the default template, define a custom template with specific TE characteristics. Choose a color (admin-group) for the PE1-P link, and make sure one of the P2MP LSP's rooted at PE1 avoids links with that color. The P2MP LSP should be signaled again and use the PE1-PE2 link, with higher IGP metric.

End-to-End C-Multicast Signaling and Traffic

Now that the Multicast VPNs are fully configured, it's time to start the C-Multicast receivers and let the traffic flow across different sites through the P2MP LSPs. To begin, let's execute the following commands at CE3 and CE4:

```
user@CE3> configure
user@CE3# delete interface ge-0/0/1 disable
user@CE3# commit
```

NOTE In order to keep this brief, only the output for VPN black is shown here. Please execute all these commands for VPN white as well.

Follow the C-Join from C-Receiver to C-Source

The CEs connected to multicast receivers are sending (C-S, C-G) PIM C-Join packets upstream. According to unicast route resolution, the next-hop towards the C-Source is the locally connected PE. Execute the following commands at CE3 and CE4:

```
user@CE3> show route 10.11.1.1 table black

black.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 01:07:08
                    > to 10.1.3.1 via ge-0/0/2.1

user@CE3> show pim join instance black
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.1.1.1
    Source: 10.11.1.1
    Flags: sparse,spt
    Upstream interface: ge-0/0/2.1
```

Once the local PEs receive a C-Join from their downstream CE, they also perform an unicast route resolution towards the C-Source. The result is an inet-vpn BGP route pointing to the MPLS core. Execute the following commands at PE3 and PE4:

```
user@PE3> show route 10.11.1.1 table black

black.inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.11.1.0/30       *[BGP/170] 00:27:43, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/2.0, Push 16, Push 301936(top)

user@PE3> show pim join instance black
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.1.1.1
    Source: 10.11.1.1
    Flags: sparse,spt
    Upstream protocol: BGP
    Upstream interface: Through BGP
```

NOTE The double label push is for the unicast route only. These labels are not used for the multicast forwarding you are currently looking at.

The C-Join state points to BGP as the upstream interface. Actually, there is no PIM neighbor relationship between the PEs. The downstream PE converts the C-PIM (C-S, C-G) state into a Type 7 - Source Tree Join BGP route, and sends it to the upstream PE en route to the C-Source. The format of the route is explained in Figure 2.9. Execute the following commands at PE3 and PE4:

```
user@PE3> show route advertising-protocol bgp 10.101.5.5 table black.mvpn extensive | find 7:

* 7:65000:100:65000:32:10.11.1.1:32:239.1.1.1/240 (1 entry, 1 announced)
 BGP group RR type Internal
     Route Distinguisher: 65000:100
     Nexthop: Self
     Flags: Nexthop Change
     Localpref: 100
     AS path: [65000] I
     Communities: target:10.101.1.1:5
```

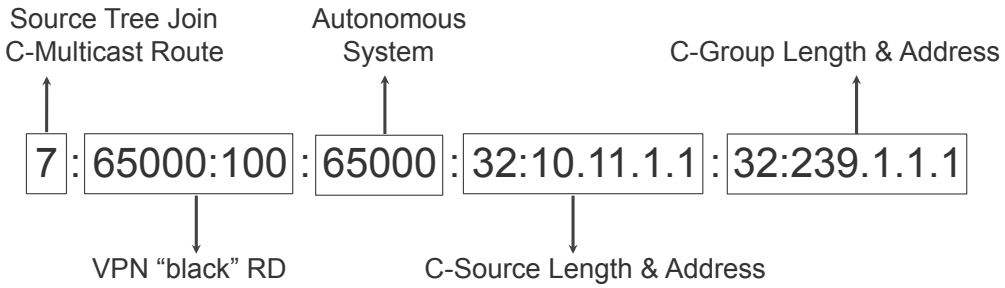


Figure 2.9 Format of a Type 7 C-Multicast – Source Tree Join Route

NOTE The 32 value shown in Figure 2.9 and the output is just the length of the address, 32 bits in IPv4. It is not a network mask.

The BGP Next Hop attribute and the MPLS labels resulting from its resolution can be safely ignored. The key attribute of the Source Tree Join route is Route Target `target:10.101.1.1:5`. Where does it come from? It does contain the Router ID of PE1, so that is a clue. Let's execute the following command at PE1:

```
user@PE1> show route advertising-
protocol bgp 10.101.5.5 extensive | match "routes|entry|communities"

black.inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
* 10.1.1.0/30 (1 entry, 1 announced)
     Communities: target:65000:111 src-as:65000:0 rt-import:10.101.1.1:5
* 10.11.1.0/30 (1 entry, 1 announced)
     Communities: target:65000:111 src-as:65000:0 rt-import:10.101.1.1:5

white.inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
* 10.2.1.0/30 (1 entry, 1 announced)
     Communities: target:65000:222 src-as:65000:0 rt-import:10.101.1.1:6
* 10.22.1.0/30 (1 entry, 1 announced)
     Communities: target:65000:222 src-as:65000:0 rt-import:10.101.1.1:6

black.mvpn.0: 5 destinations, 6 routes (5 active, 1 holddown, 0 hidden)
* 1:65000:100:10.101.1.1/240 (1 entry, 1 announced)
     Communities: target:65000:111

white.mvpn.0: 4 destinations, 5 routes (4 active, 1 holddown, 0 hidden)
* 1:65000:200:10.101.1.1/240 (1 entry, 1 announced)
    Communities: target:65000:22
```

As you can see in the output, the unicast VPN routes have three different communities: Route Target, Source AS, and Route Import. As you know, a Route Target has been explicitly configured for the unicast routes, via `vrf-target` (black) or `vrf-export` (white). Where do the other two communities come from? Well, Source AS and Route Import communities were automatically added when `protocols mvpn` at the VRFs were configured.

TRY THIS Issue the `show policy` command at PE1 to list the internal policies currently applied. Examine all the policies via `show policy <name>` and try to spot which one is adding the Route Import community to the inet-vpn routes.

A key property of the Route Import community is its unique value for each (PE,

VRF) combination. This is ensured by including the PE router ID in the community, together with a locally generated number that the advertising PE associates to each VRF. In the output just reviewed, `rt-import:10.101.1.1:5` and `rt-import:10.101.1.1:6` identify (PE1, black) and (PE1, white) respectively. The per-VRF number is local to each PE and auto-configured by the system, so there is no need for two different PEs to agree on its value.

Let's go back to the Source Tree Join route originated by PE3. Its Route Target `target:10.101.1.1:5` precisely matches the Route Import of the unicast route to the C-Source, as shown in Figure 2.10. By setting that Route Target value, PE3 makes sure that its advertised route is deterministically imported by PE1 in VRF black. So the Source Tree Join is a targeted operation that only involves two routers: PE1 and PE3.

NOTE This control plane reduction associated to BGP is one of its advantages over PIM. In draft-rosen, a C-PIM Join packet is processed by all the neighbors, not just by the Upstream Neighbor.

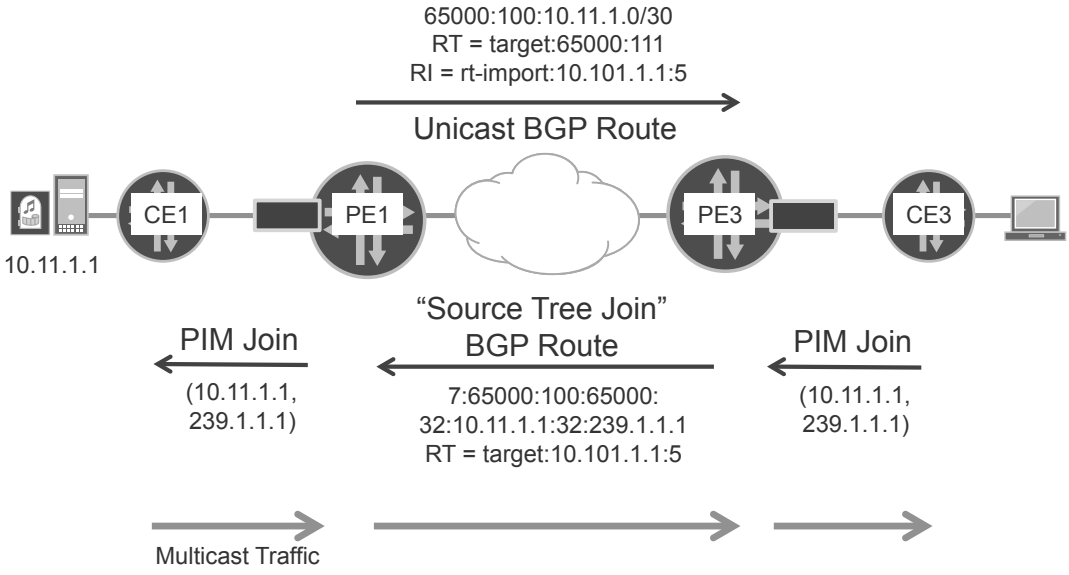


Figure 2.10 Interaction Between C-PIM Join and C-Multicast BGP Routes

PE3 and PE4 both send identical Source Tree Join routes to PE1 for VRF black. The Route Reflector P selects one of them (the best), and sends it to PE1. The choice is not important, as the relevant information (prefix, route target) contained in these routes is the same.

PE1 receives and installs the Source Tree Join route, which then triggers a PIM C-Join. Let's check PE1:

```
user@PE1> show route 10.11.1.1 table black
black.inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.11.1.0/30       *[Static/5] 10:01:22
                    > to 10.1.1.2 via ge-0/0/2.1
```

```
user@PE1> show pim join instance black
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.1.1.1
    Source: 10.11.1.1
    Flags: sparse,spt
    Upstream interface: ge-0/0/2.1
```

If a Source Tree Join BGP route is equivalent to a (C-S, C-G) PIM Join, what would be the equivalent of a (C-S, C-G) PIM Prune? Well, simply the withdrawal of a Source Tree Join route.

What if a source is multihomed to both PE1 and PE2? How do PE3 and PE4 decide where to target the Source Tree Join route? It is critical that both PE3 and PE4 make the same choice and elect a single forwarder among PE1 and PE2. Otherwise, both PE1 and PE2 would forward the C-Multicast traffic, bringing two copies of each packet to end receivers. There are a number of single forwarder election mechanisms described in RFC 6513. The default is to select the numerically highest IP address PE loopback, in this case PE2. Alternatively, you can simply make PE3/PE4 choose the best unicast route to the C-Source, using the standard BGP path selection algorithm. If you like the last approach, use the knob `set routing-instances black protocols mvpn unicast-umh-election`.

CAUTION With `unicast-umh-election`, IGP metric is the tie-breaker so PE3 chooses PE1 while PE4 chooses PE2, causing C-Multicast traffic duplication. In order to ensure that PE1 is elected as a single forwarder, make sure PE1 announces its unicast routes with a higher Local Preference.

Follow the C-Multicast Traffic Downstream

The C-Multicast traffic flow (10.11.1.1, 239.1.1.1) should be forwarded end-to-end now. Let's verify that CE1 is sending the traffic down to PE1:

```
user@CE1> show multicast route instance black
Family: INET

Group: 239.1.1.1
    Source: 10.11.1.1/32
    Upstream interface: ge-0/0/1.1
    Downstream interface list:
        ge-0/0/2.1

Group: 239.11.11.11
    Source: 10.11.1.1/32
    Upstream interface: ge-0/0/1.1

user@CE1> show multicast route instance black extensive | match pps
    Statistics: 18 kBps, 100 pps, 217769 packets
    Statistics: 18 kBps, 100 pps, 218289 packets
```

Now let's verify that PE1 is sending the C-Multicast packets into the black Inclusive Tree locally rooted at PE1:

```
user@PE1> show multicast route instance black
Family: INET

Group: 239.1.1.1
    Source: 10.11.1.1/32
    Upstream interface: ge-0/0/2.1
```

```
    Downstream interface list:
        so-0/1/0.0

user@PE1> show mvpn c-multicast instance-name black
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : SPT-ONLY
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.11.1.1/32:239.1.1.1/32     RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1      RM

user@PE1> show route table black.inet.1
black.inet.1: 1 destinations, 2 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

239.1.1.1,10.11.1.1/32*[MVPN/70] 00:13:41
                    > via so-0/1/0.0, Push 303376
                    [PIM/105] 00:13:41
                      Multicast (IPv4)
```

From all the information gathered, you can conclude that the C-Multicast flow (10.11.1.1, 239.1.1.1) is already integrated in MVPN black and sent by PE1 into the black Inclusive P-Tunnel. To be completely sure, you can execute a `show rsvp session statistics` command.

The table `<vrf-name>.inet.1` stores the multicast routes of a given VRF. Verify that the pushed label displayed (303376 in this example) corresponds to the Labelout of the RSVP session whose P2MP name is `65000:100:mvpn:black`.

PE3 and PE4 should be receiving C-Multicast traffic from the P2MP LSPs, and sending it to the downstream receivers, so execute the following commands at PE3, PE4 and PE2:

```
user@PE3> show multicast route instance black
Family: INET

Group: 239.1.1.1
    Source: 10.11.1.1/32
    Upstream interface: lsi.0
    Downstream interface list:
        ge-0/0/2.1

user@PE3> show mvpn c-multicast instance-name black
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : SPT-ONLY
  C-mcast IPv4 (S:G)            Ptnl                                           St
  10.11.1.1/32:239.1.1.1/32     RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1
```

The traffic is being received over a logical lsi (label switched interface), which is the receive-only interface associated to the VPN black label. PE3 and PE4 send the C-Multicast traffic down to their directly connected CEs, which are, in turn, connected to receivers.

TRY THIS Based on the `show mvpn c-multicast` information, could you explain what the RM flag means?

PE2 is receiving the C-Multicast traffic from the P2MP LSP mapped to black VPN. This is a characteristic of Inclusive Trees. One single Source Tree Join route installed

at PE1 is enough to bring the traffic down to all the neighbors of PE1 at MVPN black. This is fine for PE3 and PE4, which have downstream receivers, but not for PE2.

There are two complementary approaches to limit this undesired flooding. One is creating a partial-mesh topology using route target policies. PE1 and PE2 are not neighbors of each other at MVPN white, so PE2 does not receive the white C-Multicast traffic forwarded by PE1. This approach however does not cover the scenario where the receivers of a given (C-S, C-G) flow are all connected to CE3, and none of them to CE4. In this case, the Source Tree Join route sent by PE3 upstream makes PE1 send the traffic down to the Inclusive Tree, also reaching PE4, which in turn discards the traffic.

If this is an issue for you and your test bed in terms of bandwidth utilization, PE1 can send specific C-Multicast flows down to Selective Trees. These only deliver traffic to the PEs with downstream C-Receivers for the mapped flows. It's a topic that is fully explained in Chapter 4.

But for now let's finish this chapter, and verify that the C-Multicast traffic is being delivered to the receivers. Execute the following commands at CE3 and CE4:

```
user@CE3> show multicast route instance black
Family: INET

Group: 239.1.1.1
    Source: 10.11.1.1/32
    Upstream interface: ge-0/0/2.1
    Downstream interface list:
        ge-0/0/1.1

user@CE3> show multicast route instance black extensive | match pps
    Statistics: 18 kBps, 100 pps, 235769 packets
```

NOTE Perform all the relevant steps for VPN white as well, if you haven't already.

Answers to Try It Yourself Sections of Chapter 2

Try It Yourself: Traffic Engineering of P2MP LSPs

Does your configuration change look like the one below? If so, well done!

```
user@PE1# show | compare
[edit protocols mpls]
+   admin-groups {
+       blue 0;
+   }
[edit protocols mpls]
+    label-switched-path no-blue-p2mp {
+        template;
+        admin-group exclude blue;
+        p2mp;
+    }
[edit protocols mpls interface so-0/1/0.0]
+    admin-group blue;
[edit routing-instances mcast provider-tunnel rsvp-te label-switched-path-template]
-      default-template;
+      no-blue-p2mp;
```

Verify that the P2MP LSP is now using the PE1-PE2 link. You are invited to try other TE functionalities like link-protection or bandwidth reservation. Instead of using templates, you can also define a static LSP. Give it a try!

Chapter 3

BGP Multicast VPN with PIM ASM as PE-CE Protocol

The C-Multicast signaling described in Chapter 2 and shown in Figure 2.10 can be summarized as: C-PIM (S, G) Join state propagates seamlessly from C-Receiver to C-Source. When traversing the MPLS core, the C-PIM Join is disguised as a Source Tree Join BGP route, but overall the signaling process is quite direct.

If C-PIM ASM is used, the process becomes more complex, for exactly the same reasons as in plain IP Multicast. First off, the source and receiver PIM signaling meet at the customer Rendezvous Point (C-RP), and traffic starts flowing through the Shared Tree, also known as Rendezvous Point Tree (RPT). Later on, there is an optional switchover to the Source Tree, also known as Shortest Path Tree (SPT). In our BGP Multicast VPN solution, the inet-mvpn address family has all the required route types to seamlessly participate in this process.

NOTE The concepts of Shared Tree (RPT) and Source Tree (SPT) are completely unrelated to the notions of Inclusive and Selective Tree. In this chapter, there is no Selective Tree and both the RPT and the SPT span Inclusive Trees (rooted at different PEs). You can think of RPT/SPT as C-Multicast concepts, while Inclusive/Selective Trees are P-Tunnels, meaningful in the Provider context only. From the C-Multicast perspective, a P-Tunnel is just "one hop."

This chapter keeps all the infrastructure built in Chapter 2, with Inclusive P-Tunnels based on RSVP P2MP LSPs. All activities are performed at VRF black.

Deployment Options for C-PIM ASM

By default, a MVPN instance runs in SPT-only mode, where the PEs do not signal (*, C-G) join state upstream to other PEs. In other words, in SPT-only mode the Shared Tree never connects two different VPN sites. Junos OS version 10.0 introduced MVPN support for RPT-SPT mode, which allows PEs to signal both (*, C-G) and (C-S, C-G) to join state via BGP.

Out of the three existing options to integrate C-PIM ASM with BGP MVPN, two of them are compatible with SPT-only mode. The third one requires RPT-SPT.

Let's look back at IP Multicast basics for a moment. In the ASM model, the receiver indicates the groups it wants to join via IGMPv2, but it does not provide any information about the sources. The last hop router generates a (*, C-G) PIM Join upstream towards the customer Rendezvous Point (C-RP), which has the information about what C-Sources are active for that particular C-G.

The placement of the C-RP is very important. The multicast C-Receiver and the C-RP may be at the same VPN site or at different sites. If they are at the same site, the (*, C-G) PIM Joins are converted into (C-S, C-G) PIM Joins locally at the site. In this case, the BGP signaling is the same as the one explained in Chapter 2 for C-PIM SSM.

Things get more complex when the C-RP is in a remote site, as the Receiver PE needs to have at least one of the following capabilities:

- Knowledge about the active C-Sources: If a PE with downstream (*, C-G) receivers knows the active (C-S, C-G) flows, it can create (C-S, C-G) Source Tree Join routes. At this point, the SSM model explained in Chapter 2 applies. There are two solutions based on this approach: C-RP instantiated in a VRF, and, MSDP session between a C-RP and a PE. Both solutions work fine if the MVPN is running in SPT-only mode.
- Signaling (*, C-G) join state via BGP: If a PE has downstream (*, C-G) receivers, it may send a Shared Tree Join route to the upstream PE en route to the

C-RP. For this solution to work, the MVPN must run in RPT-SPT mode.

The examples used in this chapter to illustrate the different solutions rely on placing the C-RP in a different site from the C-Sources and the C-Receivers. That should provide a more complete learning experience as it is the most challenging scenario from the point of view of the network. Even though there is only one C-RP in the scenarios, it is perfectly possible to have several C-RPs for redundancy, meshed to each other according to the Anycast model.

The figures shown in this chapter only depict one receiver site (CE3, PE3), merely for simplicity. Since traffic is injected in an Inclusive Tree, and Multicast VPN black is fully meshed, all the sites receive the C-Multicast traffic requested by one single site. So, even though adding one more receiver site (CE4, PE4) results in additional C-Multicast BGP routes, data forwarding inside the P-Tunnel is unchanged. Note that this behavior can be modified with Selective P-Tunnels (see Chapter 4).

C-RP Instantiated in a VRF

This solution is compatible with SPT-only mode. The C-PIM instance running in VRF black at PE2 acts as a customer Rendezvous Point (C-RP) as illustrated in Figure 3.1. The first-hop router CE1 sends (C-S, C-G) Registers unicast to the C-RP, allowing the C-RP to learn about all the active (C-S, C-G) flows, so it is aware of every source and every group. This information is advertised to the other PEs using Type 5 Source Active Auto-Discovery BGP routes. Source Active A-D routes are functionally equivalent to MSDP Source Active messages exchanged between PEs. PE3 can then convert the local (*, C-G) PIM join state into a (C-S, C-G) Source Tree Join. There is no need to exchange (*, C-G) state via BGP. Since the C-PIM Register packet is unicast, it is forwarded as any other unicast data packet in the VPN.

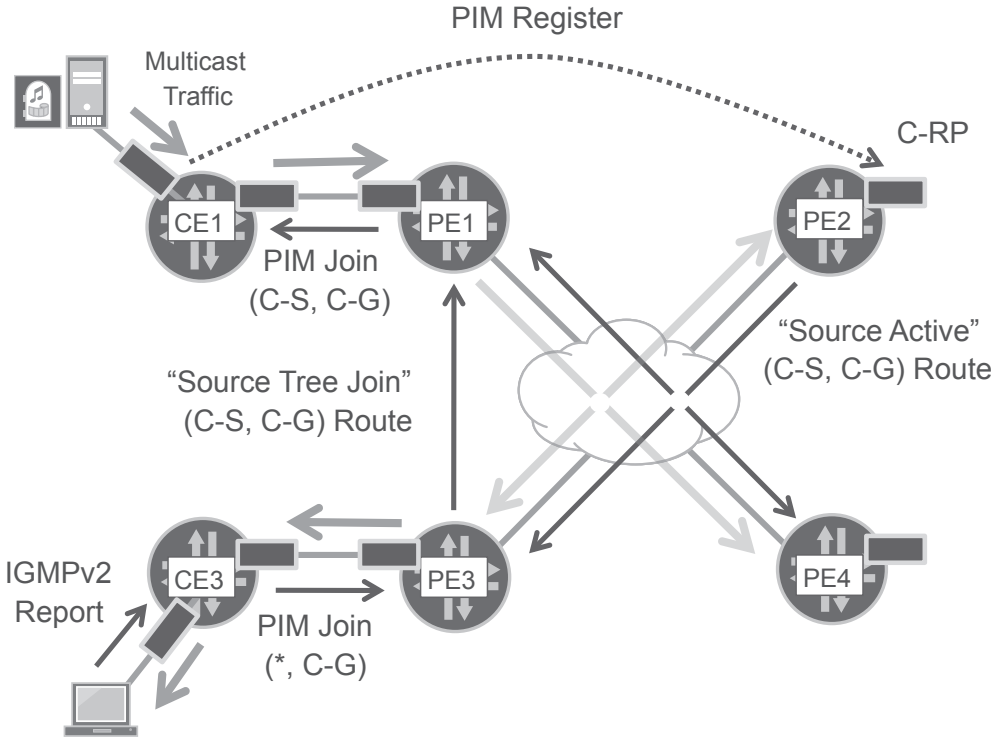


Figure 3.1 Multicast VPN with C-RP Instantiated in a VRF

In this model, only Shortest Path Trees span different sites or, said differently, the Shared Tree does not traverse the backbone. The service provider is not involved in C-Multicast SPT switchover signaling. On the other hand, unless you use custom PIM policies the service provider is aware of all the C-Multicast flows, even the site-local ones not requiring MVPN services at all.

An obvious implication of this model is that end customers need to outsource the C-RP function to the service provider. Hence they lose the control of their own C-RP, which can be an advantage or a disadvantage depending on the customer requirements.

MSDP Session Between a C-RP and a PE

This model is also compatible with SPT-only mode. In Figure 3.2, the C-RP (CE2) peers with VRF black at PE2 via MSDP. The MSDP session is used to communicate all the (C-S, C-G) information via Source Active messages. The rest of the signaling is identical to the previously discussed model (C-RP instantiated in a VRF).

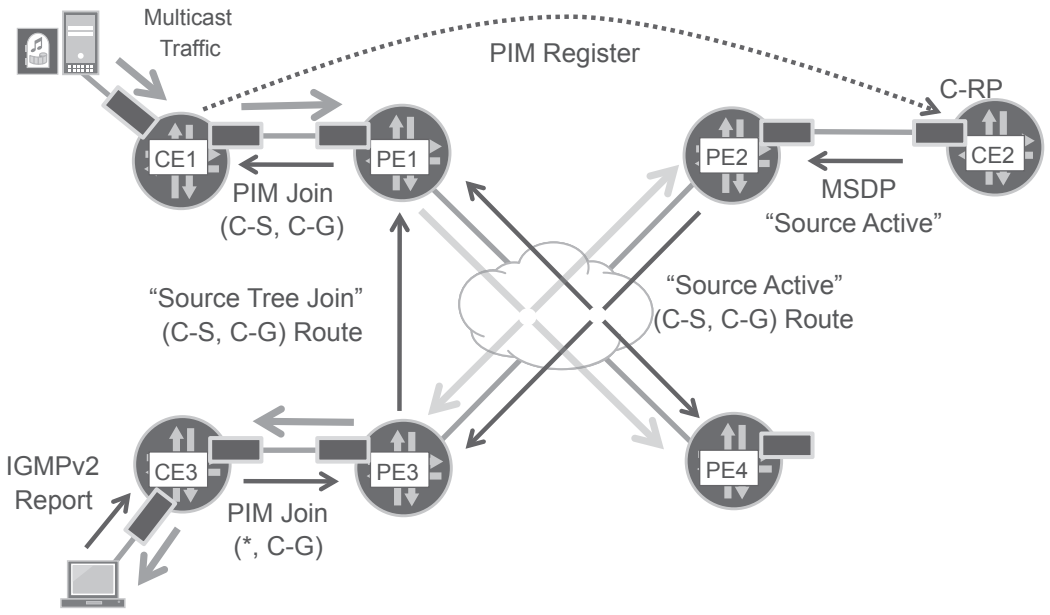


Figure 3.2 Multicast VPN with MSDP Session Between the C-RP and a VRF

One advantage of this model is that it allows the end customer to manage their C-RP but it also requires an extra control session that is responsible for the synchronization of active source knowledge between the C-RP and PE2. Apart from this, the advantages and disadvantages are very similar to the previously discussed model ("C-RP instantiated in a VRF"). With default PIM and MSDP policies, the service provider is aware of all the active sources, and the Shared Tree does not traverse the backbone.

DETAIL In current implementation, a MSDP Source Active message is automatically converted into a BGP Source Active A-D route. The reverse is not true though: there is no redistribution from BGP into MSDP.

Multicast VPN Running in RPT-SPT Mode

This is a more recently deployed feature allowing the end customers to perceive the VPN as a transparent extension of their multicast-enabled network. From the service provider perspective it is the most complex approach, since the PEs need to signal both Shared and Source Trees while playing a key role in the SPT switchover. Although end customers perceive RPT-SPT as a seamless solution, service providers usually prefer one of the SPT-only variants due to their simplified operation.

In RPT-SPT mode, PEs can exchange Type 6 C-Multicast - Shared Tree Join BGP routes, functionally equivalent to (*, C-G) PIM Joins. These routes allow the C-Multicast Shared Trees to span different MVPN sites.

NOTE The two ASM models based on SPT-only are very similar to SSM from the perspective of the PEs. In this chapter, you will configure MVPN black in RPT-SPT mode, which is a complex choice but it's also more complete for the purpose of this book.

When the C-RP is placed between the C-Source and the C-Receivers, signaling is simplified. If the C-RP is in a different site, things get more challenging. This is actually the case for plain PIM too. The steps in BGP Multicast VPN are similar to the ones in PIM, explained back in Figure 1.12: Rendezvous Point in a Stick.

In the following example, the C-RP role is performed by CE2, and site 4 (PE4, CE4) has been omitted for simplicity. Please refer to Figures 1.3, 1.4, and 1.5 to understand the whole process.

The full sequence of events starts here with the steps illustrated in Figure 3.3 (the details of the C-Register process are omitted for the sake of brevity):

1. The C-Receiver sends an IGMPv2 (*, C-G) Report.

2. CE3 sends a PIM (*, C-G) Join to PE3 at VRF black.

3. PE3 sends a (*, C-G) Shared Tree Join BGP route. This route is targeted to PE2, the PE en route to the C-RP.

4. PE2 at VRF black sends a PIM (*, C-G) Join to the C-RP (CE2).

5. As soon as the source C-S starts sending traffic to C-G, CE1 informs CE2 of the existence of (C-S, C-G) via C-Registers.

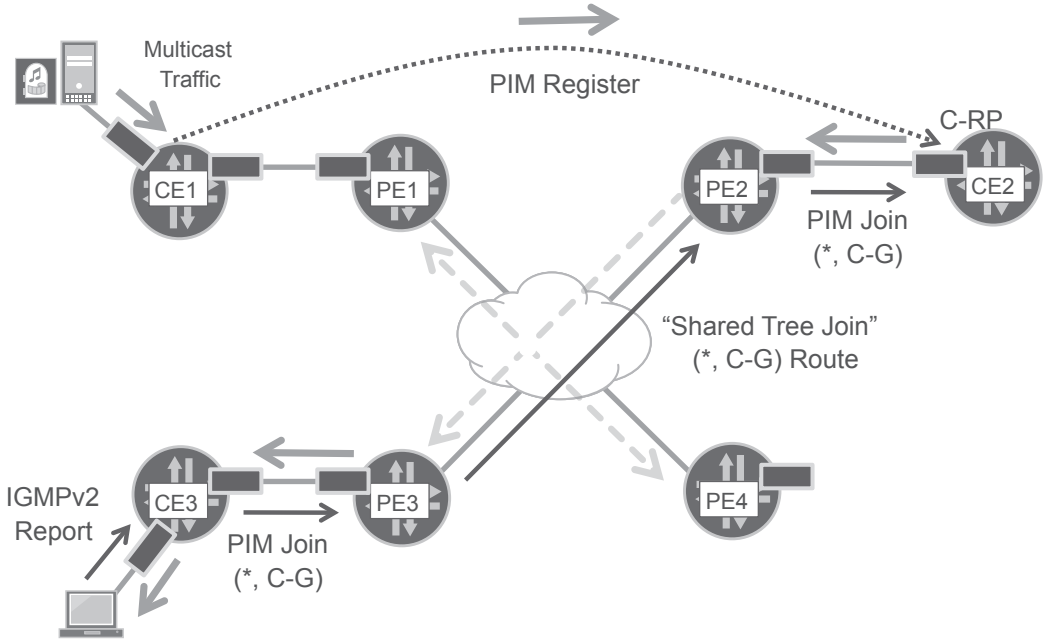


Figure 3.3 Multicast VPN in RPT-SPT Mode – Shared Tree

Figure 3.3 shows a dotted grey line inside the backbone. Is C-Multicast traffic flowing down the Shared Tree or not? Well, it may if PE2 has a PMSI - by default it would use an Inclusive P-Tunnel rooted at PE2 -, but it's completely optional.

IMPORTANT The following steps are a pure control plane process. When the C-RP joins (C-S, C-G), it is indirectly triggering a SPT switchover among all the PEs in the VPN.

The process continues as illustrated in Figure 3.4:

6. CE2 sends a PIM (C-S, C-G) Join to PE2.

7. PE2 sends a (C-S, C-G) Source Tree Join BGP route. This route is targeted to PE1, the PE en route to C-S.

8. PE1 at VRF black sends a PIM (C-S, C-G) Join to CE1.

At this stage, C-Multicast traffic starts flowing down the Source Tree. By default, PE1 uses the Inclusive P-Tunnel rooted at PE1 and mapped to VRF black. Although bringing C-Multicast traffic down to to the receivers is essential for the service, it's not a necessary step for the above process to complete.

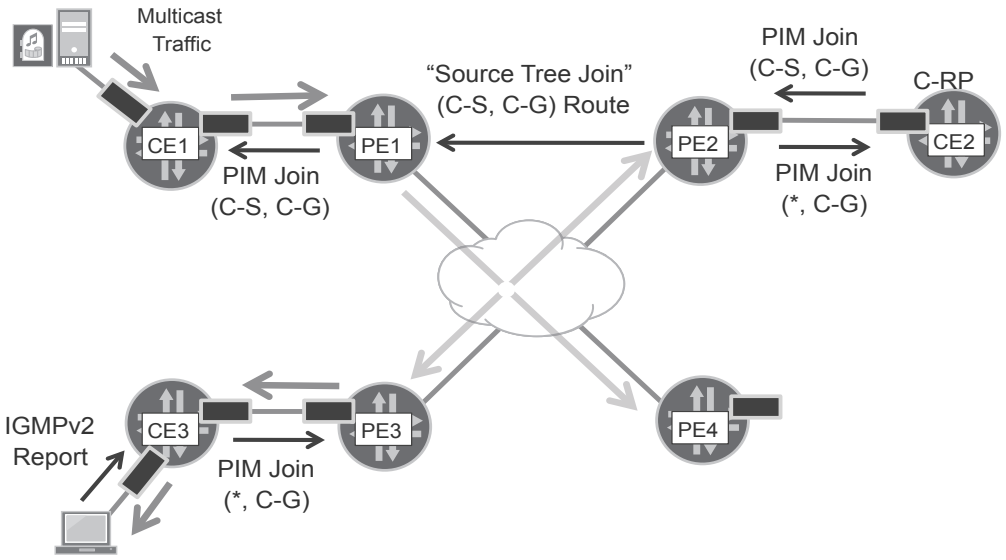


Figure 3.4 Multicast VPN in RPT-SPT Mode – C-RP Joins the Source Tree

At this point, traffic is already flowing end-to-end from the C-Source to the C-Receiver. In the MPLS backbone, it goes from PE1 to PE3 via the Inclusive P-Tunnel rooted at PE1. No further signaling is required from the end-user service perspective. For completeness, the next steps are illustrated in Figure 3.5:

9. PE1 sends a (C-S, C-G) Source Active A-D route targeted to all members of Multicast VPN black.

10. CE3 starts receiving the traffic and, if the SPT threshold is exceeded, it sends a PIM (C-S, C-G) Join to PE3.

11. PE3 sends a (C-S, C-G) Source Tree Join. This route is targeted to PE1, the PE en route to C-S.

12. PE2 sends a PIM (C-S, C-G) Prune to CE2, maintaining the (*, C-G) Join towards CE2.

13. CE2 sends a PIM (C-S, C-G) Prune to PE2.

14. PE2 keeps its (C-S, C-G) Source Tree Join towards PE1, as long as PE2 has downstream (*, C-G) state and the C-Multicast cache entry does not expire.

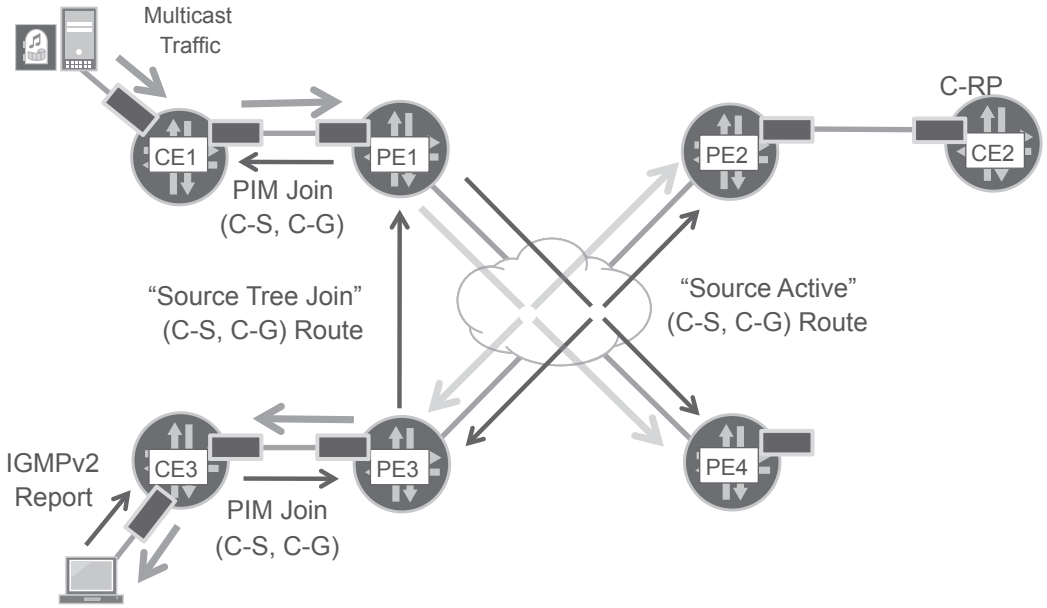


Figure 3.5 Multicast VPN in RPT-SPT Mode – SPT Switchover Completion

From the perspective of PE1, Step 11 makes no difference: since there is a remote Source Tree Join route installed (from PE2), it keeps sending the traffic down the Inclusive P-Tunnel.

Although not shown in Figures 3.4 and 3.5, the Shared Tree Join route sent from PE3 to PE2 is not withdrawn, for the same reason that PIM (*, C-G) state is not pruned after SPT switchover. If a new C-S sends traffic to the same C-G, it would start using the Shared Tree.

Does it look complex? It is indeed complex, but not more than regular SPT convergence process in PIM (see Figure 1.12 and Try It Yourself at the end of Chapter 1).

The steps required to bring the C-Multicast stream to the end receiver are a pure control plane process that brings robustness and deterministic behavior to a process (SPT convergence) that is complex by definition, especially in this topology (C-RP in a stick).

The test bed described requires PE1 and PE2 to be neighbors in the MVPN, as is the case in VPN black. However, they are not neighbors in MVPN white. Does it mean that BGP partial mesh topologies are not an option in the ASM model? No, it just means that sites local to the (one or several) C-RP must be fully meshed to all the other sites in a given MVPN. But there is no need for two non-RP sender-only sites (or two non-RP receiver-only sites) to be connected to each other. It is relatively easy to let route targets play the magic to achieve exactly the partial mesh required for the solution to work in the most efficient way. In small topologies the advantage is not that clear but as the number of sites scale, it becomes an interesting feature.

TIP If you want to capture the whole BGP exchange, feel free to use tcpdump (`monitor traffic interface` in the Junos OS CLI) during any of the steps of this chapter. The tcpdump tool only captures control traffic, so its impact on resources is controlled.

Try It Yourself: Different C-RP Locations

Try to figure out the steps required if CE1 is the C-RP. And what if CE3 is the C-RP? Things become so much simpler!

Choosing and Configuring a C-RP

Very frequently, end customers prefer to have administrative access to the RP, so the RP function is configured in a CE. If the C-RP is directly connected to a source or a receiver, the SPT switchover (if any) is very simple. In order to see and learn about the signaling involved in the C-PIM ASM scenario, an interesting place to define the C-RP function is on CE2, which is not connected to any source or receiver. This is probably not the best network design option, but it's a good choice for this book's purposes.

WARNING Both the first-hop router CE1 and the C-RP CE2 require tunnel-capable hardware in order to encapsulate and decapsulate C-Registers, respectively.

Let's start with a configuration at CE2:

```
user@CE2> configure
user@CE2# set interfaces lo0.1 family inet address 10.111.1.1/32
user@CE2# set routing-instances black interface lo0.1
user@CE2# set routing-instances black protocols pim rp local address 10.111.1.1
user@CE2# commit and-quit
```

Now configure the following at PE2:

```
user@PE2> configure
user@PE2# set routing-instances black routing-options static route 10.111.1.1/32 next-
hop 10.1.2.2
user@PE2# commit and-quit
```

Verify unicast connectivity to the C-RP address 10.111.1.1 from all the routers at VPN black, using ping and displaying the unicast routes.

Now let's configure the following at all PEs and CEs, except for CE2, but including PE2:

```
user@CE1> configure
user@CE1# set routing-instances black protocols pim rp static address 10.111.1.1
user@CE1# commit and-quit
```

Verify at all routers the static C-RP settings with the `show pim rps instance black` operational command.

CAUTION Beware of keywords `local` and `static` as they have different meanings. CE2 must be configured with `rp local`, whereas all the other routers must have `rp static`. This is a very common mistake when building PIM ASM networks.

The first-hop router encapsulates C-Multicast traffic into unicast C-PIM Register packets using a pe- (PIM encapsulation) interface. Let's execute the following command at CE1:

```
user@CE1> show pim interfaces instance black
Instance: PIM.black
```

```
Name             Stat Mode    IP V State NbrCnt JoinCnt(sg) JoinCnt(*g) DR address
ge-0/0/1.1       Up   Sparse   4 2 DR         0           0           0 10.11.1.2
ge-0/0/2.1       Up   Sparse   4 2 DR         1           0           0 10.1.1.2
pe-0/1/0.32769  Up   Sparse   4 2 P2P        0           0           0
```

The C-RP decapsulates C-PIM Register packets using a pd- (PIM decapsulation) interface. Execute the following command at CE2:

```
user@CE2> show pim interfaces instance black
Instance: PIM.black

Name             Stat Mode    IP V State NbrCnt JoinCnt(sg) JoinCnt(*g) DR address
ge-0/0/1.1       Up   Sparse   4 2 DR         0           0           0 10.11.2.2
ge-0/0/2.1       Up   Sparse   4 2 DR         1           0           0 10.1.2.2
lo0.1             Up   Sparse   4 2 DR         0           0           0 10.111.1.1
pd-0/0/0.32769  Up   Sparse   4 2 P2P        0           0           0
```

Now let's verify that CE2 is receiving C-PIM Registers successfully:

```
user@CE2> show pim statistics instance black | match "pim|v2 register"
PIM Message type        Received       Sent  Rx errors
V2 Register                  188          0          0
V2 Register Stop               0        186          0
```

C-Multicast data typically flows through the Shared Tree before SPT switchover, and this requires a P-Tunnel rooted at PE2 for VRF black, so configure the following at PE2:

```
user@PE2> configure
user@PE2# edit routing-instances black
user@PE2# set provider-tunnel rsvp-te label-switched-path-template default-template
user@PE2# commit and-quit
```

NOTE The only reason to configure a P-Tunnel rooted at PE2 is to ensure data flows through the RPT while SPT switchover takes place. However, SPT switchover is a fast process, so the previous configuration step is completely optional. You don't need data transport in the RPT to make the solution work (over the SPT).

The new P-Tunnel should be signaled at this stage. Follow the steps explained in Chapter 2, section *Signaling Inclusive Provider Tunnels*, to verify.

Generating (*, C-G) Join State

CE3 and CE4 have downstream receivers at VRF black for C-G 239.11.11.11. The source is unknown by the receivers, which just joined the multicast group. This behavior can be achieved in three different ways:

- The receivers send IGMPv2 reports for C-G 239.11.11.11.
- The receivers send IGMPv3 reports for C-G 239.11.11.11, specifying a null list of excluded sources.
- The downstream last-hop router interfaces have static (*, C-G) IGMP reports configured. You do not need to change the IGMP version in this case – this is the approach followed in this book, but the previous options are valid, too.

Configure the following at CE3 and CE4:

```
user@CE3> configure
user@CE3# edit protocols igmp
user@CE3# set interface ge-0/0/1.1 static group 239.11.11.11
user@CE3# commit and-quit
```

This configuration triggers a (*, 239.11.11.11) C-Join upstream towards the C-RP. Now execute the following at CE3 and CE4:

```
user@CE3> show route 10.111.1.1 table black

black.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 01:07:08
                    > to 10.1.3.1 via ge-0/0/2.1

user@CE3> show pim join instance black 239.11.11.11
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.11.11.11
    Source: *
    RP: 10.111.1.1
    Flags: sparse,rptree,wildcard
    Upstream interface: ge-0/0/2.1
```

Once PE3 and PE4 receive a (*, C-G) PIM Join from their downstream CE (CE3 and CE4, respectively), they also perform an unicast route resolution towards the C-RP. The result is an inet-vpn BGP route pointing to the MPLS core. Let's see by executing the following commands at PE3 and PE4:

```
user@PE3> show route 10.111.1.1 table black

black.inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.111.1.1/32      *[BGP/170] 00:13:01, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/2.0, Push 16, Push 301472(top)



user@PE3> show pim join instance black 239.11.11.11
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.11.11.11
    Source: *
    RP: 10.111.1.1
    Flags: sparse,rptree,wildcard
    Upstream protocol: BGP
    Upstream interface: Through BGP
```

If you suspect that "Upstream protocol: BGP" is an indication that the (*, C-G) C-Join is converted by PE3 and PE4 into a BGP route, you are fully right. To verify it, execute the following command at PE3 and PE4:

```
user@PE3> show route table black.mvpn extensive | find 6:65000
6:65000:100:65000:32:10.111.1.1:32:239.1.1.1/240 (1 entry, 1 announced)
        *PIM    Preference: 105
                Next hop type: Multicast (IPv4)
```

```
Next-hop reference count: 1
State: <Active Int Ext>
Age: 4:21
Task: PIM.black
Announcement bits (2): 0-PIM.black 1-mvpn global task
AS path: I
Communities: no-advertise target:10.101.2.2:5
```

There is indeed a Type 6 C-Multicast – Shared Tree Join BGP route ready to be sent upstream, whose format is shown in Figure 3.6.

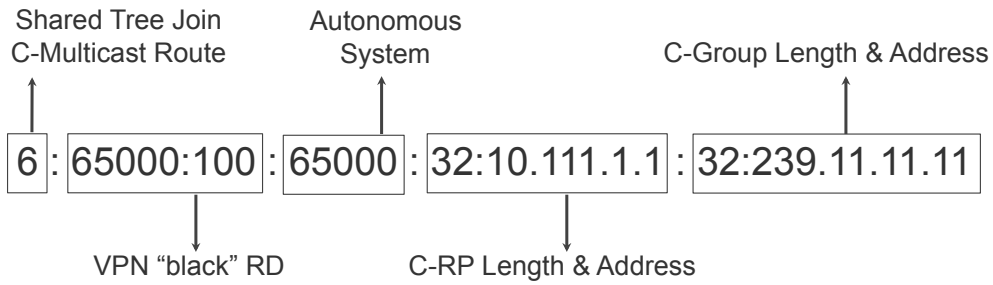


Figure 3.6 Format of a Type 6 C-Multicast – Shared Tree Join Route

However, the route is not advertised at all due to the no-advertise community, as you can verify by executing the following command at PE3 and PE4:

```
user@PE3> show route advertising-protocol bgp 10.101.5.5 table black.mvpn | match 6:65000

user@PE3>
```

Since MVPN black is running in the default `spt-only` mode, the Shared Tree Join is not sent upstream. As a result, C-Multicast traffic to 239.11.11.11 is not forwarded end-to-end:

```
user@PE3> show multicast route instance black group 239.11.11.11

user@PE3>
```

You need something else to make it work. The C-Multicast islands are not fully connected yet.

Turning RPT-SPT Mode On

As previously discussed, there are three models to integrate C-PIM ASM with BGP MVPN. For the purpose of this book the most interesting one is RPT-SPT, which involves a SPT switchover.

CAUTION At the time of this writing, Junos OS does not officially support RPT-SPT mode if the C-Sources or C-Receivers are locally connected to the PEs. It is mandatory to have a CE between a PE and a host. This limitation is specific of RPT-SPT, it does not affect the default SPT-only mode.

TIP The next step will make C-Multicast traffic flow end-to-end, immediately causing a SPT switchover. If you want to watch all the chapters of the movie, including the Shared Tree phase, you may want to configure `set routing-instances black protocols pim spt-threshold infinity 239/8` at CE3 and CE4 now. Don't forget to remove it later!

First let's configure the following at all PEs:

```
user@PE1> configure
user@PE1# set routing-instances black protocols mvpn mvpn-mode rpt-spt
user@PE1# commit and-quit
```

In the time that you read these lines, the SPT switchover is very likely complete. The following procedure guides you step-by-step through all the routes that have taken part in the whole process.

Once the `rpt-spt` mode is active, the downstream PEs advertise Type 6 C-Multicast – Shared Tree Join routes. Let's examine by executing the following command at PE3 and PE4:

```
user@PE3> show route advertising-protocol bgp 10.101.5.5 table black.mvpn

black.mvpn.0: 8 destinations, 9 routes (8 active, 1 holddown, 0 hidden)
Prefix                          Nexthop        MED     Lclpref    AS path
  1:65000:100:10.101.3.3/240
*                          Self                         100        I
  6:65000:100:65000:32:10.111.1.1:32:239.11.11.11/240
*                          Self                         100        I
  7:65000:100:65000:32:10.11.1.1:32:239.1.1.1/240
*                          Self                         100        I
  7:65000:100:65000:32:10.11.1.1:32:239.11.11.11/240
*                          Self                         100        I


user@PE3> show route advertising-protocol bgp 10.101.5.5 table black.
mvpn extensive | match communities
Communities: target:65000:111
Communities: target:10.101.2.2:5
Communities: target:10.101.1.1:5
Communities: target:10.101.1.1:5
```

As you can see there are two Type 7 (C-S, C-G) Source Tree Join routes with a common C-S, 10.11.1.1. The first C-G is 239.1.1.1, a SSM group configured in Chapter 2. The other one is 239.11.11.11, for which the SPT convergence process has just finished.

You can now focus on the Type 6 (*, C-G) Shared Tree Join route. Its Route Target matches the Route Import of the unicast route pointing to the C-RP. Execute the following at PE3 and PE4:

```
user@PE3> show route 10.111.1.1 table black extensive | match communities
Communities: target:65000:111 src-as:65000:0 rt-import:10.101.2.2:5
```

Let's verify that the Shared Tree Join route is correctly imported by the upstream PE en route to the C-RP, by executing the following command at PE2:

```
user@PE2> show route receive-protocol bgp 10.101.5.5 table black.mvpn extensive | find 6:65000
* 6:65000:100:65000:32:10.111.1.1:32:239.11.11.11/240 (1 entry, 0 announced)
     Import Accepted
     Route Distinguisher: 65000:100
     Nexthop: 10.101.5.5
     Localpref: 100
     AS path: I (Originator) Cluster list:  10.101.5.5
     AS path:  Originator ID: 10.101.5.5
     Communities: target:10.101.2.2:5
```

You can see that this route generates local (*, C-G) PIM Join state at PE2 VRF black, towards the C-RP:

```
user@PE2> show pim join instance black
Instance: PIM.black Family: INET
R = Rendezvous Point Tree, S = Sparse, W = Wildcard

Group: 239.11.11.11    Source: *
    RP: 10.111.1.1
    Flags: sparse,rptree,wildcard
    Upstream interface: ge-0/0/2.1

Group: 239.11.11.11
    Source: 10.11.1.1
    Flags: sparse
    Upstream protocol: BGP
    Upstream interface: Through BGP
```

Why is there a (C-S, C-G) Join as well? Well, as part of the signaling when C-RP joined the source, PE2 generated a Source Tree Join BGP route. This route remains as long as there is downstream (*, C-G) state and the multicast cache entry does not expire. Let's execute the following command at PE2:

```
user@PE2> show route advertising-protocol bgp 10.101.5.5 extensive | find 7:65000
* 7:65000:100:65000:32:10.11.1.1:32:239.11.11.11/240 (2 entries, 2 announced)
 BGP group RR type Internal
     Route Distinguisher: 65000:100
     Nexthop: Self
     Flags: Nexthop Change
     Localpref: 100
     AS path: [65000] I
     Communities: target:10.101.1.1:5
```

During the convergence process, PE1 generates a Type 5 Source Active A-D route, illustrated in Figure 3.7. Let's check PE1:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5 table black.mvpn

black.mvpn.0: 7 destinations, 9 routes (7 active, 2 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
  1:65000:100:10.101.1.1/240
*                         Self                          100        I
  5:65000:100:32:10.11.1.1:32:239.11.11.11/240
*                         Self                          100        I


user@PE1> show route advertising-protocol bgp 10.101.5.5 table black.
mvpn extensive | match communities
Communities: target:65000:111
Communities: target:65000:111
```

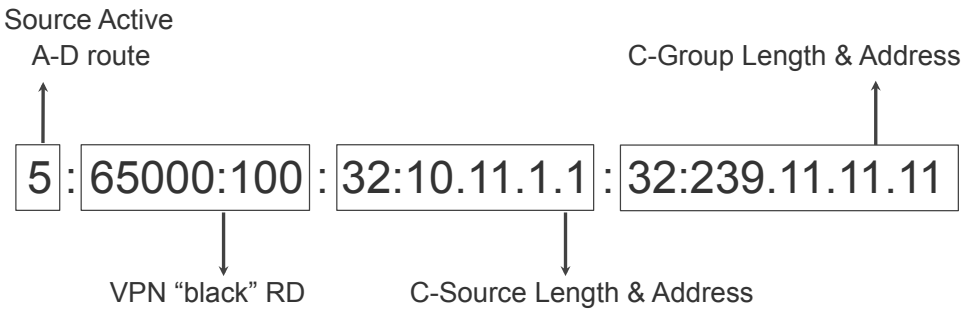


Figure 3.7 Format of a Type 5 Source Active A-D Route

Based on the Route Target, this route should be installed at all the other PEs. If you're following along on your test bed, you should check that at this time.

Now, in order to explore the complete C-Multicast state, execute the following command at all PEs. For completeness, the output for *all* the sites is displayed this time:

```
user@PE1> show mvpn c-multicast instance-name black
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : RPT-SPT
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.11.1.1/32:239.1.1.1/32     RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1     RM
  10.11.1.1/32:239.11.11.11/32  RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1     RM

user@PE2> show mvpn c-multicast instance-name black
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : RPT-SPT
  C-mcast IPv4 (S:G)            Ptnl                                          St
  0.0.0.0/0:239.11.11.11/32     RSVP-TE P2MP:10.101.2.2, 2087,10.101.2.2      RM
  10.11.1.1/32:239.11.11.11/32  RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1

user@PE3> show mvpn c-multicast instance-name black
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : RPT-SPT
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.11.1.1/32:239.1.1.1/32     RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1
  0.0.0.0/0:239.11.11.11/32     RSVP-TE P2MP:10.101.2.2, 2087,10.101.2.2
  10.11.1.1/32:239.11.11.11/32  RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1

user@PE4> show mvpn c-multicast instance-name black
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : RPT-SPT
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.11.1.1/32:239.1.1.1/32     RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1
  0.0.0.0/0:239.11.11.11/32     RSVP-TE P2MP:10.101.2.2, 2087,10.101.2.2
  10.11.1.1/32:239.11.11.11/32  RSVP-TE P2MP:10.101.1.1, 31395,10.101.1.1
```

Now all that's left for you to do is to verify final PIM signaling at all CE-PE links, as well as end-to-end C-Multicast traffic flow. If you need a refresher, the steps are explained in Chapter 2.

TRY THIS Move the C-RP to other sites and see the effect. Also, define an additional C-RP, and build a redundant C-RP architecture based on Anycast. This is quite transparent for Multicast VPN in terms of configuration and signaling.

Answers to Try It Yourself Sections of Chapter 3

Try It Yourself: Different C-RP Locations

If CE1 is the Customer Rendezvous Point (C-RP), the sequence is:

1. The C-Receiver sends an IGMPv2 (*, C-G) Report.
2. CE3 sends a PIM (*, C-G) Join to PE3 at VRF black.
3. PE3 sends a (*, C-G) Shared Tree Join BGP route, targeted to PE1.
4. PE1 at VRF black sends a PIM (*, C-G) Join to CE1.
5. Traffic flows down the Shared Tree, which happens to match the Shortest Path Tree.
6. If CE3 starts a (data-driven) SPT switchover, move to next step. Otherwise, stop here.
7. CE3 sends a PIM (C-S, C-G) Join to PE3 at VRF black.
8. PE3 sends a (C-S, C-G) Source Tree Join BGP route, targeted to PE1.
9. PE1 at VRF black sends a PIM (C-S, C-G) Join to CE1.
10. PE1 sends a (C-S, C-G) Source Active A-D BGP route, targeted to all PEs in MVPN black.

If CE3 is the Customer Rendezvous Point (C-RP), the sequence is:

1. The C-Receiver sends an IGMPv2 (*, C-G) Report.
2. CE1 encapsulates C-Multicast traffic into unicast C-Register-Start packets sent to the C-RP (CE3).
3. CE3 decapsulates the C-Multicast traffic and sends it down to the C-Receiver.
4. CE3 sends a PIM (C-S, C-G) Join to PE3 at VRF black.
5. PE3 sends a (C-S, C-G) Source Tree Join BGP route, targeted to PE1.
6. PE1 at VRF black sends a PIM (C-S, C-G) Join to CE1.
7. C-Multicast traffic flows down the Shortest Path Tree.
8. PE1 sends a (C-S, C-G) Source Active A-D BGP route, targeted to all PEs in MVPN black.
9. CE3 sends a C-PIM Register-Stop to CE1.

Chapter 4

Selective Trees for Bandwidth Optimization

All multicast technologies have a classic dilemma: if you try to minimize the signaling required to build distribution trees, forwarding is not optimized and bandwidth is wasted. If, on the other hand, you make sure traffic is distributed only where it should be, it typically comes at the expense of more signaling and higher control-plane load. *Inclusive Trees* are a strategy for the first dilemma, and *Selective Trees* are a strategy for the second dilemma. Of course, both have advantages and drawbacks.

NOTE There is a description of the Inclusive and Selective PMSI concepts in Chapter 1.

Inclusive Trees are an appropriate solution to transport (C-S, C-G) flows with local C-Receivers in most of the Egress PEs. They are also a good design choice for small volume flows with low bit rate. But when the overall bandwidth requirements and the distribution of the receivers makes the Inclusive Trees unsuitable, it is necessary to map certain C-Multicast flows to Selective Trees.

Review of I-PMSI and S-PMSI Auto-Discovery

The BGP procedures to Auto-Discover Inclusive and Selective Trees have significant differences.

First, let's quickly review Inclusive Tree Auto-Discovery. Back in Figure 2.4, PE1 looks at all the I-PMSI A-D routes received for VPN black. It finds one from each of the remote PEs. As a result, PE2, PE3, and PE4 become leaves of the Inclusive Tree rooted at PE1. When a Receiver PE like PE3 sends an I-PMSI A-D route, it is basically saying: if you are a Sender PE for VPN black and install this route, feel free to make me a leaf of the Inclusive Tree rooted at you. It's more like a wildcard than a specific request to join a particular tree.

If you flip back to Figure 2.6, you can see the distribution of I-PMSI A-D routes can be controlled with Route Targets. This has a direct influence on the layout of the Inclusive Trees, but it is still likely that C-Multicast traffic reaches an Egress PE with no local C-Receivers for the particular flow.

PE1 gets from PE3 a C-Multicast – Source Tree Join request to send (C-S, C-G) traffic into the white Inclusive Tree. PE3 has downstream receivers, but that is not the case for PE4. So the traffic is silently discarded by PE4.

Selective Trees have a more targeted Auto-Discovery Mechanism. Upon reception of a (C-S, C-G) Source Tree Join route, PE1 checks its local configuration for a matching PMSI. If the result is a Selective PMSI, a new tunnel is signaled like in Figure 4.2. PE1 sends a Type 3 S-PMSI A-D BGP route, announcing itself as the root of a new P-Tunnel that will transport (C-S, C-G) traffic. PE3 and PE4 install the route in the the white.mvpn.0 table according to the Route Target policies.

Only the PEs with downstream receivers for (C-S, C-G) or (*, C-G) become leaves of the Selective Tree. PE3 sends a Type 4 Leaf A-D route targeted to PE1. This route contains enough information for PE1 to realize that it was sent in response of its own S-PMSI A-D route. PE4 does not have downstream receivers so it does not send any Leaf A-D route. Once PE1 has discovered the leaves of the Selective Tree, it signals a P2MP RSVP LSP. In this case, the LSP only has one branch, which makes it look like point-to-point, but of course it could have more leaves if other Receiver PEs sent a 'Leaf A-D' route.

NOTE If the chosen P-Tunnel technology is based on P-PIM or LDP, then Leaf A-D routes are not necessary since the tunnels are signaled from downstream PEs.

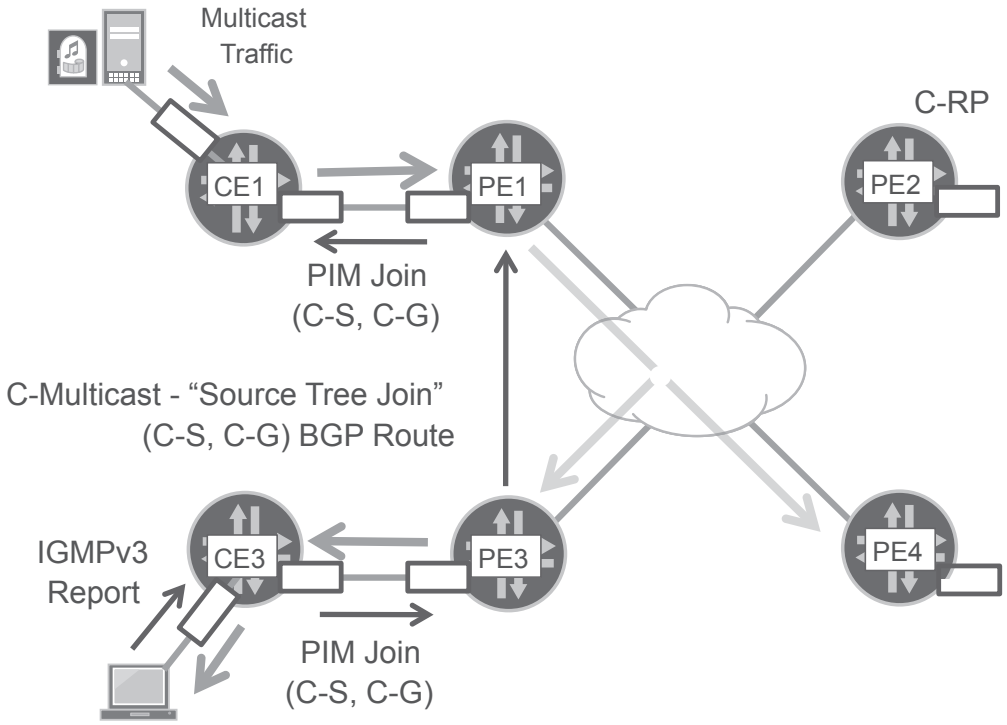


Figure 4.1 C-Multicast Traffic Transported via an Inclusive Tree

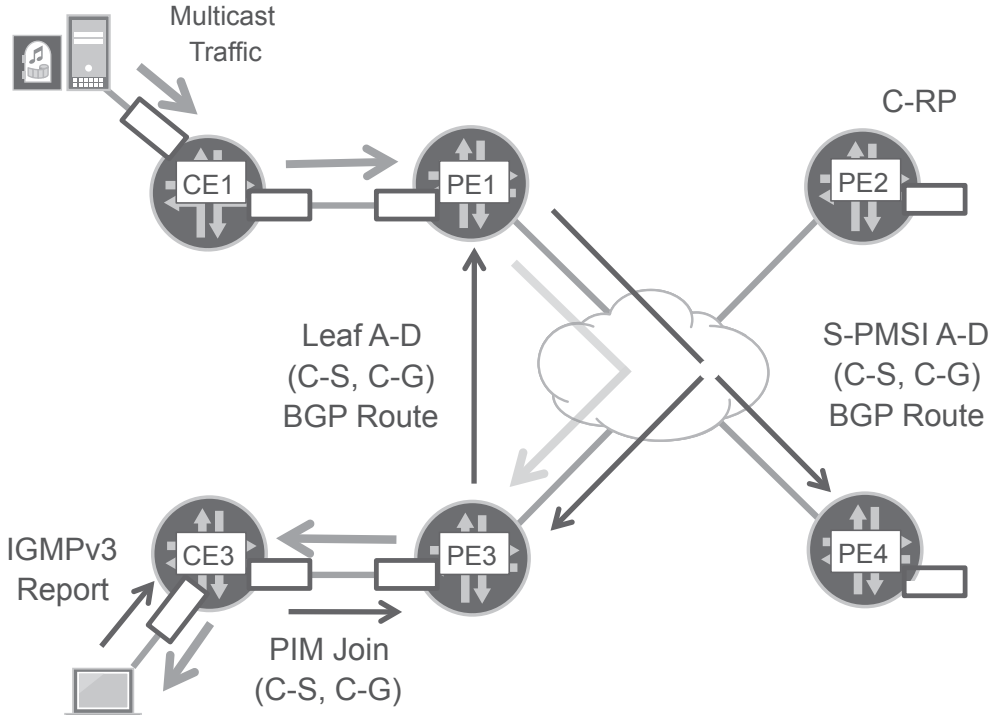


Figure 4.2 Selective Tree Signaling (The Source Tree Join route from PE3 to PE1 is not shown for the sake of simplicity.)

If you configure a Selective Tree for a flow that is already being tunneled into the Inclusive Tree, PE1 does not switch (C-S, C-G) traffic to the S-PMSI immediately. PE1 keeps sending it to the I-PMSI for some time before switchover, allowing PE3 to get ready to receive C-Multicast packets from the Selective Tree.

The Leaf A-D route is a request to be a leaf of a Selective Tree, but it does not generate any C-Multicast state. In order to send (C-S, C-G) traffic into a P-Tunnel, PE1 needs an explicit request, and this request already came from PE3 as a Type 7 - Source Tree Join route targeted to PE1. This route is maintained before and after switching traffic from the Inclusive to the Selective Tree.

What if PE4 wants to become a leaf of the Selective Tree? It already has the S-PMSI A-D route in its `white.mvpn.0` table, so it just needs to send a Leaf A-D route targeted to PE1.

A Sender PE in a given VPN can be the root of several Selective Trees. The Sender PE (PE1) has a local configuration mapping specific (C-S, C-G) to S-PMSI type and properties. In Junos, source and group netmasks are allowed for greater flexibility. In order to determine where a given (C-S, C-G) is mapped to, a best-match lookup is performed, and if there is no matching S-PMSI, the traffic is mapped to the I-PMSI. If there is no I-PMSI defined, then that particular flow cannot be transported to other sites. As with unicast, you can think of S-PMSI as specific routes, and of I-PMSI as the default route. All of them are optional.

NOTE Suppose that the local configuration at PE1 maps (10/8, 239.1.1.1/32) to a given RSVP P2MP LSP template, and there is no other specific mappings. Then PE1 receives two Source Tree Join routes for (10.1.1.1, 239.1.1.1) and (10.1.1.10, 239.1.1.1), respectively. At this point, PE1 signals two different Selective Trees, one for each (C-S, C-G). Both RSVP P2MP LSPs are cloned from the same template, but the two (C-S, C-G) S-PMSI A-D routes have different PMSI attributes.

Subscribing to a New (C-S, C-G)

There is already a C-Multicast flow (10.22.1.1, 239.2.2.2) at VPN white transported via the Inclusive Tree. The traffic reaches PE3 and PE4, which have downstream receivers connected to CE3 and CE4, respectively. The other active flow (10.22.1.1, 239.22.22.22) is not being transported since it has no receivers yet.

Let's begin by configuring the following only at CE3:

```
user@CE3> configure
user@CE3# edit protocols igmp
user@CE3# set interface ge-0/0/1.2 static group 239.22.22.22 source 10.22.1.1
user@CE3# commit and-quit
```

Following the same steps as in Chapter 2, you can check the signaling that is involved in order to bring the C-Multicast flow to the receiver. PE1 installs a Type 7 C-Multicast – Source Tree Join route originated by PE3 for (10.22.1.1, 239.22.22.22). As a result, the new flow reaches both PE3 and PE4. Let's issue the following commands at PE3 and PE4:

```
user@PE3> show mvpn c-multicast instance-name white
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : white
```

```
  MVPN Mode : SPT-ONLY
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.22.1.1/32:239.2.2.2/32     RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1
  10.22.1.1/32:239.22.22.22/32  RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1

user@PE3> show multicast route instance white group 239.22.22.22
Family: INET

Group: 239.22.22.22
    Source: 10.22.1.1/32
    Upstream interface: lsi.2
    Downstream interface list:
        ge-0/0/2.2

user@PE3> show multicast route instance white group 239.22.22.22 extensive | match pps
    Statistics: 18 kBps, 99 pps, 20505 packets


user@PE4> show mvpn c-multicast instance-name white
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : white
  MVPN Mode : SPT-ONLY
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.22.1.1/32:239.2.2.2/32     RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1

user@PE4> show multicast route instance white group 239.22.22.22
Family: INET

Group: 239.22.22.22
    Source: 10.22.1.1/32
    Upstream interface: lsi.1

user@PE4> show multicast route instance white group 239.22.22.22 extensive | match pps
    Statistics: 17 kBps, 98 pps, 21972 packets
```

You can see that as opposed to PE4, PE3 has C-Multicast state and downstream receivers for (10.22.1.1, 239.22.22.22). However, PE4 is also receiving the traffic, and this is inefficient bandwidth utilization, so let's proceed.

Mapping (C-S, C-G) to a Selective PMSI

All P-Tunnels are rooted at an Ingress PE and that's also the case for Selective Trees. So if the relevant configuration is performed at PE1, it is up to the Receiver PEs to join the new tree or not. Let's start by configuring the following at PE1:

```
user@PE1> configure
user@PE1# edit routing-instances white
user@PE1# set provider-tunnel selective group 239.22.22.22/32 source 10.22.1.1/32 rsvp-te label-
switched-path-template default-template
user@PE1# commit-and-quit
```

With this, PE1 starts advertising a Type 3 S-PMSI Auto-Discovery route. The format of the route is illustrated in Figure 4.3, but let's execute the following commands at PE3 before viewing the figure:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5 table white.mvpn

white.mvpn.0: 7 destinations, 9 routes (7 active, 2 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
```

```
  1:65000:200:10.101.1.1/240
*                         Self                          100        I
  3:65000:200:32:10.22.1.1:32:239.22.22.22:10.101.1.1/240
*                         Self                          100        I

user@PE1> show route advertising-protocol bgp 10.101.5.5 table white.
mvpn extensive | match “communities|pmsi”
     Communities: target:65000:22
     PMSI: Flags 0x0: Label[0:0:0]: RSVP-TE: Session_13[10.101.1.1:0:31396:10.101.1.1]
     Communities: target:65000:22
     PMSI: Flags 0x1: Label[0:0:0]: RSVP-TE: Session_13[10.101.1.1:0:51182:10.101.1.1]
```

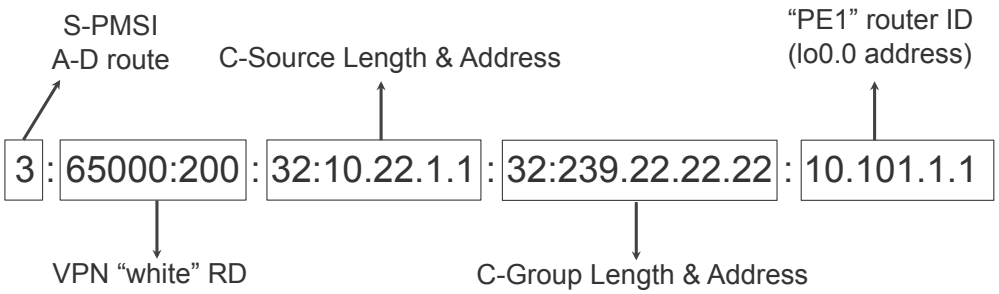


Figure 4.3 Format of a Type 3 S-PMSI Auto-Discovery Route

The Route Targets (RT) match the configured VPN white policies: a route originated by a Sender PE and targeted to all Receiver PEs, carries RT `target:65000:22`.

The PMSI attribute in the I-PMSI and S-PMSI routes is practically identical. The only change is the Tunnel ID, which is different.

NOTE The generation of a S-PMSI A-D route is driven by a pure control plane mechanism. It is triggered by the reception of matching C-Multicast - Source/Shared Tree Join routes, and not by the (C-S, C-G) traffic.

Upon reception of the Type 3 S-PMSI A-D route, PE3 sends a Type 4 Leaf A-D route targeted to PE1. The format of this route is illustrated in Figure 4.4. Let’s first examine the output of the following commands at PE3 where you can see the format of a Type 4 Leaf A-D route at work:

```
user@PE3> show route advertising-protocol bgp 10.101.5.5 table white.mvpn

white.mvpn.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
  1:65000:200:10.101.3.3/240
*                         Self                          100        I
  4:3:65000:200:32:10.22.1.1:32:239.22.22.22:10.101.1.1:10.101.3.3/240
*                         Self                          100        I
  7:65000:200:65000:32:10.22.1.1:32:239.2.2.2/240
*                         Self                          100        I
  7:65000:200:65000:32:10.22.1.1:32:239.22.22.22/240
*                         Self                          100        I

user@PE3> show route advertising-protocol bgp 10.101.5.5 table white.
mvpn extensive | match communities
     Communities: target:65000:2
     Communities: target:10.101.1.1:0
     Communities: target:10.101.1.1:6
     Communities: target:10.101.1.1:6
```

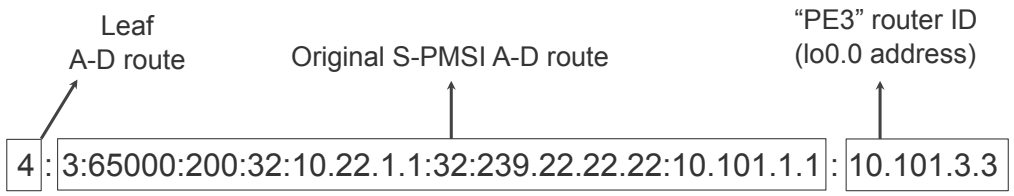


Figure 4.4 Format of a Type 4 Leaf Auto-Discovery Route

And here is an explanation of the different Route Targets used:

- I-PMSI A-D route: This route is announced by Receiver PEs and must be imported by Sender PEs. According to VPN white configured policies and Figure 2.6, the RT is `target:65000:2`.
- Leaf A-D route: This route is targeted to PE1 only, so the RT `target:10.101.1.1:0` contains its Router ID.
- C-Multicast – Source Tree Join route: This route is targeted to PE1 and VRF white. The RT `target:10.101.1.1:6` contains PE1 Router ID and a local identifier of the VRF.

Try It Yourself: Examine the Internal Policies

Route Targets `target:65000:2` and `target:65000:22` have been configured in VRF white policies. However, there are other RTs that also play their part in the solution. At PE1, execute the commands `show policy` and `show policy <policy_name>` to answer these questions:

1. What policy is responsible for importing Leaf A-D Routes?
2. What policy is responsible for importing C-Multicast Routes into VRF white?
3. What policy is responsible for setting Route Import communities to unicast (`family inet`) routes exported from VRF white?

Forwarding in Selective Trees

The technology chosen here for the Selective P-Tunnels is also based on RSVP P2MP LSPs, which you already explored back in Chapter 2. At this point, there is one Inclusive and one Selective Tree rooted at PE1 for VRF white. Let's take a quick look at PE1:

```
user@PE1> show rsvp session p2mp ingress
Ingress RSVP: 6 sessions
P2MP name: 65000:200:mvpn:white, P2MP branch count: 2
To            From          State   Rt Style Labelin Labelout LSPname
10.101.3.3    10.101.1.1    Up      0  1 SE      -   303488  10.101.3.3:65000:200:mvpn:white
10.101.4.4    10.101.1.1    Up      0  1 SE      -   303488  10.101.4.4:65000:200:mvpn:white
P2MP name: 65000:200:mv1:white, P2MP branch count: 1
To            From          State   Rt Style Labelin Labelout LSPname
10.101.3.3    10.101.1.1    Up      0  1 SE      -   304272 10.101.3.3:65000:200:mv1:white

/*** Lines related to black Inclusive Tree are ommited ***/
```

As expected, the Selective Tree only has one leaf. Now let's look at PE3:

```
user@PE3> show rsvp session p2mp egress
Egress RSVP: 3 sessions
P2MP name: 65000:200:mvpn:white, P2MP branch count: 1
To              From            State   Rt Style Labelin Labelout LSPname
10.101.3.3      10.101.1.1      Up       0  1 SE      17        - 10.101.3.3:65000:200:mvpn:white
P2MP name: 65000:200:mv1:white, P2MP branch count: 1
To              From            State   Rt Style Labelin Labelout LSPname
10.101.3.3      10.101.1.1      Up       0  1 SE      17        - 10.101.3.3:65000:200:mv1:white

/*** Lines related to black Inclusive Tree are ommited ***/
```

PE1 has a different next-hop for (10.22.1.1, 239.2.2.2) and (10.22.1.1, 239.22.22.22) C-Multicast routes, the first going to the I-PMSI, and the second to the S-PMSI just signaled. Execute the following command at all PEs (PE1 and PE3 are shown here):

```
user@PE1> show mvpn c-multicast instance-name white
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : SPT-ONLY
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.22.1.1/32:239.2.2.2/32     RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1     RM
  10.22.1.1/32:239.22.22.22/32  S-RSVP-TE P2MP:10.101.1.1, 51182,10.101.1.1   RM

user@PE3> show mvpn c-multicast instance-name white
[...]
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : black
  MVPN Mode : SPT-ONLY
  C-mcast IPv4 (S:G)            Ptnl                                          St
  10.22.1.1/32:239.2.2.2/32     RSVP-TE P2MP:10.101.1.1, 31396,10.101.1.1
  10.22.1.1/32:239.22.22.22/32  S-RSVP-TE P2MP:10.101.1.1, 51182,10.101.1.1
```

But the final proof is checking that PE3 is receiving the (10.22.1.1, 239.22.22.22) C-Multicast traffic, while PE4 is not. Execute the following commands at PE3 and PE4:

```
user@PE3> show multicast route instance white group 239.22.22.22 extensive | match pps
    Statistics: 18 kBps, 100 pps, 38285 packets

user@PE4> show multicast route instance white group 239.22.22.22 extensive | match pps

user@PE4>
```

The output at PE4 may have one line with 0 pps, if the entry has not expired yet from the cache. Feel free to do a more exhaustive control and forwarding plane check on your own test bed if you're following along, as explained in Chapter 1.

TRY THIS Generate more flows and Selective Trees at VPN white. Explore the flexibility of mapping C-Multicast traffic to S-PMSIs templates using network masks.

MORE? In RPT-SPT mode, you can use the `wildcard-source` and `wildcard-group-inet` keywords to effectively map (*, C-G) or even (*, *) traffic to a S-PMSI. This reduces the amount of signaling, as a single Selective P-Tunnel can transport many C-Multicast flows, at the expense of less bandwidth optimization. A given (C-S, C-G) flow would be mapped to a PMSI in the following way, starting from the most preferred option if available. S-PMSI configured for (C-S, C-G). S-PMSI configured for (*, C-G). S-PMSI configured for (*, *). Finally, I-PMSI. You can check the details at [S-PMSI-WILDCARD] or, even better, try it!

TRY THIS Up to a challenge? In the RPT-SPT mode example described in Chapter 3, you can prevent CE3 from initiating SPT switchover by setting its SPT threshold to infinity. In that case, traffic was going from PE1 to PE3 via the Inclusive Tree, but what if PE1 uses Selective Trees in VPN black? Short answer: PE3 still joins the Selective Tree. Long answer: give it a try!

References

[S-PMSI-WILDCARD] http://tools.ietf.org/html/draft-rekhter-l3vpn-mvpn-wildcard-spmsi.

Answers to Try It Yourself Sections of Chapter 4

Try It Yourself: Examine the Internal Policies

These are the answers to the three questions:

1. __vrf-mvpn-import-cmcast-leafAD-global-internal__
2. __vrf-mvpn-import-cmcast-white-internal__
3. __vrf-mvpn-export-inet-white-internal__

Appendix

This Appendix contains the initial configuration of each of the routers included in the test bed for this book. Only the relevant sections for the Multicast VPN scenario are displayed: `interfaces`, `protocols`, `policy-options`, `routing-options`, and `routing-instances`. Other generic information like system or management IP addresses are omitted, as they have no influence on the solution built here in this book, and would be different on your test bed anyway.

NOTE The hardware paths of the interfaces (like FPC and PIC slot numbers) are not important for the setup. Using other types of access or core interfaces is also allowed, *as long as* the core links support `vrf-table-label`.

Initial Configuration of a CE Router

All the CEs have nearly identical baseline configurations, except for the fact that different IP addresses are configured at the interfaces and static route hierarchies. The configuration below corresponds to CE1. Please refer to Figure 2.1 in order to configure the other CEs.

```
interfaces {
    ge-0/0/1 {
        vlan-tagging;
        unit 1 {
            vlan-id 101;
            family inet {
                address 10.11.1.2/30;
            }
        }
        unit 2 {
            vlan-id 102;
            family inet {
                address 10.22.1.2/30;
            }
        }
    }
    ge-0/0/2 {
        vlan-tagging;
        unit 1 {
            vlan-id 101;
            family inet {
                address 10.1.1.2/30;
            }
        }
        unit 2 {
            vlan-id 102;
            family inet {
                address 10.2.1.2/30;
            }
        }
    }
}
routing-instances {
    black {
        instance-type virtual-router;
        interface ge-0/0/1.1;
        interface ge-0/0/2.1;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.1.1.1;
            }
```

```
        }
    }
    white {
        instance-type virtual-router;
        interface ge-0/0/1.2;
        interface ge-0/0/2.2;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.2.1.1;
            }
        }
    }
}
```

Initial Configuration of a PE Router

All the PEs have nearly identical baseline configurations, except for the IP and ISO addresses, static routes, and core interface hardware paths (i.e. so-0/1/2 vs so-0/1/0). The configuration below corresponds to PE1. Please refer to Figure 2.1 in order to configure the other PEs.

```
interfaces {
    ge-0/0/2 {
        vlan-tagging;
        unit 1 {
            vlan-id 101;
            family inet {
                address 10.1.1.1/30;
            }
        }
        unit 2 {
            vlan-id 102;
            family inet {
                address 10.2.1.1/30;
            }
        }
    }
    ge-0/0/3 {
        unit 0 {
            family inet {
                address 10.100.5.1/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/1/0 {
        unit 0 {
            family inet {
                address 10.100.1.2/30;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.101.1.1/32;
            }
```

```
            family iso {
                address 49.0000.0000.0001.00;
            }
        }
    }
}
routing-options {
    router-id 10.101.1.1;
    autonomous-system 65000;
}
protocols {
    rsvp {
        interface ge-0/0/3.0;
        interface so-0/1/0.0;
    }
    mpls {
        interface ge-0/0/3.0;
        interface so-0/1/0.0;
        }
    }
    bgp {
        group RR {
            type internal;
            local-address 10.101.1.1;
            family inet-vpn {
                unicast;
            }
            neighbor 10.101.5.5;
        }
    }
    isis {
        level 1 disable;
        interface ge-0/0/3.0 {
            level 2 metric 50;
        }
        interface so-0/1/0.0;
    }
    ldp {
        interface ge-0/0/3.0;
        interface so-0/1/0.0;
    }
}
routing-instances {
    black {
        instance-type vrf;
        interface ge-0/0/2.1;
        route-distinguisher 65000:100;
        vrf-target target:65000:111;
        vrf-table-label;
        routing-options {
            static {
                route 10.11.1.0/30 next-hop 10.1.1.2;
            }
        }
    }
    white {
        instance-type vrf;
        interface ge-0/0/2.2;
        route-distinguisher 65000:200;
        vrf-import white-imp;
        vrf-export white-exp;
        vrf-table-label;
        routing-options {
```

```
            static {
                route 10.22.1.0/30 next-hop 10.2.1.2;
            }
        }
    }
}
policy-options {
    policy-statement white-exp {
        term unicast {
            from family inet;
            then {
                community add white-target;
                accept;
            }
        }
    }
    policy-statement white-imp {
        term unicast {
            from {
                family inet;
                community white-target;
            }
            then accept;
        }
    }
    community white-target members target:65000:222;
}
```

NOTE If you're wondering why VRF black includes a vrf-target statement, whereas VRF white has a set of `vrf-import` and `vrf-export` policies, the reason is explained in depth in Chapter 2.

Initial Configuration of the P Router

There is only one P-router in the topology depicted in Figure 2.1. This is its initial configuration:

```
interfaces {
    so-0/1/0 {
        unit 0 {
            family inet {
                address 10.100.1.1/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/1/1 {
        unit 0 {
            family inet {
                address 10.100.2.1/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/1/2 {
        unit 0 {
            family inet {
                address 10.100.3.1/30;
```

```
            }
            family iso;
            family mpls;
        }
    }
    so-0/1/3 {
        unit 0 {
            family inet {
                address 10.100.4.1/30;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.101.5.5/32;
            }
            family iso {
                address 49.0000.0000.0005.00;
            }
        }
    }
}
routing-options {
    router-id 10.101.5.5;
    autonomous-system 65000;
}
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        group RR-CLIENTS {
            type internal;
            local-address 10.101.5.5;
            family inet-vpn {
                unicast;
            }
            cluster 10.101.5.5;
            neighbor 10.101.1.1;
            neighbor 10.101.2.2;
            neighbor 10.101.3.3;
            neighbor 10.101.4.4;
        }
    }
    isis {
        level 1 disable;
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
```

```
        ldp {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
```

Basic Connectivity Tests

Multicast VPN services rely on a stable unicast routing infrastructure. Before configuring IP Multicast, it is important to verify that all routers have the necessary unicast routes, and to check end-to-end PE-CE and CE-CE reachability.

Unicast Routes and End-to-End Reachability at the CEs

The CE routers just have a set of directly connected routes, as well as a default static route pointing to the directly attached PE:

```
user@CE1> show route table black

black.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 1d 03:22:40
                    > to 10.1.1.1 via ge-0/0/2.1
10.1.1.0/30        *[Direct/0] 1d 03:22:40
                    > via ge-0/0/2.1
10.1.1.2/32        *[Local/0] 1d 03:22:40
                      Local via ge-0/0/2.1
10.11.1.0/30       *[Direct/0] 1d 03:22:40
                    > via ge-0/0/1.1
10.11.1.2/32       *[Local/0] 1d 03:22:40
                      Local via ge-0/0/1.1

user@CE1> show route table white

white.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 1d 03:22:40
                    > to 10.2.1.1 via ge-0/0/2.2
10.2.1.0/30        *[Direct/0] 1d 03:22:40
                    > via ge-0/0/2.2
10.2.1.2/32        *[Local/0] 1d 03:22:40
                      Local via ge-0/0/2.2
10.22.1.0/30       *[Direct/0] 1d 03:22:40
                    > via ge-0/0/1.2
10.22.1.2/32       *[Local/0] 1d 03:22:40
                      Local via ge-0/0/1.2
```

Once the unicast routes are verified, you can check the reachability of all the remote CEs. Here is an example of a test between CE1 and CE2 for VPN black:

```
user@CE1> ping 10.1.2.2 routing-instance black count 1
PING 10.1.2.2 (10.1.2.2): 56 data bytes
64 bytes from 10.1.2.2: icmp_seq=0 ttl=62 time=1.340 ms

--- 10.1.2.2 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
```

```
round-trip min/avg/max/stddev = 1.340/1.340/1.340/0.000 ms
```

The 0% packet loss figure should be reported for the remaining tests from CE1 (and accordingly from other CEs):

```
user@CE1> ping 10.11.2.2 routing-instance black count 1
user@CE1> ping 10.1.3.2 routing-instance black count 1
user@CE1> ping 10.11.3.2 routing-instance black count 1
user@CE1> ping 10.1.4.2 routing-instance black count 1
user@CE1> ping 10.11.4.2 routing-instance black count 1
user@CE1> ping 10.2.2.2 routing-instance white count 1
user@CE1> ping 10.22.2.2 routing-instance white count 1
user@CE1> ping 10.2.3.2 routing-instance white count 1
user@CE1> ping 10.22.3.2 routing-instance white count 1
user@CE1> ping 10.2.4.2 routing-instance white count 1
user@CE1> ping 10.22.4.2 routing-instance white count 1
```

Unicast Routes and Routing Protocols at the PEs

Each PE has IS-IS and LDP adjacencies with the P and with another PE, according to Figure 2.1. In other words, there are two IS-IS adjacencies, two LDP neighbors, and two LDP sessions at each PE. You can check that with the following commands:

```
user@PE1> show isis adjacency
user@PE1> show ldp neighbor
user@PE1> show ldp session
```

Each PE has one single BGP session with the P router, which acts as a Route Reflector:

```
user@PE1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0           12         12          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.101.5.5            65000         70         58       0       1       24:00 Establ
  bgp.l3vpn.0: 12/12/12/0
  black.inet.0: 6/6/6/0
  white.inet.0: 6/6/6/0
```

There are three remote PEs, and each PE advertises both a direct and a static route (PE-CE link and CE-host link). So there should be six different inet-vpn BGP routes imported at each VRF. Verify that each PE is advertising two routes for each VRF:

```
user@PE1> show route advertising-protocol bgp 10.101.5.5

black.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
* 10.1.1.0/30             Self                          100        I
* 10.11.1.0/30            Self                          100        I

white.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
  Prefix     Nexthop       MED     Lclpref    AS path
* 10.2.1.0/30             Self                          100        I
* 10.22.1.0/30            Self                          100        I
```

Display all the routes installed at each VRF (only black is shown here):

```
user@PE1> show route table black

black.inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
```

```
+ = Active Route, - = Last Active, * = Both

10.1.1.0/30        *[Direct/0] 09:21:10
                    > via ge-0/0/2.1
10.1.1.1/32        *[Local/0] 09:21:10
                      Local via ge-0/0/2.1
10.1.2.0/30        *[BGP/170] 00:23:57, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/0.0, Push 16, Push 301472(top)
10.1.3.0/30        *[BGP/170] 00:23:53, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/0.0, Push 17, Push 303312(top)
10.1.4.0/30        *[BGP/170] 00:02:57, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/0.0, Push 16, Push 303328(top)
10.11.1.0/30       *[Static/5] 09:21:10
                    > to 10.1.1.2 via ge-0/0/2.1
10.11.2.0/30       *[BGP/170] 00:23:57, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/0.0, Push 16, Push 301472(top)
10.11.3.0/30       *[BGP/170] 00:23:53, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/0.0, Push 17, Push 303312(top)
10.11.4.0/30       *[BGP/170] 00:02:57, localpref 100, from 10.101.5.5
                      AS path: I
                    > via so-0/1/0.0, Push 16, Push 303328(top)
```

TRY THIS What MPLS labels are being pushed? The top label is learned via LDP, and inner label via BGP. Execute `show route table inet.3`, `show ldp database`, and `show route receive-protocol 10.101.5.5` for more details.

What to Do Next & Where to Go

www.juniper.net/dayone

The *Day One* book series is available free download in PDF format here. Select titles also feature a *Copy and Paste* edition for direct placement of Junos configurations. (The library is available in eBook format for iPads and iPhones from the Apple iBookstore, or download to Kindles, Androids, Blackberrys, Macs and PCs by visiting the Kindle Store. In addition, print copies are available for sale at Amazon or Vervante.com.)

www.juniper.net/books

Check out the complete Juniper Networks Books library.

forums.juniper.net/jnet

The Juniper-sponsored J-Net Communities forum is dedicated to sharing information, best practices, and questions about Juniper products, technologies, and solutions. Register to participate in this free forum.

www.juniper.net/techpubs/

Juniper Networks technical documentation includes everything you need to understand and configure all aspects of Junos, including BGP MVPN. The documentation set is both comprehensive and thoroughly reviewed by Juniper engineering.

www.juniper.net/training/fasttrack

Take courses online, on location, or at one of the partner training centers around the world. The Juniper Network Technical Certification Program (JNTCP) allows you to earn certifications by demonstrating competence in configuration and troubleshooting of Juniper products. If you want the fast track to earning your certifications in enterprise routing, switching, or security use the available online courses, student guides, and lab guides.

www.juniper.net/us/en/local/pdf/whitepapers/2000291-en.pdf

A white paper on emerging Multicast VPN Applications.

www.juniper.net/us/en/local/pdf/whitepapers/2000320-en.pdf

A white paper on understanding Junos OS BGP Multicast VPNs.