Avoiding Unnecessary Upstream Traffic in Bidir-PIM

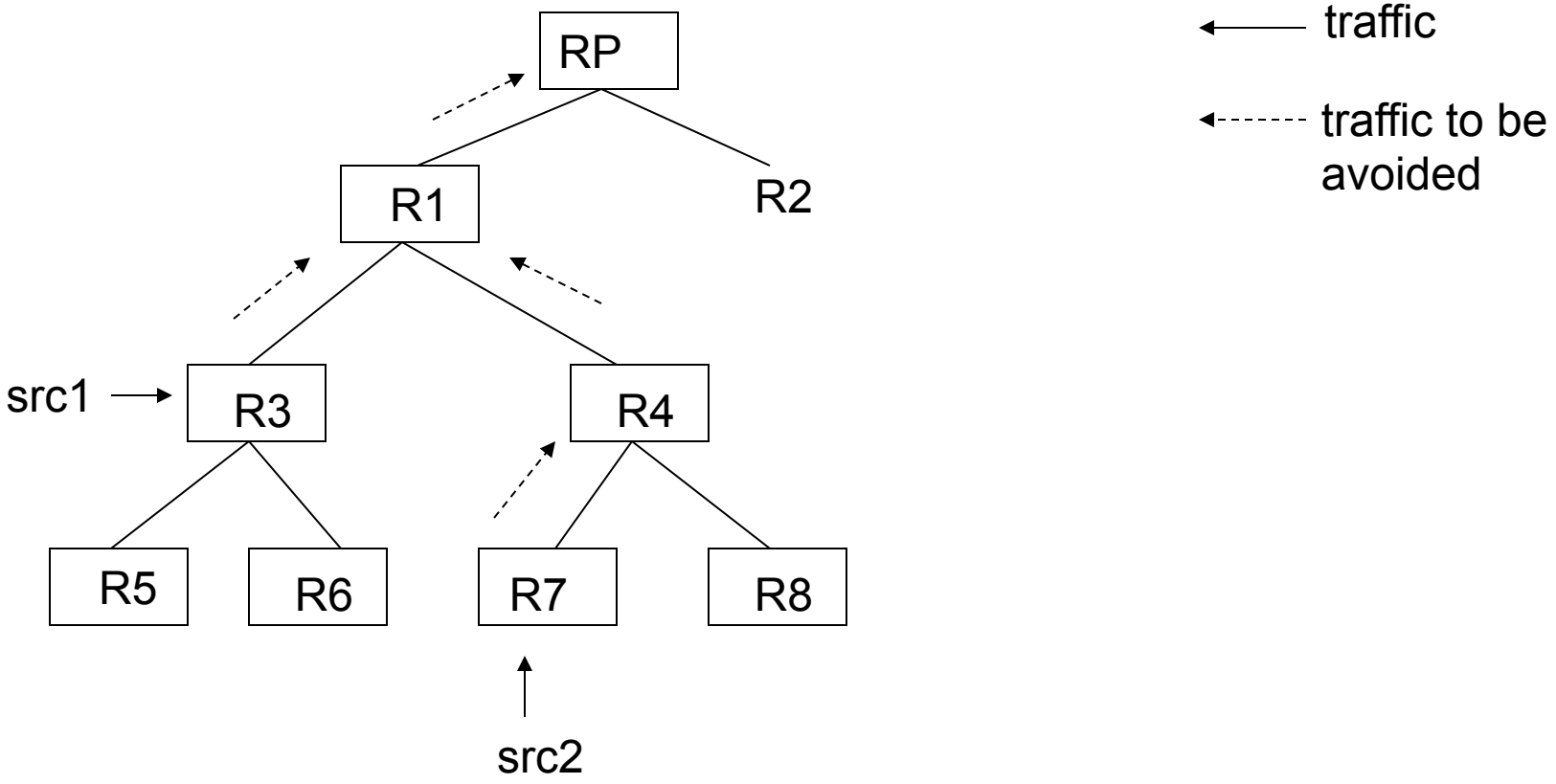
Jeffrey Zhang
Weesan Lee

Juniper Networks

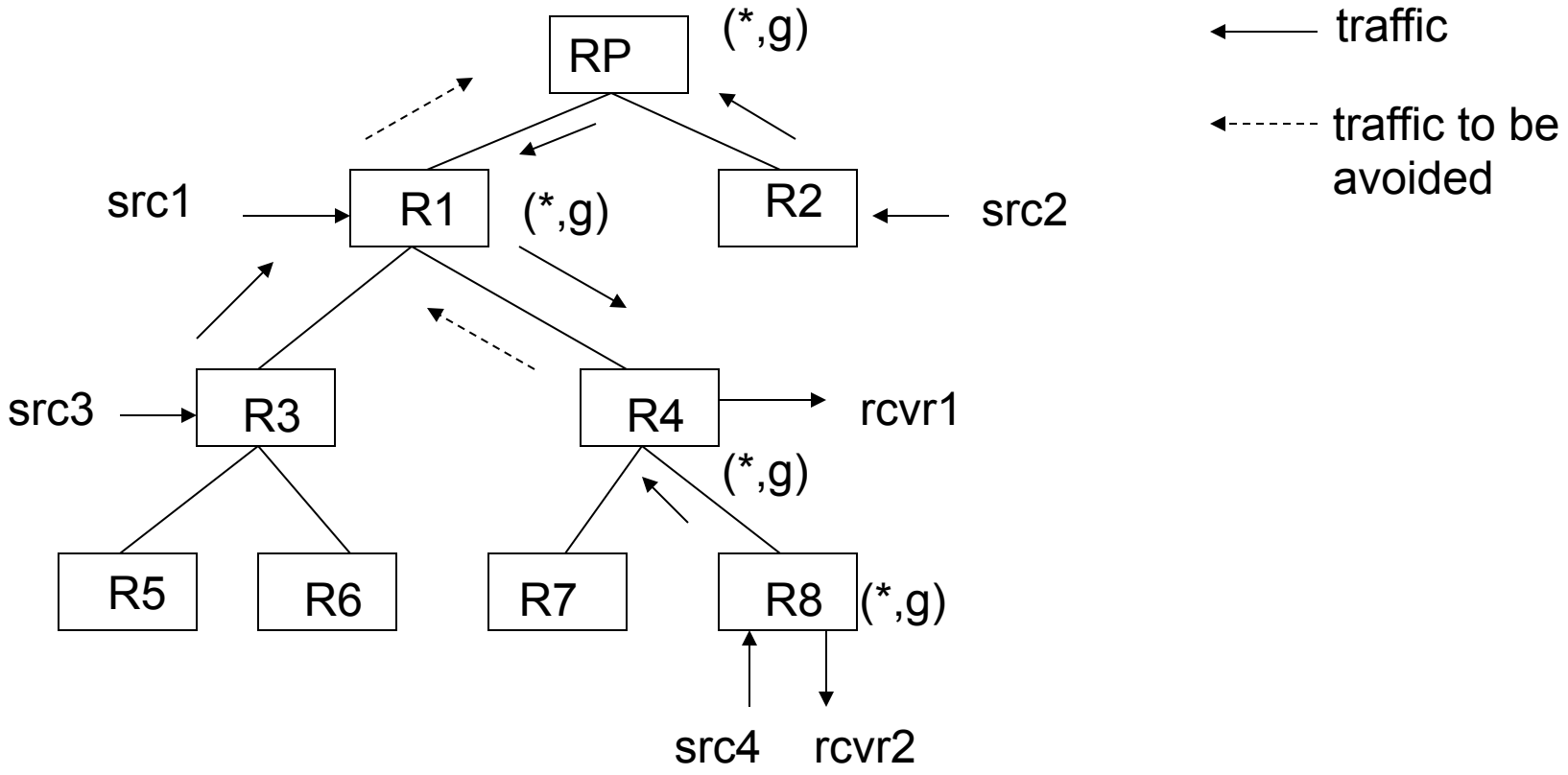
82th IETF, Taipei

Scenario #1 of unnecessary upstream traffic

- The tree depicts all routers and their RPF interfaces (towards RP)
  - Assuming RPA is on a loopack interface
    - More general situation described later
- No receiver anywhere – traffic should be stopped at the FHR R3/R7



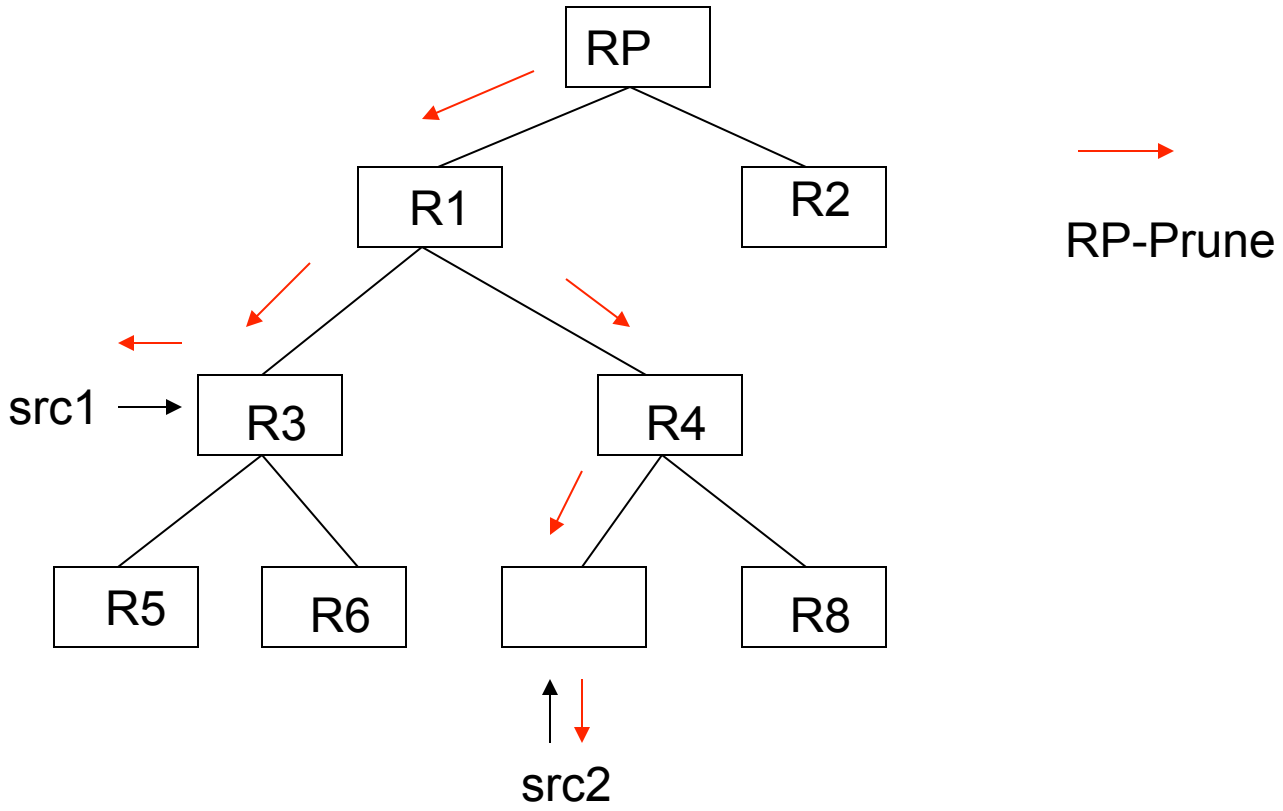
Scenario #2 of unnecessary upstream traffic



RP-Prune procedures for scenario #1

- RP has (*, G-prefix) notification route
  - vs. the normal “forwarding to RPL” route
  - Traffic hitting the route triggers throttled control plane notification
    - Indicating traffic is being received unnecessarily
      - Otherwise it would have hit specific (*,G) forwarding route
  - The notification triggers RP-Prune out of the incoming IF
    - Multicast to ALL-PIM-ROUTERS
      - Interface-wide RP-Prune
- Upon receiving the RP-Prune, downstream router installs (*, G) notification route
  - Subsequent traffic hits the (*, G) notification route, triggering RP-Prune further downstream, instead of being forwarded upstream
- RP-Prune is data-triggered hop-by-hop:
  - by the pre-installed (*,G-prefix) notification route on RP
  - by the triggered (*,G) notification route on downstream routers

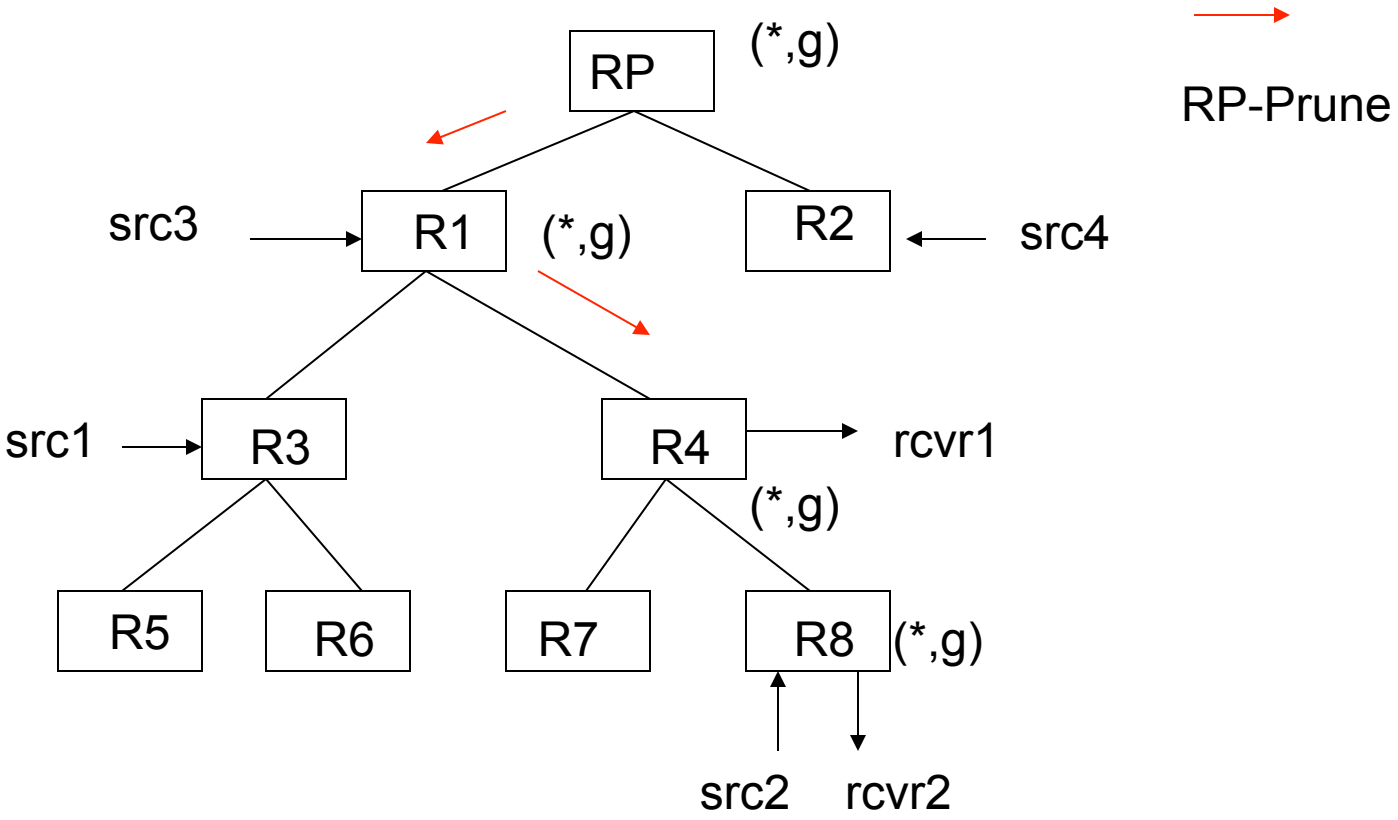
RP-Prune scenario #1 illustration



RP-Prune procedures for scenario #2

- RP sends periodic neighbor-specific RP-Prune to the only downstream neighbor in (*,G) join state
  - Assumes Explicit Tracking
    - If ET is not used, another downstream with the (*,G) state needs to trigger overriding (*,G) join upon receiving the RP-Prune, and the target of RP-Prune needs to ignore the RP-Prune when receiving that overriding (*,G) join
- Downstream router prunes the RPF IF from (*,G) forwarding route's OIF list
  - Stops traffic from going upstream
- Downstream router further propagates RP-Prune to its only downstream neighbor
  - Terminates when there are more than one downstream IF/ neighbor – R4 in the example

RP-Prune scenario #2 illustration



RP-Prune states

- Upstream IF & Neighbor
- List of downstream IF where RP-Prune was sent
  - Flag for interface-wide or nbr-specific RP-Prune
  - List of downstream neighbor on the IF
    - A nbr-specific prune was sent to it, or
    - A RP-Prune-Keep was received from it
      - A timer to time out if downstream stops refreshing RP-Prune-Keep
      - Interface-wide RP-Prune case (scenario #1)

RP-Prune state maintenane: Interface-wide RP-Prunes (Scenario #1)

- Maintained from downstream upwards
  - FHR keeps receiving data traffic, keeps the RP-Prune state alive for the incoming interface, and refreshes RP-Prune-Keep towards its upstream
    - Each hop will refresh its upstream by RP-Prune-Keep
  - After source stops sending, FHR times out its RP-Prune state, and sends RP-Prune-Cancel upstream
    - Each hop propagates the RP-Prune-Cancel if it no longer has other downstream in the RP-Prune state (i.e., no other sending branches)

RP-Prune state maintenance: Neighbor-specific RP-Prunes (Scenario #2)

- Maintained from RP downwards
  - RP refreshes RP-Prune as long as it has only one downstream neighbor
  - Each hop propagates the RP-Prune unless it has more than one downstream IF/Nbr

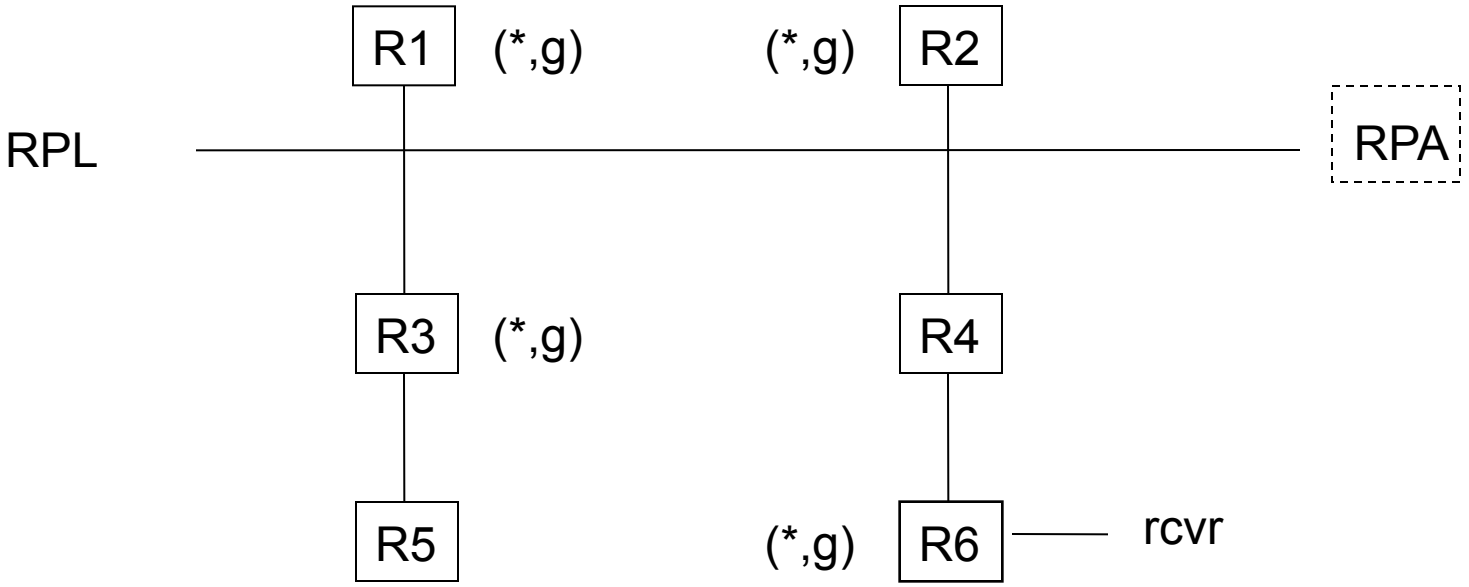
RP-Graft Procedures

- RP sends interface-wide RP-Graft to wherever it sent interface-wide RP-Prune before, when it gets initial corresponding (*,G) join state
  - also sends neighbor-specific RP-Prune to the new/only downstream neighbor (scenario #1 becomes #2)
- Any router (RP or not) who sent Neighbor-specific RP-Prune before triggers RP-Graft when it first gets more than one downstream IF/nbr for a (*, G) join state
  - Neighbor-specific RP-Graft sent to the downstream neighbor recorded in the RP-Prune state
- Upon receiving RP-Graft, downstream routers:
  - Removes (*,G) notification route (scenario #1), or
  - Adds RPF IF to (*,G) forwarding route (scenario #2)
  - Reply with RP-Prune-Cancel as acks
    - Upstream retransmits RP-Graft until all applicable downstream neighbors in the RP-Prune state have ack’ ed

What if RPA is not on a loopback IF?

  - Traffic always forwarded to the RPL
  - Join/Prunes not sent to the RPL
- Modified behavior:
  - Join/Prunes sent to ALL-PIM-ROUTERs on RPL
    - But not further downstream
  - Traffic forwarded to the RPL only if a corresponding join has been received on the RPL
  - Each router on the RPL is called a virtual RP and follows the RP-Prune/Graft procedures defined earlier
    - Another virtual RP considered as a downstream for a particular group if a join has been received from it
    - RP-Graft/Prune/Prune-Keep/Prune-Cancel messages are not sent to the RPL though

Virtual RP illustration



- Virtual RP R1~R3 all have only one downstream (R4 on the RPL), but do not send RP-Prune (to R4 on the RPL)
- Virtual RP R4 has one downstream R6 and sends RP-Prune to it
  - If R5 gets a receiver, R4 will send RP-Graft to R6 because it now

Handling of topology changes

- Sends RP-Graft downstream

  - Remove (*,G) notification routes, or
  - Add back, or add new RPF IF to (*,G) forwarding route

Scaling properties

- Data-driven (*,G) RP-Prune states on relevant sending branch only
  - For traffic that nobody wants

    traffic with receivers does go through w/o data-driven events
  - Time out when source stops sending
- Scenario #2
  - Control-driven (*,G) RP-Prune states on relevant

Minimum states for stopping unnecessary upstream traffic

Next steps

- Seek comments and WG adoption