

APRICOT 2006

Introducing MPLS L3VPNs to Existing IP Networks

Ariff Premji
premji@juniper.net
Feb 2006



Workshop Objective

- Carriers already have IP networks and may want to take advantage of 2547 VPNs for certain applications or customers
- Use tools that are available to ease the migration to MPLS VPNs while maintaining existing IP network
- IP and MPLS PE nodes co-existing in the same network

Agenda

- Theory
 - Brief Overview on RFC2547bis MPLS VPNs
- Hands-on Lab
 - Bring up IGP (with support for TED)
 - Provision RSVP signaled LSPs between all PE routers
 - Migration Step I
 - Prepare MPLS VPN framework - including BGP RRs
 - Migration Step II
 - Pre-configure RIB group solution to accommodate migration
 - Identify VPN candidates and roll them into VPNs
 - Migrations Step III
 - Remove RIB groups

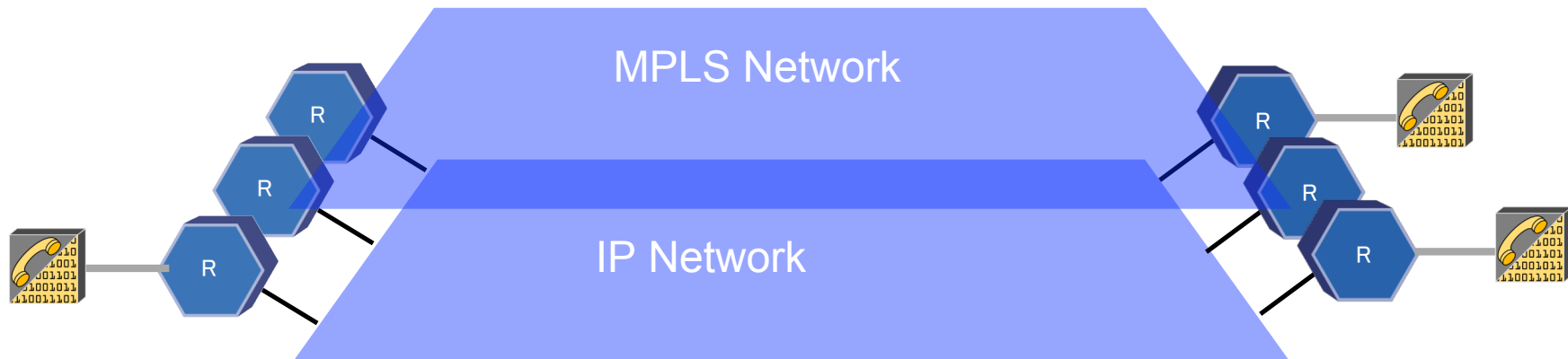
Existing IP Network

- Service provider has an existing IP network
- Desire to introduce MPLS services in this network



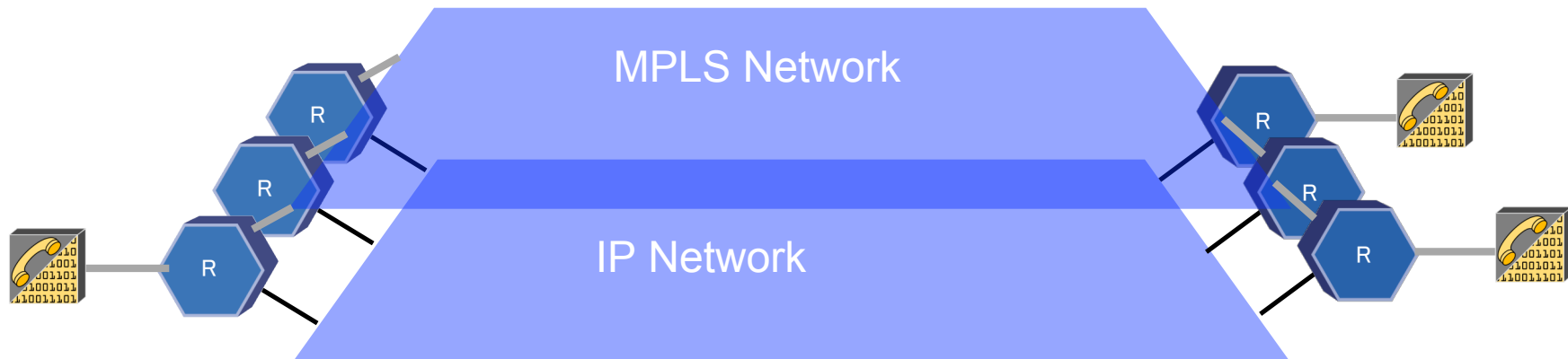
Overlaying MPLS

- Service provider has an existing IP network
- Desire to introduce MPLS services in this network
- Create a MPLS 'plane'



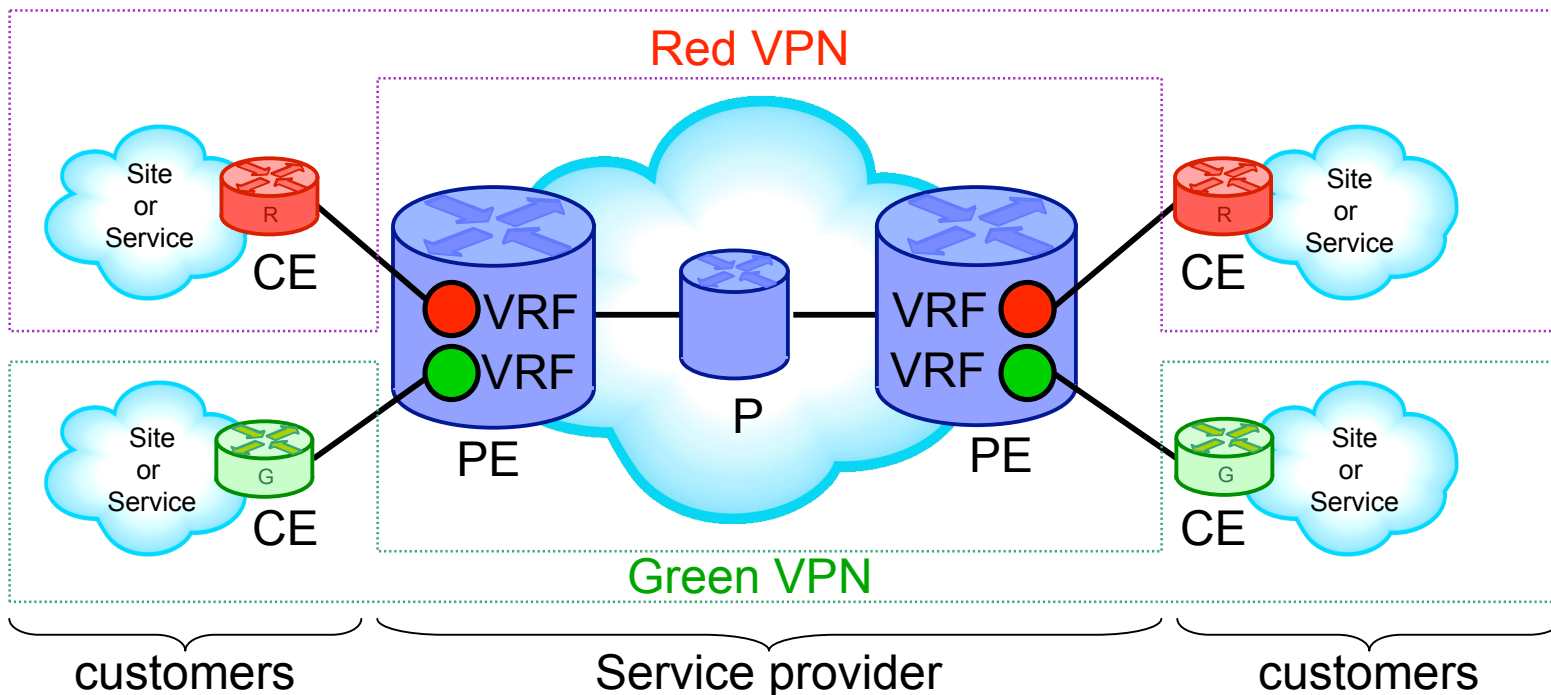
Overlaying MPLS

- Service provider has an existing IP network
- Desire to introduce MPLS services in this network
- Create a MPLS 'plane'
- Packets will have access to both the IP and MPLS planes
- Not all routers may participate in this new design



VPN Model (RFC 2547)

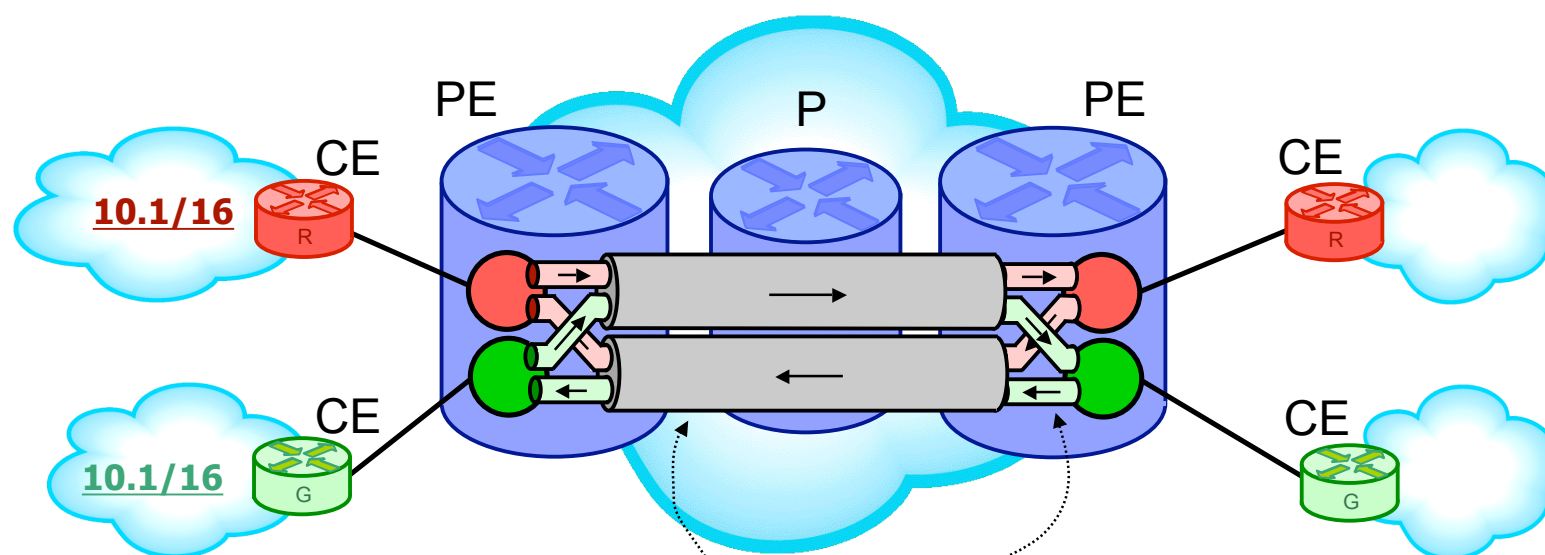
MPLS VPNs provide protection and isolation of traffic and routing information across a common IP infrastructure.



CE Customer Edge
PE Provider Edge
P Provider core

VRF VPN Routing / Forwarding table
VPN Virtual Private Network

VPN tunnels using MPLS LSPs example



BASE TUNNEL

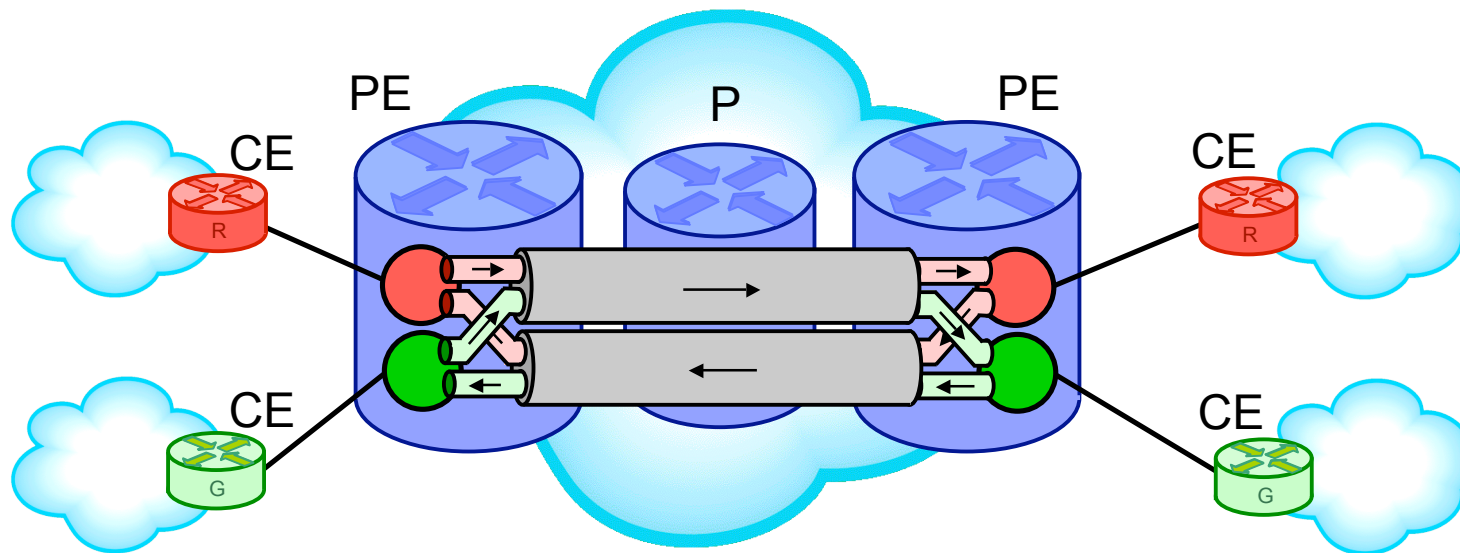
- From PE to PE
- Signaled using LDP / RSVP-TE

STACKED TUNNEL

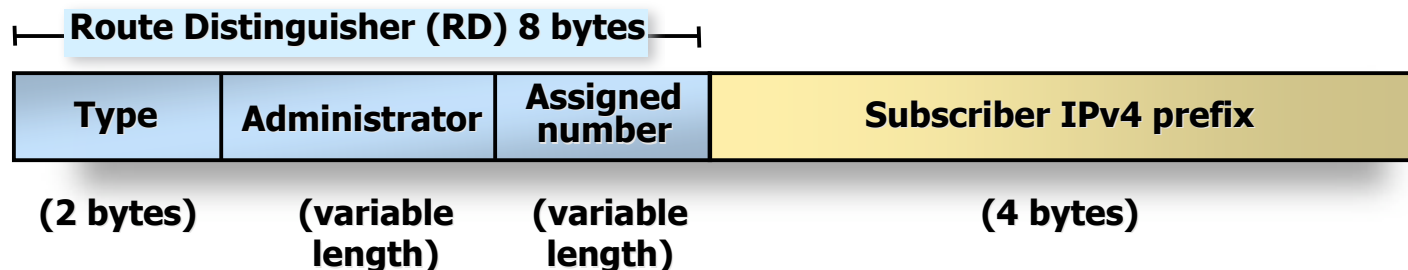
- Also known as VPN tunnel / BGP tunnel
- From VRF to VRF
- Stacked on base tunnel
- Signaled using MP-BGP + extensions
- Not visible to P routers

VRFs

- Each VRF is populated with:
 - Routes received from directly connected CE sites associated with the VRF
 - Routes received from other PE routers with acceptable BGP attributes
- Only the VRF associated with a site is consulted for packets from that site

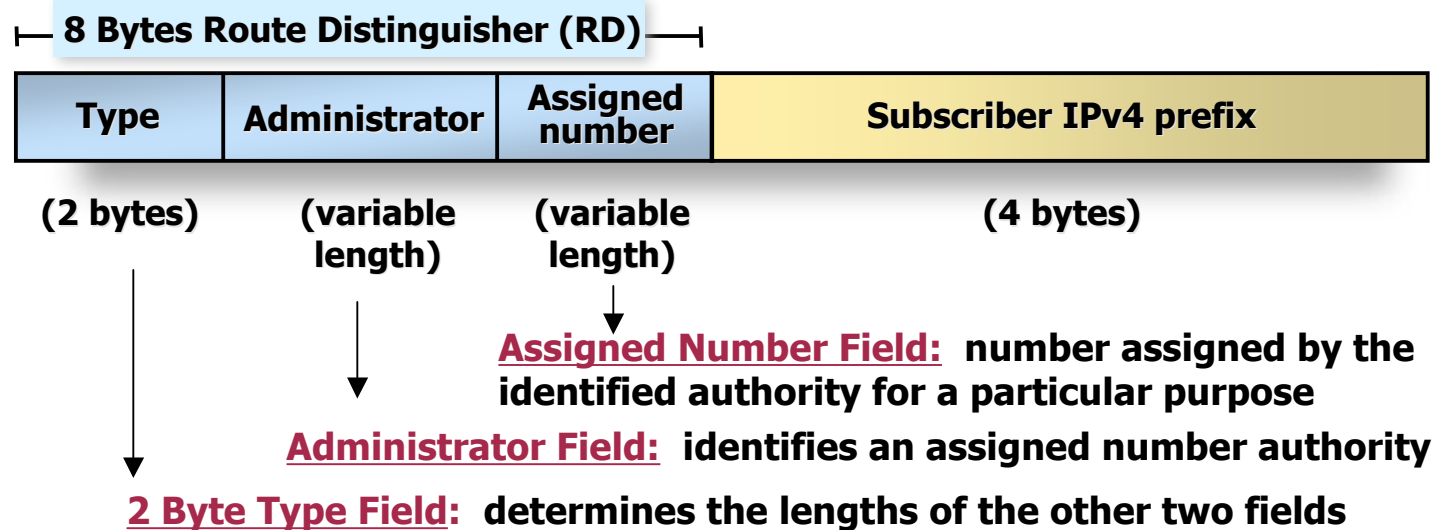


VPN-IPv4 Address Family



- VPN-IPv4 address family
 - BGP-4 address family identifier
 - Route Distinguisher (RD) + Subscriber IPv4 prefix
 - Route distinguisher disambiguates IPv4 addresses
 - Supports the private IP address space
 - Allows SP to administer its own “numbering space”
- VPN-IPV4 addresses are distributed by MP-BGP
 - Uses ‘Multiprotocol Extensions for BGP4’ (RFC 2283)
- VPN-IPV4 addresses are used only in the control plane

VPN-IPv4 Address Family



- Two values are defined for Type Field: 0 and 1
 - Type 0: Adm Field = 2 bytes, AN Field = 4 bytes
 - Adm field must contain an Autonomous System Number (ASN) from IANA
 - AN field is a number assigned by SP
 - Type 1: Adm Field = 4 bytes, AN field = 2 bytes
 - Adm field must contain an IP address assigned by IANA
 - AN field is a number assigned by SP
- Examples: 10458:22:10.1.0.0/16 or 1.1.1.1:33:10.1.0.0/16

VPN-IPv4 Address Family

- For the lab exercise, we will use the type-I “administrator field”
 - ROUTER-ID:NUMBER:<ip-prefix>

```
lab@r9> show route table bgp.13

bgp.13vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.8:2:188.188.188.188/32
    *[BGP/170] 00:08:47, localpref 100, from 192.168.0.7
    AS path: I
    > via so-0/0/2.0, label-switched-path R9>R8

lab@r9>
```

VPN-IPv4 Address Family

- Route distinguisher disambiguates IPv4 addresses
 - For example the same router could have 2 VPNs with the same overlapping address space. What sets them apart is the RD since (RD-VPN1 \neq RD-VPN2) on same router.

- We can define RD in two places:
 - Directly within each VRF
 - Or define a global RD id and let the router define the RD for you automatically (less configuration) - preference ?

VPN-IPv4 Address Family

- Define RD within the VRF (manually for each VRF)

```
juniper@lab# show routing-instances
my-vpn {
  instance-type vrf;
  interface ge-0/0/1.110;
  route-distinguisher 10.0.0.15:10;    // RD definition
  vrf-target target:65500:10;
  vrf-table-label;
  routing-options {
    static {
      route 10.10.115.0/24 next-hop 10.10.15.254;
    }
  }
}
```

- Or define the RD-id under routing-options once and let router create RDs for you:

```
juniper@lab# set routing-options route-distinguisher-id ?
Possible completions:
  <route-distinguisher-id> Identifier used in route distinguishers for routing
instances
```

Route Targets

- Route-targets are special extended bgp communities that attach to the vpn-ip prefixes
 - Recap - RDs help MBGP speakers exchange unique routes across the core
 - Target communities are what VRFs use to import and export these routes

- Two ways to define route-targets
 - Define import/export policies per VRF
 - More control - can specify exactly what goes into and out of a VRF
 - Use the JUNOS feature - “vrf-target” in each VRF
 - Auto-exports and imports routes within a VPN - This is what we will use (because it is simple to configure)

Route Targets

- Two ways to define route-targets
 - Use the JUNOS shortcut - “vrf-target” in each VRF

```
juniper@PE0# show routing-instances
my-data-vpn {
  instance-type vrf;
  interface ge-0/0/1.110;
  route-distinguisher 10.0.0.15:10;
  vrf-target target:65500:10;    <----*****
  vrf-table-label;
  routing-options {
    static {
      route 10.10.115.0/24 next-hop 10.10.15.254;
    }
  }
}
```

- This seems a lot simpler to configure
- We can always define more specific policies to override this behavior

Route Targets

- Two ways to define route-targets
 - Define import/export policies per VRF
 - > show policy-options

```
policy-statement VOICE-VPN-IMPORT {  
  term 10 {  
    from {  
      protocol bgp;  
      community voice-vpn;  
    }  
    then accept;  
  }  
  term 20 {  
    then reject;  
  }  
}
```

```
policy-statement VOICE-VPN-EXPORT  
  term 10 {  
    from protocol direct;  
    then {  
      community add voice-vpn;  
      accept;  
    }  
  }  
  term 20 {  
    then reject;  
  }  
}
```

- These policies would then be referenced by the vrf-import, vrf-export within each VRF

Summary - VRF Definition

- We have now defined our VRF with the key elements
- We now need to add some routing protocols to each VRF

```
juniper@PE0# show routing-instances
my-data-vpn {
    instance-type vrf;
    interface ge-0/0/1.110;
    vrf-target target:65500:10;
    vrf-table-label;
    routing-options {
        static {
            route 10.10.115.0/24 next-hop 10.10.15.254;
        }
    }
}
```

MPLS VPNs

■ Re-cap:

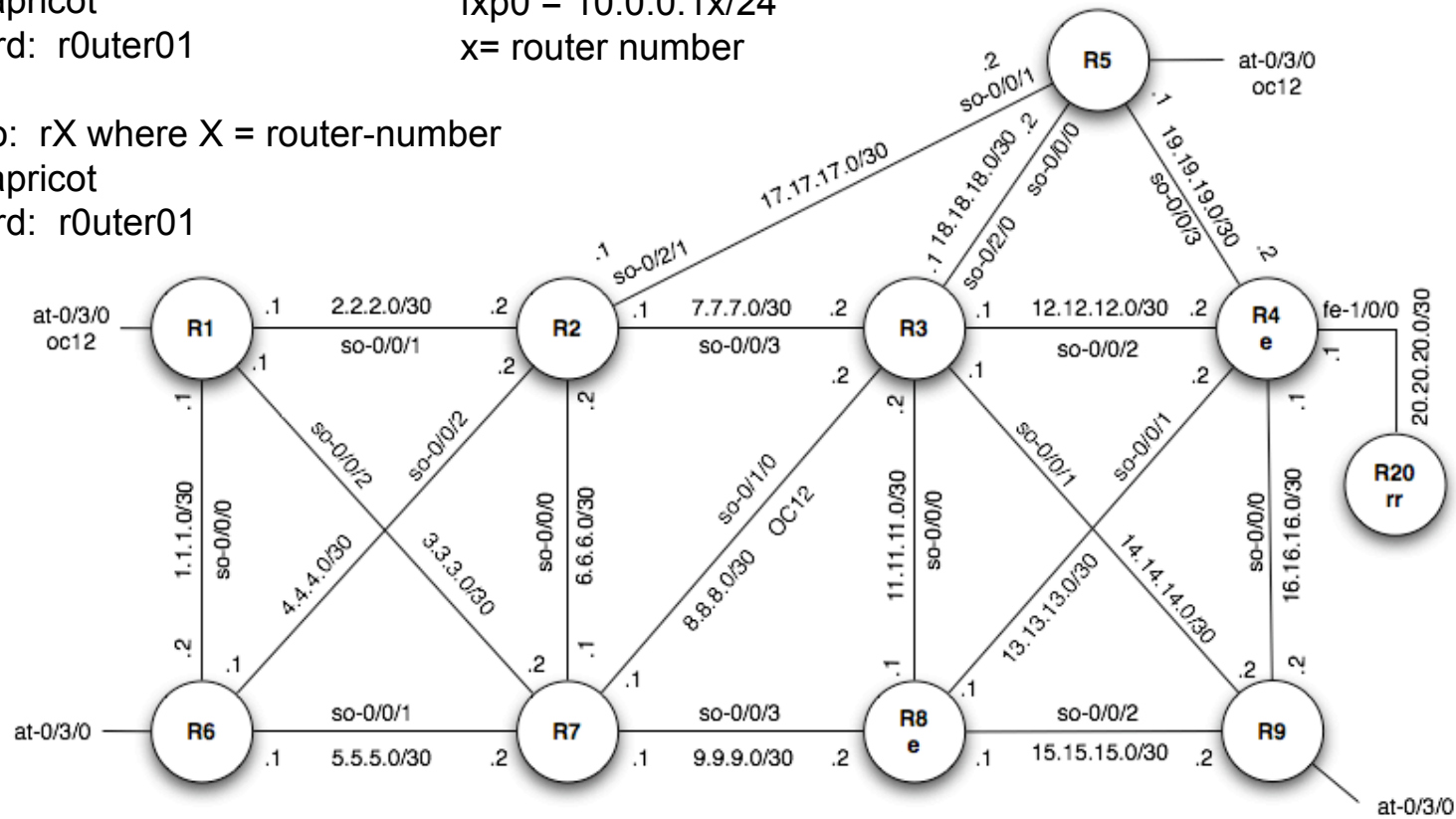
- PEs exchange vpn-ipv4 routes with each other
- PEs also exchange the “inner label” values using MBGP on a per VPN basis.
- Each vpn-ipv4 route has a next-hop associated with it
- PE's use inet.3 to resolve the “next-hop”
- The resolution of the bgp next-hop results in what becomes the outer label
- When one defines a LSP, the LSP route appears in inet.3
- Summary -
 - outer label provided by the configured LSP in inet.3
 - inner label provided by mBGP

Lab Diagram

ssh to: 12.32.57.116 port 2022
Login: apricot
Password: r0uter01

Loopback0 = 192.168.0.x/32
fxp0 = 10.0.0.1x/24
x= router number

Telnet to: rX where X = router-number
Login: apricot
Password: r0uter01



Loopback0 network 192.168.0.x
Router access: 10.0.0.1X

Juniper SEA-POC
Rev 2
Jan 2005

Lab 1

- IGP setup

Lab 1: Basic OSPF Configuration

```
user@router# configure
user@router# set protocols ospf area 0 interface all
user@router# set protocols ospf area 0 interface fxp0 disable
user@router# set protocols ospf traffic-engineering
User@router# commit
```

OSPF on specific interfaces

```
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-x/x/x.0;
      interface so-x/x/x.0;
    }
  }
}
```

OSPF on all interfaces

```
protocols {
  ospf {
    area 0.0.0.0 {
      interface all;
      fxp0 disable;
    }
  }
}
```

Lab 1: Basic OSPF Configuration

- `show route terse`
 - Look for ospf routes 192.168.0.x where x = 1-9

- `show ospf neighbor ?`
 - View status of OSPF neighbors
 - `user@host> show ospf neighbor ?`
 - Possible completions:
 - | | |
|--------------------------------|-----------------------|
| • <code><[Enter]></code> | Execute this command |
| • <code>brief</code> | Show brief status |
| • <code>detail</code> | Show detailed status |
| • <code>extensive</code> | Show extensive status |

- `clear ospf neighbor`
 - Clears and restarts an adjacency

 - `user@host> clear ospf neighbor 192.168.254.225`

Lab 2

- RSVP/MPLS

Lab 2: Enable RSVP

- Enable RSVP and MPLS on all interfaces

```
user@router# set protocols rsvp interface all
user@router# set protocols mpls interface all
User@router# commit
```

- If desired, disable RSVP on a particular interface

```
user@router# set protocols rsvp interface so-0/0/0 disable
```

Lab 2: Enable Family MPLS

- Enable 'family mpls' on all interfaces that will process MPLS packets

```
user@router# set interface so-0/0/0 unit 0 family mpls
```

- Can use groups to simplify this requirement:

```
groups {  
  SONET {  
    interfaces {  
      <so-*> {  
        unit 0 {  
          family mpls;  
        }  
      }  
    }  
  }  
}  
apply-groups SONET;
```

Lab 2: Minimum LSP Configuration

- Configure a signaled LSP path from your PE to all other PE routers
 - For example, if you are using R1, type:
 - set protocols mpls label-switched-path R1>R2 to 192.168.0.2
 - [commit]
 - Repeat until you have LSPs to all other Pes
- Configured on ingress router only

```
protocols {  
  mpls {  
    label-switched-path R6>R8 {  
      to 192.168.0.8;  
    }  
  }  
}
```

Quick MPLS Troubleshooting

- show mpls lsp
- show mpls lsp detail
- show mpls lsp extensive
- show mpls lsp name <lsp name> detail
- show mpls lsp name <lsp name> extensive
- ping mpls ?
- show rsvp session [detail]
 - Reveals FRR RROs
- monitor label-switched-path R6>R8

Optional: Enabling auto-bandwidth

```
groups
  LSP-AUTOBW {
    protocols {
      mpls {
        label-switched-path <*> {
          node-link-protection;
          auto-bandwidth {
            adjust-interval 600;
            adjust-threshold 5;
          }
        }
      }
    }
  }
}
set apply-groups LSP-AUTOBW
```



Lab 3

- We now have a baseline network
- Identify services which belong in VPNs
- Create VPN framework
 - We will add interfaces to these VPNs in the next lab

Lab 3: BGP Route-Reflectors [pre-configured]

- RR BGP Configuration
- Activate reflected routes by either defining a static in inet.3

```
bgp {  
  group RR {  
    type internal;  
    local-address 192.168.0.20;  
    family inet-vpn {  
      unicast;  
    }  
    cluster 192.168.0.1;  
    allow 192.168.0.0/24;  
  }  
}
```

```
routing-options {  
  rib inet.3 {  
    static {  
      route 0.0.0.0/0 discard;  
    }  
  }  
  route-distinguisher-id 192.168.0.6;  
  autonomous-system 65000;  
}
```

- Or define LSPs from RR to each PE

Migration Step I

Prepare MPLS VPN Framework - PE

```
protocols {
  bgp {
    group IBGP {
      type internal;
      local-address 192.168.0.x;
      family inet-vpn {
        unicast;
      }
      neighbor 192.168.0.20;
    }
  }
}
routing-options {
  route-distinguisher-id 192.168.0.x;
  autonomous-system 65000;
}
```

- Point PE to route-reflector
- Define PE's AS to be 65000
- Allow router to auto-generate *route-distinguisher-id*

```
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 192.168.0.1
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 192.168.0.20
Set routing-options route-distinguisher-id 192.168.0.x
commit
```


Migration Step I

Prepare MPLS VPN Framework - PE

```
routing-instances {  
    VOICE {  
        instance-type vrf;  
        interface lo0.1;  
        vrf-target target:65000:1;  
    }  
}
```

- Create a VOICE VPN
- Activate interface lo0 unit 1 and place in VPN
- Define target community for this VPN to be *65000:1*

```
set routing-instances VOICE instance-type vrf  
set routing-instances VOICE interface lo0.1  
set routing-instances VOICE vrf-target target:65000:1
```

Migration Step I

Prepare MPLS VPN Framework - PE

```
protocols {
  rsvp {
    interface all;
  }
  mpls {
    label-switched-path R1>R6 {
      to 192.168.0.6;
    }
    interface all;
  }
  bgp {
    group IBGP {
      type internal;
      local-address 192.168.0.x;
      family inet-vpn {
        unicast;
      }
      neighbor 192.168.0.20;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface all;
    }
  }
}
```

```
routing-options {
  route-distinguisher-id 192.168.0.1;
  autonomous-system 65000;
}

routing-instances {
  VOICE {
    instance-type vrf;
    interface lo0.1;
    vrf-target target:65000:1;
  }
}
```

Lab 4

- Refer to Lab handout section 4

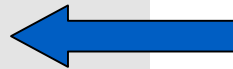
Migration Step II

RIB group configuration

```
routing-options {  
  rib-groups {  
    ACCESS {  
      import-rib inet.0;  
    }  
  }  
  
  route-distinguisher-id 192.168.0.1;  
  autonomous-system 65000;  
}
```

```
routing-instances {  
  VOICE {  
    instance-type vrf;  
    interface ge-7/0/2.0;  
    .  
    routing-options {  
      auto-export {  
        family inet {  
          unicast {  
            rib-group ACCESS;  
          }  
        }  
      }  
    }  
    protocols {  
      .  
    }  
  }  
}
```

- Place CE facing interfaces directly in the VPN
- Share the VPN routes with inet.0 using the “auto-export” knob
- Define a rib-group called ACCESS which imports these routes to inet.0



Migration Step II

RIB group configuration

```
routing-options {
  rib-groups {
    ACCESS {
      import-rib inet.0;
    }
  }

  route-distinguisher-id 192.168.0.1;
  autonomous-system 65000;
}
```

```
routing-instances {
  VOICE {
    instance-type vrf;
    interface ge-7/0/2.0;
    .
    routing-options {
      static {
        route 0.0.0.0/0 next-table inet.0;
      }

      auto-export {
        family inet {
          unicast {
            rib-group ACCESS;
          }
        }
      }
    }
  }
}
```

- At this point, the VPN routes are visible in inet.0
- However, this does not mean that other routers will see these VPN routes (next slide)
- We also need to define a way for packets to exit this VPN if there are no bgp destinations in the VPN
- We use a static default pointing to inet.0

Migration Step II

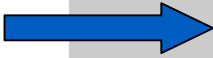
RIB group configuration

- So far
 - Packets can enter a VPN and can ride LSPs if we have LSPs up between PEs i.e. we have inet-vpn bgp routes
 - Packets can hop out of the VPN using a static default and use inet.0 to find their destination if there is no MPLS exit
- Still have one more thing to take care of:
 - How do we advertise VPN routes to non-VPN sites ?
 - We use a policy that calls VPN routes to be advertised in the IGP (inet.0). Now the non-VPN routers can see everything

Migration Step II

RIB group configuration

- We use a policy that exports all routes from VPNs to OSPF speakers



```
lab@r6# show policy-options
policy-statement FROM-VPN {
  term t1 {
    from instance VOICE;
    then accept;
  }
}

[edit]

lab@r6# show protocols
ospf {
  export FROM-VPN;
  area 0.0.0.0 {
    interface so-0/0/1.0;
    interface lo0.0;
  }
}


[edit]
lab@r6#
```

Migration Step III


Remove RIB-groups and static to next-table

- **MPLS used between MPLS VPN enabled PEs !**
- **Non-MPLS routers can still see VPN routes from PEs**
- **Once all sites have been converted to support MPLS VPNs, we remove the following:**

```
routing-options {  
  rib-groups {  
    ACCESS {  
      import-rib inet.0;  
    }  
  }  
}  
  
route-distinguisher-id 192.168.0.1;  
autonomous-system 65000;
```



```
routing-instances {  
  VOICE {  
    instance-type vrf;  
    interface ge-7/0/2.0;  
    .  
    routing-options {  
      static {  
        route 0.0.0.0/0 next-table inet.0;  
      }  
      auto-export {  
        family inet {  
          unicast {  
            rib-group ACCESS;  
          }  
        }  
      }  
    }  
  }  
}
```



Migration Step II

Some Tuning if required

```
routing-options {  
  rib-groups {  
    ACCESS {  
      import-rib inet.0;  
      import-policy BLOCK-DEFAULT-FROM-VPN;  
    }  
  }  
  
  route-distinguisher-id 192.168.0.1;  
  autonomous-system 65000;  
}
```

```
policy-options {  
  policy-statement BLOCK-DEFAULT-FROM-VPN {  
    term t1 {  
      from {  
        route-filter 0.0.0.0/0 exact;  
      }  
      then reject;  
    }  
    term t2 {  
      then accept;  
    }  
  }  
}
```

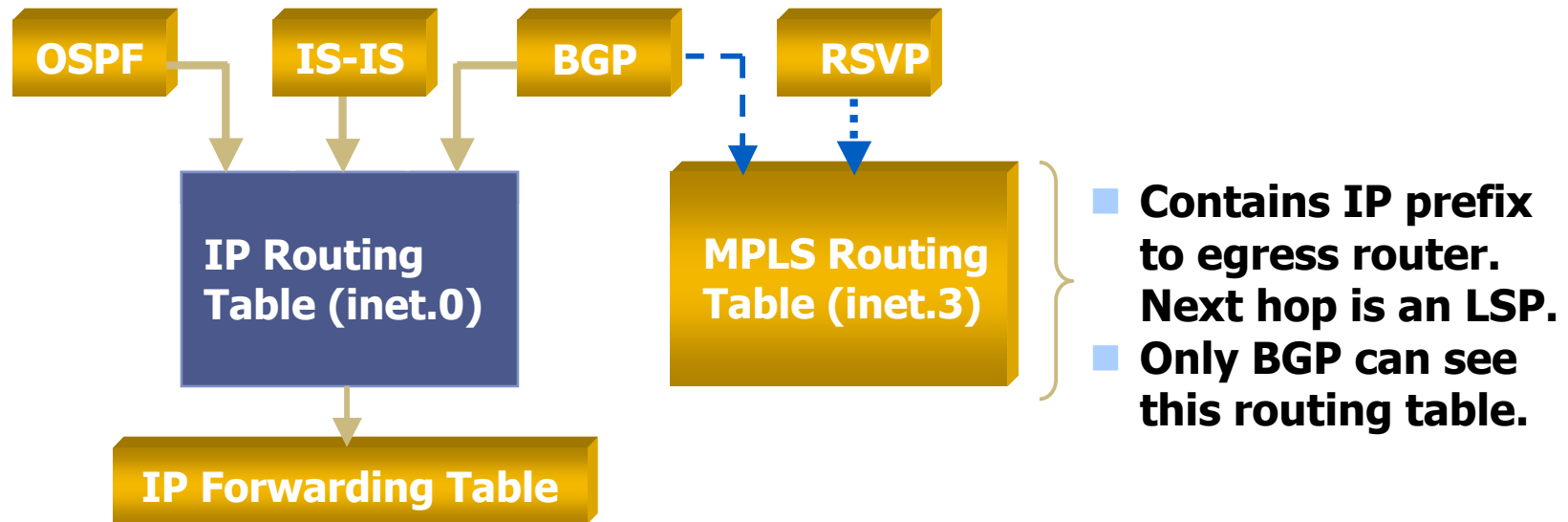


Juniper *your* Net

Useful commands

- `show route [detail|extensive]`
- `show route table inet.0`
- `show route table bgp.l3vpn.0`
- `show route table <vpn-name>`
- `ping routing-instance <name> <host>`
- `traceroute routing-instance <name> <host>`
- `ping mpls ?`
- More...

Route Resolution: Ingress Router Behavior



- RSVP
 - Signals LSP to egress router
 - Installs IP prefix to egress router into MPLS routing table
 - Next hop is the outbound interface and label of the first hop in the LSP

Further Reading

- New Book
 - MPLS-Enabled Applications
- Whitepapers/Application Notes
 - BGP Route Reflection in L3 VPN Networks
https://www.juniper.net/solutions/literature/white_papers/200160.pdf
 - Introducing MPLS L3 VPNs in Mobile Operator Networks
https://www.juniper.net/solutions/literature/app_note/350058.pdf

