

Chapter 8

Routing Instances Configuration Guidelines

You can create multiple instances of Border Gateway Protocol (BGP), Intermediate System-to-Intermediate System (IS-IS), Label Distribution Protocol (LDP), Open Shortest Path First (OSPF), Protocol Independent Multicast (PIM), Routing Information Protocol (RIP), and static routes by including statements at the [edit routing-instances *routing-instance-name* protocols] hierarchy level.

Each routing instance consists of sets of the following:

- Routing tables

- Interfaces that belong to these routing tables

- Routing option configurations

You can configure four types of routing instances:

Forwarding—Use this routing instance type for filter-based forwarding applications. For this instance type, there is no one-to-one mapping between an interface and a routing instance. All interfaces belong to the default instance `inet.0`.

L2VPN—Use this routing instance type for Layer 2 Virtual Private Network (VPN) implementations.

Nonforwarding—Use this routing instance type when a separation of routing table information is required. There is no corresponding forwarding table. All routes are installed into the default forwarding table. IS-IS instances are strictly nonforwarding instance types.

VRF—VPN routing and forwarding. Use this routing instance type for Layer 3 VPN implementations. For this instance type, there is a one-to-one mapping between an interface and a routing instance. Each VRF instance corresponds with a forwarding table. Routes on an interface go into the corresponding forwarding table.

For more detailed information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

This chapter describes the following tasks for configuring routing instances:

- Routing Instances Minimum Configuration on page 140

- Configure Multiple Instances of BGP on page 143

- Configure Multiple Instances of IS-IS on page 144

- Configure Multiple Instances of LDP on page 147
- Configure Multiple Instances of OSPF on page 148
- Configure Multiple Instances of PIM on page 152
- Configure Multiple Instances of RIP on page 153
- Configure an Instance on page 153
- Configure VPNs on page 154
- Configure an Instance Type on page 154
- Configure a Route Distinguisher (Address Extension) on page 155
- Configure Filter-Based Forwarding on page 155
- Configure Class of Service-Based Forwarding on page 157
- Configure Secondary VRF Import and Export Policy on page 158
- Configure Policy-based Export for Routing Instances on page 159
- Configure a VRF Table Label on page 164
- Configure a VRF Target on page 164
- Configure an OSPF Domain ID on page 164
- Configure a Route Limit for Routing Tables on page 168

To configure routing instances, include the following statements at the [edit routing-instances] hierarchy level of the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    description text;
    forwarding-options {
      family (inet | inet6 | mpls) {
        filter {
          input filter-name;
        }
      }
    }
    hash-key {
      family inet {
        layer-3;
        layer-4;
      }
      family mpls {
        label-1;
        label-2;
      }
    }
  }
}
```

```

helpers {
  bootp {
    description description-of-service;
    interface interface-group {
      description description-of-interface;
      maximum-hop-count number;
      minimum-wait-time seconds;
      no-listen;
      server [ addresses ]
    }
    maximum-hop-count number;
    minimum-wait-time seconds;
    server address < [ routing-instance routing-instance-name ] >;
  }
  domain {
    description description-of-service;
    server address;
    interface interface-name {
      description description-of-interface;
      no-listen;
      server address < [ routing-instance routing-instance-name ] >;
    }
  }
}
tftp {
  description description-of-service;
  server address < [ routing-instance routing-instance-name ] >;
  interface interface-name {
    description description-of-interface;
    no-listen;
    server address;
  }
}
traceoptions {
  file filename {
    files number;
    size bytes;
  }
  flag flag;
  level level;
}
}
sampling {
  disable;
  input {
    family inet {
      max-packets-per-second number;
      rate number;
      run-length number;
    }
  }
}
}

```

```

output {
  cflowd host-name {
    aggregation {
      autonomous-system;
      destination-prefix;
      protocol-port;
      source-destination-prefix {
        caida-compliant;
      }
      source-prefix;
    }
    autonomous-system-type (origin | peer);
    (local-dump | no-local-dump);
    port port-number;
    version format;
  }
  file {
    disable;
    filename filename;
    files number;
    size bytes;
    (stamp | no-stamp);
    (world-readable | no-world-readable);
  }
  port-mirroring {
    interface interface-name;
    next-hop address;
  }
}
tracoptions {
  file filename {
    files number;
    size bytes;
    (world-readable | no-world-readable);
  }
}
}
interface interface-name;
instance-type (forwarding | l2vpn | no-forwarding | vrf);
route-distinguisher (as-number:number | ip-address:number);
vrf-import [ policy-names ];
vrf-export [ policy-names ];
vrf-table-label;
vrf-target {
  export community-name;
  import community-name;
}

```

```

protocols {
  bgp {
    bgp-configuration;
  }
  isis {
    isis-configuration;
  }
  ldp {
    ldp-configuration;
  }
  ospf {
    ospf-configuration;
  }
  pim {
    pim-configuration;
  }
  rip {
    rip-configuration;
  }
}
routing-options {
  aggregate {
    defaults {
      aggregate-options;
    }
    route destination-prefix {
      policy policy-name;
      aggregate-options;
    }
  }
}
auto-export {
  (disable | enable);
  family {
    inet {
      multicast {
        (disable | enable);
        rib-group rib-group;
      }
      unicast {
        (disable | enable);
        rib-group rib-group;
      }
    }
  }
}
traceoptions {
  file name <replace> <size size> <files number> <no-stamp> <world-readable>;
  flag flag <flag-modifier> <disable>;
}
}
autonomous-system autonomous-system <loops number>;
confederation confederation-autonomous-system members autonomous-system;
fate-sharing {
  group group-name;
  cost value;
  from address [to address];
}
forwarding-table {
  export [ policy-names ];
}

```

```

generate {
  defaults {
    generate-options;
  }
  route destination-prefix {
    policy policy-name;
    generate-options;
  }
}
instance-export [ policy-names ];
instance-import [ policy-names ];
interface-routes {
  rib-group group-name;
}
martians {
  destination-prefix match-type <allow>;
}
maximum-routes route-limit <log-only | threshold value>;
multicast {
  scope scope-name {
    interface interface-name;
    prefix destination-prefix;
  }
  ssm-groups {
    addresses;
  }
}
options {
  syslog (level level | upto level);
}
resolution {
  tracefilter [ policy-names ];
  traceoptions {
    file name <replace> <size size> <files number> <no-stamp>
      <(world-readable | no-world-readable)>;
    flag flag <flag-modifier> <disable>;
  }
}
rib routing-table-name {
  aggregate {
    defaults {
      aggregate-options;
    }
    route destination-prefix {
      policy policy-name;
      aggregate-options;
    }
  }
  generate {
    defaults {
      generate-options;
    }
    route destination-prefix {
      policy policy-name;
      generate-options;
    }
  }
  martians {
    destination-prefix match-type <allow>;
  }
}

```

```

static {
  defaults {
    static-options;
  }
  rib-group group-name;
  route destination-prefix {
    lsp-next-hop {
      metric metric;
      preference preference;
    }
    next-hop;
    qualified-next-hop {
      metric metric;
      preference preference;
    }
    static-options;
  }
}
}
rib-groups {
  group-name {
    import-policy [ policy-names ;
    import-rib [ group-names ];
    export-rib group-name;
  }
}
route-record;
router-id address;
static {
  defaults {
    static-options;
  }
  rib-group group-name;
  route destination-prefix {
    lsp-next-hop {
      metric metric;
      preference preference;
    }
    next-hop;
    qualified-next-hop {
      metric metric;
      preference preference;
    }
    static-options;
  }
}
traceoptions {
  file name <replace> <size size> <files number> <no-stamp>
  <(world-readable | no-world-readable)>;
  flag flag <flag-modifier> <disable>;
}
}
}
}

```

Routing Instances Minimum Configuration

You can configure BGP, IS-IS, OSPF, PIM, and RIP routing instances.

BGP

BGP instances are supported only for VRF instance types. To configure a routing instance for BGP, you must include at least the following statements in the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type forwarding | l2vpn | no-forwarding | vrf;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
    vrf-table-label;
    protocols {
      bgp {
        bgp configuration;
      }
    }
  }
}
```

For more information about the BGP configuration statements, see “BGP Configuration Guidelines” on page 367. For more information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

IS-IS

To configure a routing instance for IS-IS, you must include at least the following statements in the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type forwarding | l2vpn | no-forwarding | vrf;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
    vrf-table-label;
    protocols {
      isis {
        isis configuration;
      }
    }
  }
}
```

For more information about the IS-IS configuration statements, see “IS-IS Configuration Guidelines” on page 181.

LDP

To create a routing instance for LDP, you must include at least the following statements in the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    instance-type (forwarding | l2vpn | no-forwarding | vrf);
    interface interface-name;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-export [ policy-names ];
    vrf-import [ policy-names ];
    vrf-table-label;
    protocols {
      ldp {
        ldp configuration;
      }
    }
  }
}
```

For more detailed information about configuring LDP, see the *JUNOS Internet Software Configuration Guide: MPLS Applications*.

LDP routing instances are used to support LDP over VPNs. For more detailed information about configuring multicast over VPNs, see *JUNOS Internet Software Configuration Guide: VPNs*.

OSPF

To configure a routing instance for OSPF, you must include at least the following statements in the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type forwarding | l2vpn | no-forwarding | vrf;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
    vrf-table-label;
    protocols {
      ospf {
        ospf configuration;
      }
    }
  }
}
```



Note

You can configure only one routing instance per logical interface.

For more information about the OSPF configuration statements, see “OSPF Configuration Guidelines” on page 237.

PIM

To create a routing instance for PIM, you must include at least the following statements in the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    instance-type (forwarding | l2vpn | no-forwarding | vrf);
    interface interface-name;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-export [ policy-names ];
    vrf-import [ policy-names ];
    vrf-table-label;
    protocols {
      pim {
        pim configuration;
      }
    }
  }
}
```

For more detailed information about configuring PIM, see the *JUNOS Internet Software Configuration Guide: Multicast*.

PIM routing instances are used to support multicast over VPNs. For more detailed information about configuring multicast over VPNs, see *JUNOS Internet Software Configuration Guide: VPNs*.

RIP

RIP instances are supported only for VPN routing and forwarding (VRF) instance types. This instance type provides support for Layer 3 VPNs. To configure a routing instance for RIP, you must include at least the following statements in the configuration:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type vrf;
    route-distinguisher (as-number:number | ip-address:number);
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
    vrf-table-label;
    protocols {
      rip {
        rip configuration;
      }
    }
  }
}
```

For more information about the RIP configuration statements, see “RIP Configuration Guidelines” on page 285. For more information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

Configure Multiple Instances of BGP

You can configure multiple instances of BGP and its configuration at the [edit routing-instances *routing-instance-name* protocols] hierarchy level. Multiple instances of BGP are primarily used for Layer 3 VPN support.

Currently, EBG (nonmultihop) peers are supported under the routing-instances hierarchy. EBG peering is established over one of the interfaces configured under the routing-instances hierarchy. Routes learned from the EBG peer are added to the instance-name.inet.0 table by default. You can configure import and export policies to control the flow of information in and out of the instance routing table.

For Layer 3 VPN support, configure BGP on the Provider Edge (PE) router to receive routes from the Customer Edge (CE) router and to send the instances' routes to the CE router if necessary. You can use multiple instances of BGP to maintain separate per-site forwarding tables for keeping VPN traffic separate on the PE router. For more detailed information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

You can configure import and export policies that allow the service provider to control and rate-limit traffic to and from the customer.

To configure multiple instances of BGP, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    interface so-1/1/1.0;
    interface so-1/1/1.1;
    instance-type vrf;
    route distinguisher (as-number:number | ip-address:number);
    protocols {
      bgp {
        group group-name {
          peer-as 01;
          type external;
          import route-name;
          export route-name;
          neighbor 10.0.0.1;
        }
      }
    }
  }
}
```

Configure Multiple Instances of IS-IS

You can use multiple instances of IS-IS for administrative separation and to support label-switched interfaces.

To configure multiple routing instances, perform the following tasks:

Configure the IS-IS default instance at the [edit protocols isis] hierarchy level with the statements needed for your network so that routes are installed in inet.0 and in the forwarding table. Make sure to include the routing table group. For more information, see “Configure the Default IS-IS Instance” on page 147 and “Configure a Routing Table Group” on page 151.

Configure an IS-IS routing instance for each additional OSPF routing entity, configuring the following:

- Interfaces

- Routing options

- IS-IS protocol statements belonging to that entity

- Routing table group

For more information, see “Configure the Routing Instances” on page 150.

Configure a routing table group to install routes from the routing instance into the inet.0 routing table. You can do this in two ways:

- Create a common routing table group so that:

 - Routes from the routing instances are installed in inet.0 and therefore installed in the forwarding table.

 - Routes from one router in a routing instance are forwarded to another router in the same routing instance.

- Create a routing table group with just the routing table from one instance and inet.0 to keep the routes from going to other instances.

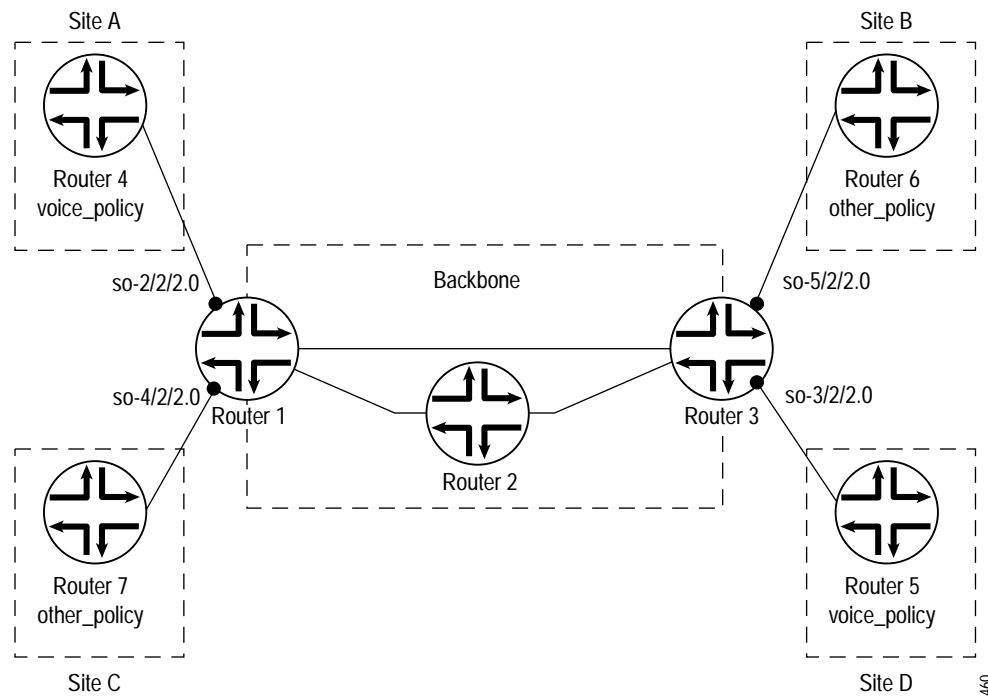
For more information, see “Configure a Routing Table Group” on page 151.

Create an export policy to export routes with a specific tag and to use that tag to export routes back into the instances. For more information, see “Configure the Default OSPF Instance” on page 152 and the *JUNOS Internet Software Configuration Guide: Policy Framework*.

Example: Configure Multiple Routing Instances

Figure 2 shows how you can use multiple instances of IS-IS to segregate traffic within a large network. The network consists of three administrative entities: `voice_policy`, `other_policy`, and the backbone or core. Each entity is composed of several geographically separate sites that are connected by the backbone and managed by the backbone entity.

Figure 2: Configuration for Multiple Routing Instances



Sites A and D belong to the `voice_policy` routing instance. Sites B and C belong to the `other_policy` instance. Router 1 and Router 3 at the edge of the backbone connect the routing instances. Each runs a separate IS-IS instance (one per entity).

Router 1 runs three IS-IS instances: one each for Site A (`voice_policy`), Site C (`other_policy`), and the backbone, otherwise known as the default instance. Router 3 also runs three IS-IS instances: one each for Site B (`other_policy`), Site D (`voice_policy`), and the backbone (default instance).

When Router 1 runs the IS-IS instances, the following occur:

- Routes from the default instance routing table are placed in the `voice_policy` and `other_policy` instance routing tables.

- Routes from the `voice_policy` routing instance are placed in the default instance routing table.

- Routes from the `other_policy` routing instance are placed in the default instance routing table.

Routes from the voice_policy routing instance do not enter the other_policy instance routing table.

Routes from the other_policy routing instance do not enter the voice_policy instance routing table.

Configure Router 1

The following sections describe how to configure Router 1 in the backbone entity with multiple routing instances.

Configure the Routing Instances

Configure the routing instances for voice-policy and other-policy. Use all routes learned from the routing tables in the routing table group common. Export routes tagged as belonging to the routing instance.

```
[edit]
routing-instances {
  voice-policy {
    interface so-2/2/2.0;
    protocols {
      isis {
        rib-group voice_to_inet;
        interface so-2/2/2.0;
      }
    }
    routing-options {...};
  }
  other-policy {
    interface so-4/2/2.0;
    protocols {
      isis {
        rib-group other_to_inet;
        interface so-4/2/2.0;
      }
    }
    routing-options {...};
  }
}
```

Configure a Routing Table Group

Configure the routing table group common to share routes with inet.0 (in the backbone entity), the voice-policy.inet.0, and the other-policy.inet.0 routing tables:

```
[edit]
routing-options {
  rib-groups {
    inet_to_voice_and_other {
      import-rib [inet.0 voice-policy.inet.0 other-policy.inet.0];
    }
  }
}
```

Configure the routing table group common to share routes with the inet.0 (in the backbone entity) and the voice-policy.inet.0 routing tables:

```
[edit]
routing-options {
  rib-groups {
    voice_to_inet {
      import-rib [inet.0 voice-policy.inet.0];
    }
  }
}
```

Configure the routing table group common to share routes with the inet.0 (in the backbone entity) and the other-policy.inet.0 routing tables:

```
[edit]
routing-options {
  rib-groups {
    other_to_inet {
      import-rib [inet.0 other-policy.inet.0];
    }
  }
}
```

Configure the Default IS-IS Instance

Configure the default IS-IS instance so that the routes learned from the routing instances are installed in inet.0 and to export the tagged routes from voice-policy and other-policy:

```
[edit]
protocols {
  isis {
    rib-group inet_to_voice_and_other;
    interface so-2/2/2.0;
    interface so-4/2/2.0;
  }
}
```

Configure Router 3

The configuration for Router 3 is the same as for Router 1 except that the interface names might differ. In this topology, the interface so-5/2/2.0 belongs to other-policy and so-3/2/2.0 belongs to voice-policy.

Configure Multiple Instances of LDP

LDP is a protocol used to distribute labels in an MPLS-enabled network.

LDP instances are used to distribute labels from a provider edge (PE) router to a customer edge (CE) router. LDP instances in a VPN are useful in carrier-of-carrier networks, where data is transmitted between two or more telecommunications carrier sites across a core provider network. Each carrier may want to restrict internet routes strictly to the PE routers.

An advantage of using LDP instances within a VPN is that a full-mesh IGBP is not required between the PE and CE routers. A router ID is required to configure an instance of LDP.

To configure multiple instances of LDP, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type vrf;
    protocols {
      ldp {
        ldp-configuration;
      }
    }
  }
}
```

For more detailed information about configuring LDP, see the *JUNOS Internet Software Configuration Guide: MPLS Applications*. For more detailed information about configuring LDP over VPNs, see *JUNOS Internet Software Configuration Guide: VPNs*.

Configure Multiple Instances of OSPF

To configure multiple routing instances of OSPF, perform the following tasks:

Configure the OSPF default instance at the [edit protocols ospf] hierarchy level with the statements needed for your network so that routes are installed in inet.0 and in the forwarding table. Make sure to include the routing table group. For more information, see “Configure a Routing Table Group” on page 151 and “Configure the Default OSPF Instance” on page 152.

Configure an OSPF routing instance for each additional OSPF routing entity, configuring the following:

Interfaces

Routing options


OSPF protocol statements belonging to that entity

Routing table group

For more information, see “Configure the Routing Instances” on page 150.

Configure a routing table group to install routes from the default route table, inet.0, into a routing instance’s route table.

Configure a routing table group to install routes from a routing instance into the default route table, inet.0.



Note Non-forwarding routing instances do not have forwarding tables that correspond to their routing tables.

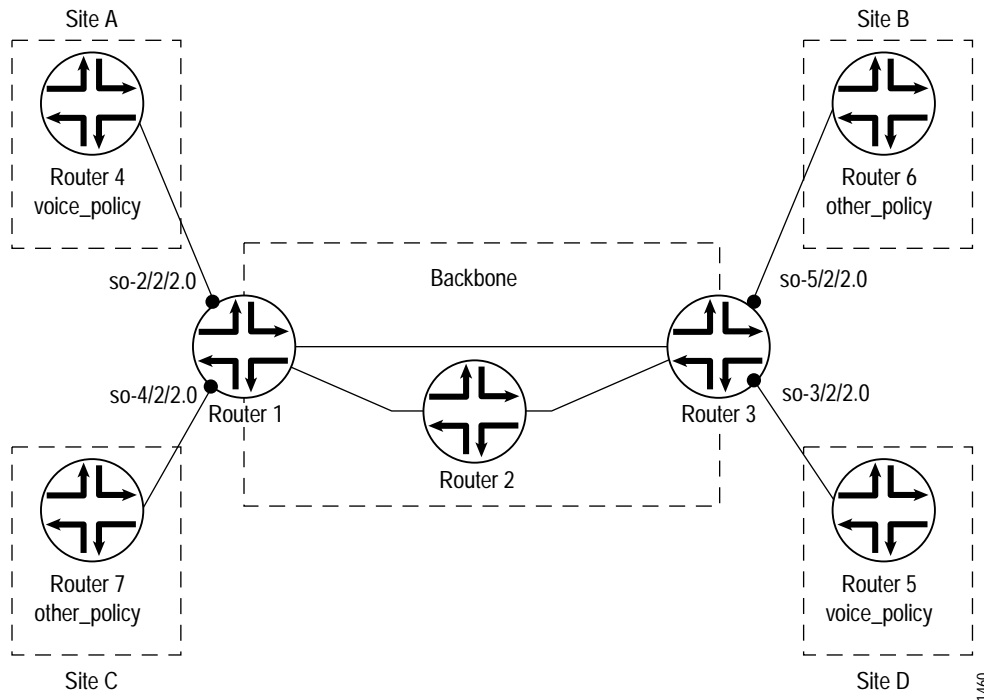
For more information, see “Configure a Routing Table Group” on page 151.

Create an export policy to export routes with a specific tag and to use that tag to export routes back into the instances. For more information, see “Configure the Default OSPF Instance” on page 152 and the *JUNOS Internet Software Configuration Guide: Policy Framework*.

Example: Configure Multiple Routing Instances

Figure 3 shows how you can use multiple routing instances of OSPF to segregate prefixes within a large network. The network consists of three administrative entities: voice_policy, other_policy, and the default routing instance. Each entity is composed of several geographically separate sites that are connected by the backbone and managed by the backbone entity.

Figure 3: Configuration for Multiple Routing Instances



Sites A and D belong to the voice_policy routing instance. Sites B and C belong to the other_policy instance. Router 1 and Router 3 at the edge of the backbone connect the routing instances. Each runs a separate OSPF instance (one per entity).

Router 1 runs three OSPF instances: one each for Site A (voice_policy), Site C (other_policy), and the backbone, otherwise known as the default instance. Router 3 also runs three OSPF instances: one each for Site B (other_policy), Site D (voice_policy), and the backbone (default instance).

When Router 1 runs the OSPF instances, the following occur:

Routes from the default instance routing table are placed in the voice_policy and other_policy instance routing tables.

Routes from the voice_policy routing instance are placed in the default instance routing table.

Routes from the other_policy routing instance are placed in the default instance routing table.

Routes from the voice_policy routing instance do not enter the other_policy instance routing table.

Routes from the other_policy routing instance do not enter the voice_policy instance routing table.

Configure Router 1

The following sections describe how to configure Router 1 in the backbone entity with multiple routing instances.

Configure the Routing Instances

Configure the routing instances for voice-policy and other-policy:

```
[edit]
routing-instances {
  voice-policy {
    interface so-2/2/2.0;
    protocols {
      ospf {
        rib-group voice_to_inet;          #Places routes from voice-policy.inet.0 into inet.0
        area 0.0.0.0 {
          interface so-2/2/2.0;
        }
      }
    }
  }
  routing-options {...};
}
```

```

other-policy {
  interface so-4/2/2.0;
  protocols {
    ospf {
      rib-group other_to_inet;      # Places routes from other-policy.inet.0 into inet.0
      area 0.0.0.0 {
        interface so-4/2/2.0;
      }
    }
  }
  routing-options {...};
}

```

Configure a Routing Table Group

Configure the routing table group `inet_to_voice_and_others` to take routes from `inet.0` (default routing table) and place them in the `voice-policy.inet.0` and `other-policy.inet.0` routing tables:

```

[edit]
routing-options {
  rib-groups {
    inet_to_voice_and_other {
      import-rib [inet.0 voice-policy.inet.0 other-policy.inet.0];
    }
  }
}

```

Configure the routing table group `voice_to_inet` to take routes from `voice-policy.inet.0` and place them in the `inet.0` default routing table:

```

[edit]
routing-options {
  rib-groups {
    voice_to_inet {
      import-rib [inet.0 voice-policy.inet.0];
    }
  }
}

```

Configure the routing table group `other_to_inet` to take routes from `other-policy.inet.0` and place them in the `inet.0` default routing table:

```
[edit]
routing-options {
  rib-groups {
    other_to_inet {
      import-rib [inet.0 other-policy.inet.0];
    }
  }
}
```

Configure the Default OSPF Instance

Configure the default OSPF instance:

```
[edit]
protocols {
  ospf {
    rib-group inet_to_voice_and_other;           # Place prefixes from inet.0 into
    area 0.0.0.0 {                               # voice-policy.inet.0 and
      interface so-2/2/2.0;                       # other-policy.inet.0
      interface so-4/2/2.0;
    }
  }
}
```

Configure Router 3

The configuration for Router 3 is the same as for Router 1 except that the interface names might differ. In this topology, the interface `so-5/2/2.0` belongs to `other-policy` and `so-3/2/2.0` belongs to `voice-policy`.

Configure Multiple Instances of PIM

PIM instances are supported only for VRF instance types. You can configure multiple instances of PIM to support multicast over VPNs.

To configure multiple instances of PIM, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type vrf;
    protocols {
      pim {
        pim-configuration;
      }
    }
  }
}
```

For more detailed information about configuring PIM, see the *JUNOS Internet Software Configuration Guide: Multicast*. For more detailed information about configuring multicast over VPNs, see *JUNOS Internet Software Configuration Guide: VPNs*.

Configure Multiple Instances of RIP

RIP instances are supported only for VRF instance types. You can configure multiple instances of RIP for VPN support only. You can use RIP in the customer edge-provider edge (CE-PE) environment to learn routes from the CE router and to propagate the PE router's instance routes in the CE router.

RIP routes learned from neighbors configured under any instance hierarchy are added to the instance's routing table—`instance-name.inet.0`.

RIP does not support routing table groups, therefore it cannot import routes into multiple tables like the OSPF protocol does.

To configure multiple instances of RIP, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name;
    instance-type vrf;
    protocols {
      rip {
        interface interface-name;
        neighbor 1.0.0.1;
      }
    }
  }
}
```

Configure an Instance

You can create multiple instances of BGP, IS-IS, OSPF, RIP, and static routes by including statements at the `[edit routing-instances routing-instance-name protocols]` hierarchy level.

Each routing instance consist of:

- A set of routing tables
- A set of interfaces that belong to these routing tables
- A set of routing option configurations

Each routing instance has a unique name and a corresponding IP unicast table. For example, if you configure a routing instance with the name `my-instance`, its corresponding IP unicast table will be `my-instance.inet.0`. All routes for `my-instance` are installed into `my-instance.inet.0`.

Configure global routing options and protocols for the default instance by including statements at the `[edit protocols]` and `[edit routing-options]` hierarchy levels. Routes are installed into the default routing instance `inet.0` by default, unless a routing instance is specified.

For details about specifying interfaces, see the *JUNOS Internet Software Configuration Guide: Interfaces and Class of Service* .

To configure a routing instance, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name ;
    instance-type (forwarding | l2vpn | no-forwarding | vrf);
    route-distinguisher (as-number:number | ip-address:number);
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
    vrf-table-label;
    protocols {
      bgp {
        bgp-configuration;
      }
      ospf {
        ospf-configuration;
      }
      rip {
        rip-configuration;
      }
    }
    routing-options {...};
  }
}
```

Configure VPNs

To configure Virtual Private Networks (VPNs), see the *JUNOS Internet Software Configuration Guide: VPNs*.

Configure an Instance Type

There are four types of routing instances that you configure at the [edit routing-instances routing-instance-name instance-type] hierarchy level:

Forwarding—Use this routing instance type for filter-based forwarding applications. For this instance type, there is no one-to-one mapping between an interface and a routing instance. All interfaces belong to the default instance inet.0.

L2VPN—Use this routing instance type for Layer 2 VPN implementations.

Nonforwarding—Use this routing instance type when a separation of routing table information is required. There is no corresponding forwarding table. All routes are installed into the default forwarding table. IS-IS instances are strictly nonforwarding instance types.

VRF—VPN routing and forwarding. Use this routing instance type for Layer 3 VPN implementations. For this instance type, there is a one-to-one mapping between an interface and a routing instance. Each VRF instance corresponds with a forwarding table. Routes on an interface go into the corresponding forwarding table.

To configure a routing instance type, include the `instance-type` statement at the [edit `routing-instances`] hierarchy level:

```
[edit]
routing-instances {
  routing-instance-name {
    interface interface-name ;
    instance-type (forwarding | l2vpn | no-forwarding | vrf);
  }
}
```

Configure a Route Distinguisher (Address Extension)

Each routing instance must have a unique route distinguisher associated with it. The route distinguisher is used to place bounds around a VPN so the same IP address prefixes can be used in different VPNs without having them overlap.

We recommend that you use a unique route distinguisher for each routing instance that you configure. Although you could use the same route distinguisher on all PE routers for the same VPN, if you use a unique route distinguisher, you can determine the CE router from which a route originated.

To configure a route distinguisher, include the `route-distinguisher` statement at the [edit `routing-instances routing-instance-name`] hierarchy level:

```
[edit routing-instances]
routing-instance-name {
  route-distinguisher ( as-number:number | ip-address:number );
}
```

The route distinguisher is a 6-byte value that you can specify in one of the following formats:

as-number:number, where *as-number* is your assigned AS number (a 2-byte value) and *number* is any 4-byte value. The AS number can be in the range of 1 through 65535.

ip-address:number, where *ip-address* is an IP address in your assigned prefix range (a 4-byte value) and *number* is any 2-byte value. The IP address can be in the range of 0 through 4294967295 ($2^{32} - 1$).

Configure Filter-Based Forwarding

You can create a filter to classify packets to determine their forwarding path within a router. Use filter-based forwarding to redirect traffic for analysis.

Use filter-based forwarding for service provider selection when customers have Internet connectivity provided by different ISPs yet share a common access layer. When a shared media (such as a cable modem) is used, a mechanism on the common access layer looks at Layer 2 or Layer 3 addresses and distinguishes between customers. You can use filter-based forwarding when the common access layer is implemented using a combination of Layer 2 switches and a single router.

With filter-based forwarding, all packets received on an interface are considered. Each packet passes through a filter that has match conditions. If the match conditions are met for a filter and you have created a routing instance, filter-based forwarding is applied to a packet. The packet is forwarded based on the next hop specified in the routing instance. For static routes, the next hop can be a specific LSP. For more information about static route configuration statements, see “Configure Static Routes” on page 41. For more information about configuring LSPs, see *JUNOS Internet Software Configuration Guide: MPLS Applications*.

To configure filter-based forwarding:

Create a match filter on an ingress router. To specify a match filter, include the filter *filter-name* statement at the [edit firewall] hierarchy level. For more information about creating a match filter for packet forwarding, see the *JUNOS Internet Software Configuration Guide: Policy Framework*. A packet that passes through the filter is compared against a set of rules to classify it and to determine its membership in a set. Once classified, the packet is forwarded to a routing table specified in the accept action in the filter description language. The routing table then forwards the packet to the next hop that corresponds to the destination address entry in the table.

Create routing instances that specify the routing table(s) to which a packet is forwarded, and the destination to which the packet is forwarded at the [edit routing-instances] hierarchy level. For example:

```
[edit]
routing-instances {
  routing-table-name1 {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 nexthop 10.0.0.1;
      }
    }
  }
  routing-table-name2 {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 nexthop 10.0.0.2;
      }
    }
  }
}
```

Create a routing table group that adds interface routes to the forwarding routing instances used in FBF, as well as to the default routing instance inet.0. The purpose of this part of the configuration is to resolve the routes installed in the routing instances to directly-connected next hops on that interface. Create the routing table group at the [edit routing-options] hierarchy level. For example, the following configuration installs interface routes into the default routing instance inet.0 as well as two forwarding routing instances routing-table-name1.inet.0 and routing-table-name2.inet.0:

```
[edit]
routing-options {
  interface-routes {
    rib-group inet group-name;
  }
  rib-groups {
    group-name {
      import-rib [inet.0 routing-table-name1.inet.0 routing-table-name2.inet.0];
    }
  }
}
```



Specify inet.0 as one of the routing instances that the interface routes will be imported into. If the default instance inet.0 is not specified, interface routes will not be imported into the default routing instance.

Configure Class of Service-Based Forwarding

Class of service-based forwarding allows you to control the next hop selection based on a packet's class of service or IP precedence. It allows path selection based on a multifield classifier.

To configure class of service-based forwarding:

Create a routing policy at the [edit policy-options] hierarchy level to limit the configuration so that routes matching the route filter will be subject to the CoS next-hop mapping specified in my-cos-map:

```
[edit]
policy-options {
  policy-statement my-cos-forwarding {
    from {
      route-filter ...;
    }
    then {
      cos-next-hop-map my-cos-map;
    }
  }
}
```

Create a CoS next-hop map. To specify a CoS next-hop map, include the cos-next-hop-map statement at the [edit class-of-service] hierarchy level. For more information about creating a CoS next-hop map, see the *JUNOS Internet Software Configuration Guide: Interfaces and Class of Service*.

Specify the exporting of the routes to the forwarding table at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  forwarding-table {
    export my-cos-forwarding;
  }
}
```

Specify a static route that has multiple next hops for load balancing at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  static {
    route 12.1.1.1/32 {
      next-hop [3.1.1.2 3.1.1.4 3.1.1.6 3.1.1.8];
    }
  }
}
```

Configure Secondary VRF Import and Export Policy

You configure a VPN routing and forwarding instance (VRF) so that routes received from the provider edge-provider edge (PE-PE) session (in the default instance) can be imported into any of an instance's VRF secondary routing tables. Importing depends on defined policies. Routes to be exported should pass through the policies listed in the export list.

To configure secondary VRF import and export policies, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    instance-type vrf;
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
  }
}
policy-options {
  policy-statement policy-name {
    from community community-name;
    then accept;
  }
}
```

For more detailed information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

Configure Policy-based Export for Routing Instances

Configuring policy-based export simplifies the process of exchanging route information between routing instances.

Exporting routing information between routing instances typically is accomplished by configuring separate routing table groups for each instance. The use of policy-based export reduces the configuration needed for exporting routes between multiple routing instances by eliminating the configuration of separate routing table groups for each instance.

There are two cases in which policy-based export is particularly useful:

Overlapping VPNs—VPN configurations in which more than one VRF has the same route target

Nonforwarding instances—Multi-level IGPs using multiple routing instances



Note

The instance-export and instance-import statements are not valid for VRF instances. The auto-export statement is valid for VRF and non-VRF instances. The instance-import statement automatically enables auto-export for non-VRF instances.

For detailed information about configuring overlapping VPNs and nonforwarding instances, see the *JUNOS Internet Software Configuration Guide: VPNs*.

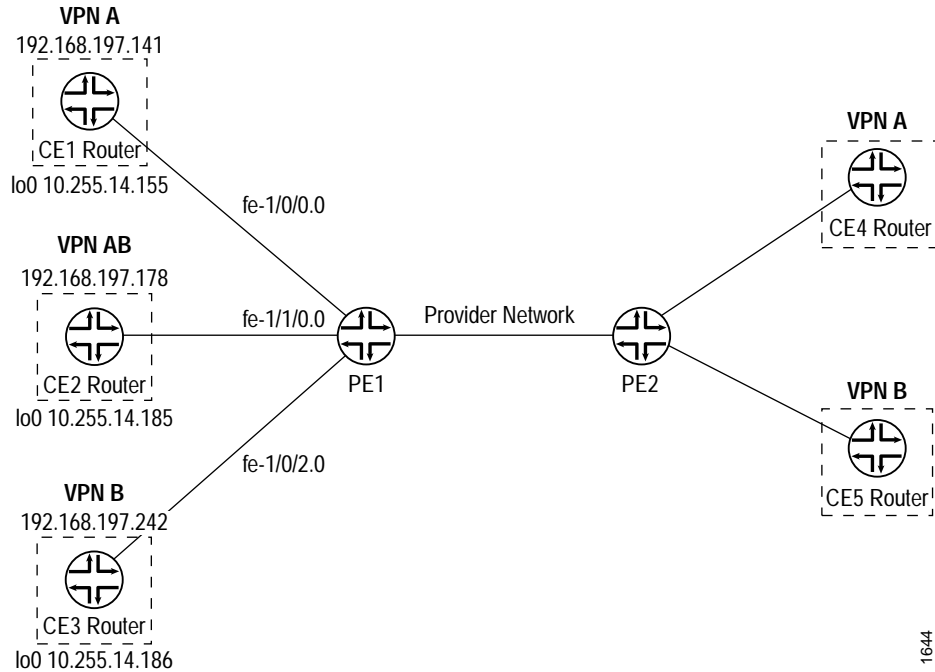
The following examples illustrate the use of both types of policy-based export.

Example: Configure Policy-based Export for an Overlapping VPN

In Layer 3 VPNs, a CE router is often a member of more than one VPN. Figure 4 illustrates the topology for the configuration example in this section. The configurations in this section illustrate local connectivity between CE routers connected to the same PE router using BGP.

The configuration statements enable the VPN AB Router CE2 to communicate with the VPN A Router CE1 and the VPN B Router CE3, both directly connected to the Router PE1. VPN routes that originate from the remote PE routers (the PE2 router, in this case) are placed in a global Layer 3 VPN routing table (bgp.l3vpn.inet.0) and routes with appropriate route targets are imported into the routing tables, as dictated by the VRF import policy configuration.

Figure 4: Configuration of Policy-based Export for Overlapping VPN



16/44

Configure Router PE1

This section describes how to configure Router PE1 in the backbone entity for this overlapping VPN by means of policy-based export.

Configure the Routing Instances

Configure the routing instances for VPN-A, VPN-AB, VPN-B.

```
[edit]
routing-instances {
  VPN-A {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.175:3;
    vrf-export A-out;
    vrf-import A-in;
    routing-options {
      auto-export;
      static {
        route 1.1.1.1/32 next-hop fe-1/0/0.0;
        route 1.1.1.2/32 next-hop fe-1/0/0.0;
      }
    }
  }
}
```

```

VPN-AB {
  instance-type vrf;
  interface fe-1/1/0.0;
  route-distinguisher 10.255.14.175:9
  vrf-export AB-out;
  vrf-import AB-in;
  routing-options {
    auto-export;
    static {
      route 1.1.3.1/32 next-hop fe-1/1/0.0;
      route 1.1.3.2/32 next-hop fe-1/1/0.0;
    }
  }
}
VPN-B {
  instance-type vrf;
  interface fe-1/0/2.0;
  route-distinguisher 10.255.14.175:9
  vrf-export B-out;
  vrf-import B-in;
  routing-options {
    auto-export;
    static {
      route 1.1.2.1/32 next-hop fe-1/0/2.0;
      route 1.1.2.2/32 next-hop fe-1/0/2.0;
    }
  }
}
}

```

Configure Router PE2

The configuration for Router PE2 is the same as that for Router PE1; however, the interface names might differ.

Example: Configure Policy-based Export for a Nonforwarding Instance

This example shows how to use the `instance-import` and `instance-export` statements to control route export between multiple instances. This is equivalent to using the `vrf-import` and `vrf-export` statements for VPNs, except these are with nonforwarding instances, not VRF instances.

There are two nonforwarding instances: `data` and `voice`. The following is the configuration for a PE router.

Configure the Routing Instances

Configure the routing instances for data and voice.

```
[edit]
routing-instances {
  data {
    instance-type no-forwarding;
    interface t3-0/1/3.0;
    routing-options {
      instance-import data-import;
      auto-export;
    }
    protocols {
      ospf {
        export accept;
        area 0.0.0.0 {
          interface all;
        }
      }
    }
  }
  voice {
    instance-type no-forwarding;
    interface t3-0/1/0.0;
    routing-options {
      instance-import voice-import;
      auto-export;
    }
    protocols {
      ospf {
        export accept;
        area 0.0.0.0 {
          interface all;
        }
      }
    }
  }
}
```

Configure Policies

Configure a master policy:

```
[edit]
policy-options {
  policy-statement {
    master-import {
      term a {
        from instance master;
        then {
          tag 11;
          accept;
        }
      }
      term b {
        from instance data;
        then {
          tag 10;
          accept;
        }
      }
    }
  }
}
```

Configure policies for each instance:

```
[edit]
policy-options {
  policy-statement {
    data-import {
      term a {
        from {
          instance master;
          tag 10;
          then accept;
        }
      }
      term b {
        then reject;
      }
    }
    voice-import {
      term a {
        from {
          instance master;
          protocol ospf;
          tag 11;
        }
      }
      term b {
        then reject;
      }
    }
  }
}
```

Configure a VRF Table Label

You configure a separate label for each VRF to provide double lookup and egress filtering.

To configure a label for a VRF, include the following statements:

```
[edit]
routing-instances {
  routing-instance-name {
    instance-type vrf;
    vrf-import [ policy-names ];
    vrf-export [ policy-names ];
    vrf-table-label;
  }
}
```

For more detailed information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

Configure a VRF Target

Configuring a Virtual Routing and Forwarding (VRF) target provides a configurable community within a VRF routing instance and allows a single policy for import and a single policy for export to replace the per-VRF policies for every community.

To configure a VRF target, include the `vrf-target` statement at the `[edit routing-instances routing-instance-name]` hierarchy level. Use the `import` and `export` options to specify the allowed communities to accept from neighbors and to send to neighbors:

```
[edit]
routing-instances {
  routing-instance-name {
    vrf-target {
      export community-name;
      import community-name;
    }
  }
}
```

For more detailed information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

Configure an OSPF Domain ID

For most OSPF configurations involving Layer 3 VPNs, you do not need to configure an OSPF domain ID. However, for a Layer 3 VPN connecting multiple OSPF domains, configuring OSPF domain IDs can help you to control LSA translation (for Type-3 and Type-5 LSAs) between the OSPF domains and back door paths. The default OSPF domain ID is 0.0.0.0. Each VRF table in a PE router associated with an OSPF instance is configured with the same OSPF domain ID.

Without the use of domain IDs, there is no method to identify which domain the routes originated from after the OSPF routes are distributed into BGP routes and advertised across the BGP VPN backbone. Distinguishing from which OSPF domain that a route originated allows classification of routes as Type-3 LSAs or Type-5 LSAs.

You accomplish this by:

1. Specifying an OSPF domain ID in the BGP extended community ID
2. Setting a route type
3. Configuring a VRF export policy to explicitly attach the outbound extended community ID onto outbound routes
4. Defining a community with members that possess the community ID

For more detailed information about configuring export policies, see the *JUNOS Internet Software Configuration Guide: Policy Framework*.

This extended community ID can then be carried across the BGP VPN backbone. When the route is redistributed back as an OSPF route on the PE router and advertised to the CE near the destination, the domain ID identifies which domain the route originated. The route is then propagated as either Type-3 LSA or Type-5 LSA.

When a PE router receives a route, it redistributes and advertises the route either as a Type-3 LSA or as a Type-5 LSA, depending on the following:

If the receiving PE router sees a Type-3 route with a matching domain ID, the route is redistributed and advertised as a Type-3 LSA.

If the receiving PE router sees a Type-3 route without a domain ID (the extended attribute field of the route's BGP update does not include a domain ID), the route is redistributed and advertised as a Type-3 LSA.

If the receiving PE router sees a Type-3 route with a non-matching domain ID, the route is redistributed and advertised as a Type-5 LSA.

If the receiving PE router sees a Type-3 route with a domain ID, but the router does not have a domain ID configured, the route is redistributed and advertised as a Type-5 LSA.

If the receiving PE router sees a Type-5 route, the route is redistributed and advertised as a Type-5 LSA, regardless of the domain ID.

To configure an OSPF domain ID, include the domain-id statement at the [edit routing-instances *routing-instance-name* protocols ospf] hierarchy level:

```
[edit routing-instances routing-instance-name protocols]
ospf {
  domain-id domain ID;
}
```

You can set a VPN tag for the OSPF external routes generated by the PE router. This is used to prevent looping when a domain ID is used as an alternate route preference. By default, this tag is automatically calculated and needs no configuration. To configure the domain VPN tag for Type 5 LSAs, include the domain-vpn-tag *number* statement at the [edit routing-instances *routing-instance-name* protocols ospf] hierarchy level:

```
[edit routing-instances routing-instance-name protocols]
ospf {
  domain-vpn-tag number;
}
```

The range is 1 through 4,294,967,295. If you set VPN tags manually, you must set the same value for all PE routers in the VPN.

To set the route type, include the route-type-community statement at the [edit routing-instances *routing-instance-name* protocols ospf] hierarchy level:

```
[edit routing-instances]
  routing-instance-name {
    protocols {
      ospf{
        route-type-community (iana | vendor);
      }
    }
  }
```

To configure an export policy to attach the extended community ID onto outbound routes, include the community statement at the [edit policy-options policy-statement *policy-name* term *term-name* then] hierarchy level:

```
[edit policy-options]
  policy-statement {
    policy-name {
      term term-name {
        from {
          match-conditions;
        }
        then {
          action;
        }
      }
    }
  }
```

To define the members of a community, include the community statement at the [edit policy-options] hierarchy level:

```
[edit policy-options]
  community name {
    members [ community-ids ];
  }
```

Example: Configure OSPF Domain ID

Configure an OSPF domain ID and configure an export policy to tag the extended community ID onto outgoing routes:

```
[edit]
routing-instances {
  CE_A {
    instance-type vrf;
    interface ge-0/1/0.0;
    route-distinguisher 1:100;
    vrf-import vrf_import_routes;
    vrf-export vrf_export_routes;
    protocols {
      ospf {
        domain-id 1.1.1.1;
        route-type-community vendor;
        export vrf_import_routes;
        area 0.0.0.0 {
          interface ge-0/1/0.0;
        }
      }
    }
  }
}
policy-options {
  policy-statement vrf_export_routes {
    term a {
      from protocol ospf;
      then {
        community add export_target;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
}
community export_target members [ target:1:100 domain-id:1.1.1.1:0];
}
```

Configure a Route Limit for Routing Tables

A route limit sets an upper limit for the number of prefixes installed in routing tables. You can, for example, use a route limit to limit the number of routes received from the CE router in a VPN. A route limit applies only to dynamic routing protocols, not to static or interface routes.

To configure a route limit, include the `maximum-routes` statement at the `[edit routing-instances routing-instance-name routing-options]` hierarchy level:

```
[edit routing-instances routing-instance-name routing-options]  
maximum-routes route-limit <log-only | threshold value>;
```

There are two modes for route limits: advisory and mandatory. An advisory limit triggers warnings. A mandatory limit rejects any additional routes after the threshold is reached.



Caution

Application of a route limit may result in unpredictable dynamic route protocol behavior. For example, once you hit the limit and start rejecting routes, BGP will not necessarily try to reinstall the rejected routes once the number of routes drops back below the limit. BGP sessions may need to be cleared.

For more detailed information about configuring VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.