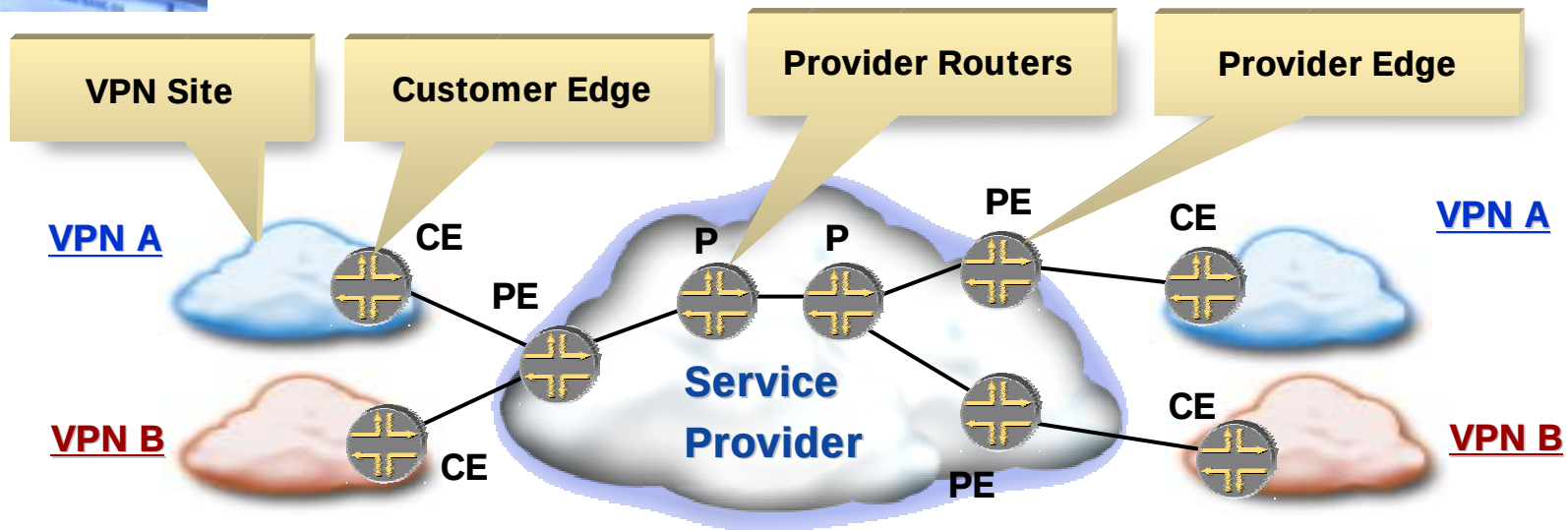
Improving Service Availability for Provider-Provisioned VPNs

Yakov Rekhter

(yakov@juniper.net)

Agenda



Service provider offers 2547 VPNs, VPLS, Layer 2 VPNs, and Internet (potentially on the same PE)

- **Protection against misbehaved customers (CEs)**
- **Handling control plane faults within the Service Provider network (P and PE routers)**
- **Handling data plane faults within the Service Provider network**
- **Summary**

Protection against misbehaved CEs

- **Problem: How to make sure that one (misbehaved) VPN customer would not disrupt VPN services for other customers**
  - **misbehavior at the control plane:**
    - **CE sends too many routes to PE, or**
    - **CE sends too many (too frequently) routing updates to PE, or**
    - **CE sends incorrect (malformed) routing updates to PE**
- **Solution: By implementing resource isolation within the control plane of the Service Provider network**
  - **including fault isolation**
- **Resource isolation per customer / per service granularity is desirable**

Control Plane Resource Isolation

- **With Layer 2/VPLS/2547 VPN services resource isolation is relevant to PE routers only**
  - **Requires the ability to contain faults and control the amount of resources (CPU, memory) at the granularity of an** individual **customer/service**
    - **Per service granularity isn't enough**
- **With Internet service resource isolation applies to PE routers only**

Control Plane Resource Isolation (cont.)

- **PE Memory** - protecting against CE sending too many routes to PE:
  - 2547 VPN: Max route limit per VRF for L3 VPN services
  - Internet: Max route limit per EBGP neighbor
  - VPLS: Max MAC addresses per VPLS Forwarding Table
- **PE CPU** - protecting against CE sending too many (too frequently) routing updates to PE:
  - 2547 VPN, Internet: on PE rate limit control traffic (e.g., BGP, RIP, OSPF updates) received from CE
    - On a per (logical) customer interface basis
    - Rate limiting control traffic implemented in the PE data plane with no negative impact on performance

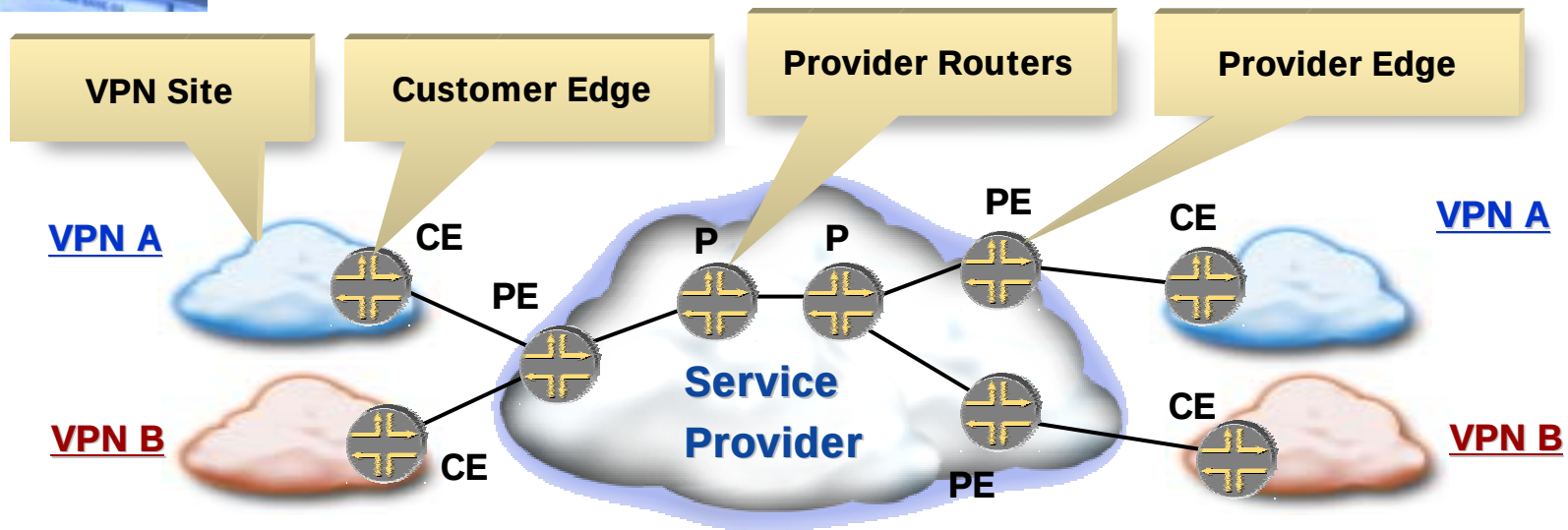
Control Plane Fault Isolation (cont.)

- Scenario 1: (control) traffic of one VPN customer may cause PE control plane crash
  - "Perfect solution" - per customer/per service instance of control plane on the PE
    - Including memory protection on a per instance (per customer/per service) basis
      - has negative impact on the overall scalability – may not be practical
  - In the absence of per customer/service control plane instance could be addressed by graceful restart on PE
    - more on this later…
- Scenario 2: (control) traffic of one VPN customer can not cause PE control plane crash
  - Single instance of control plane on the PE for all customers/services is adequate

Protection against misbehaved customers (CEs): Summary

- Limit CPU usage on PEs by rate limiting CE to PE control traffic (routing updates)
  - On a per (logical) customer interface basis
- Limit memory usage on PEs by enforcing upper bound on the size of individual VRFs/VPLS Forwarding Table
- Focus: improving service availability in the presence of misbehaved CEs
  - Does not deal with service disruption due to other faults (e.g., PE control plane faults, etc…)
- Not a perfect solution
  - Folks who are interested in a perfect solution should look elsewhere
- But a useful tool to deal with a specific set of problems

Agenda



Service provider offers 2547 VPNs, VPLS, Layer 2 VPNs, and Internet

- Protection against misbehaved customers (CEs)
- Handling control plane faults within the Service Provider network (P and PE routers)
- Handling data plane faults within the Service Provider network
- Summary

Why does control plane restart ?

- Due to control plane software upgrades
  - aka “planned restart”
- Due to control plane software bugs
  - aka “unplanned restart”

Both exist in the real life -> need a solution that handles both !!!

Impact of control plane restart - current situation

- Disrupt services by disrupting data path used by the services on the restarting node
- Control plane restart on PE is especially disruptive
  - As all the VPN sites that have connectivity just to that PE loose connectivity to other VPN sites
  - Disruption lasts as long as it takes for the PE to restart and to reacquire all the routing information
    - Both from other routers within the service provider network (both PEs and Ps) as well as from the directly connected CEs

Impact of control plane restart - current situation (cont.)

- Also disrupt services due to transient forwarding loops that could happen during routing convergence in response to control plane restart
  - Disruption happens twice: once when the control plane goes down, and once when the control plane comes back
  - Disruption involves multiple nodes, not just the node whose control plane restarts
    - The scope of the affected nodes is hard to predict (other than providing the worst case scenario)
  - Disruption lasts for as long as it takes routing to converge
    - The time it takes routing to converge is hard to predict
      - Because the scope of the nodes that have to converge is hard to predict (other than providing the worst case scenario)
  - ⇨ The scope and the duration of the disruption is hard to predict (other than providing the worst case scenario)

Impact of control plane restart - current situation (cont.)

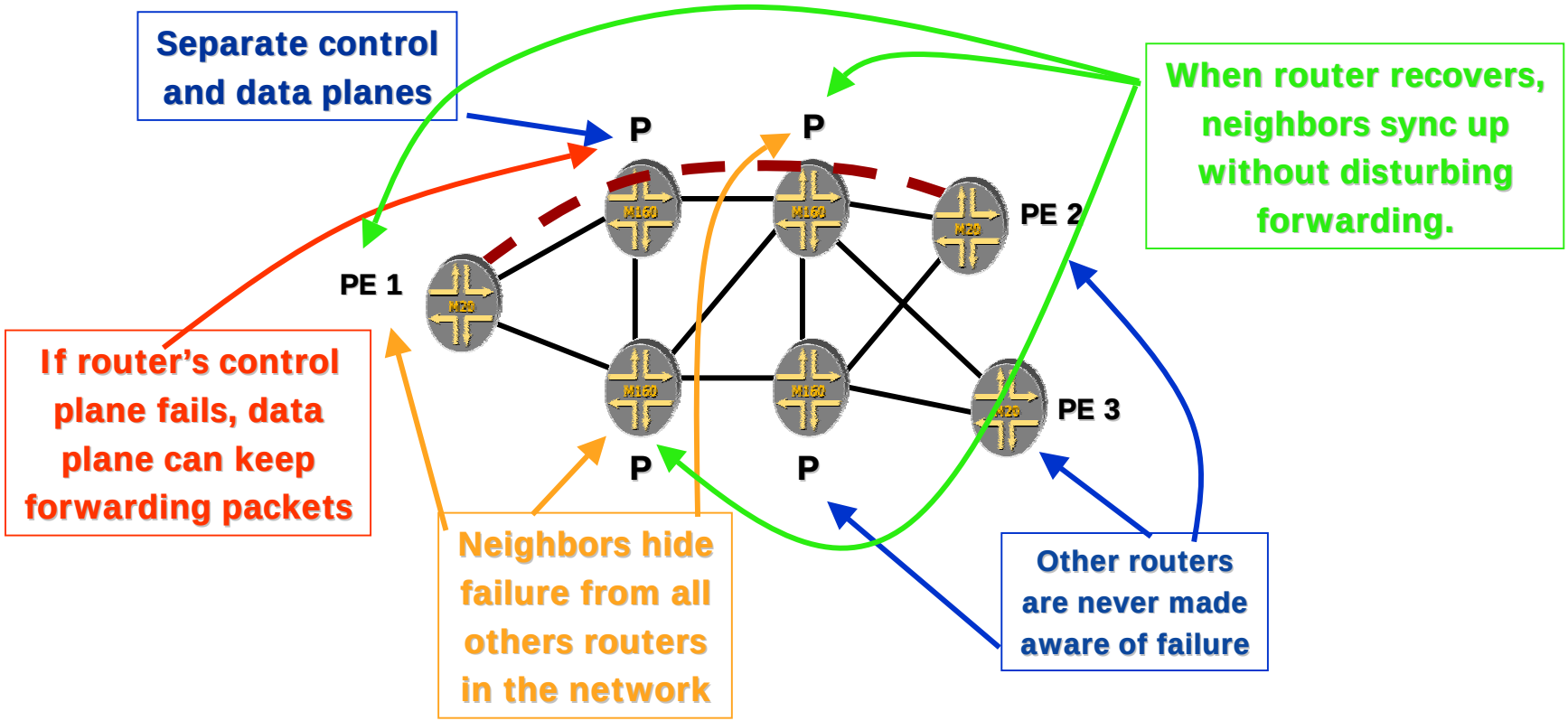
- Increases the load on the control plane
  - Involves multiple nodes, not just the node whose control plane restarts
    - The scope of the affected nodes is hard to predict (other than providing the worst case scenario)
  - Adversely impacts the scalability of the control plane
  - Adversely impacts the adaptability and convergence of the control plane

Bottom line: control plane restart adversely impacts service availability !!!

Graceful Restart - objectives:

- Improve service availability by minimize disruption of services (e.g., 2547 VPNs, L2 VPNs, VPLS, Internet, etc...) due to the control plane restarts
- Restart could happen anywhere in the network that delivers the service
  - Either at the edge (PE), or in the middle (P)
- Handle either planned (e.g., control plane software upgrade), or unplanned (control plane crash) restart

Graceful Restart - How ?



Graceful Restart - How ? (cont.)

- On the restarting node separate control component from forwarding component:
  - e.g., RE – control component (control plane)
  - e.g., PFE – forwarding component (forwarding state)
- On the restarting node preserve the forwarding state (forwarding component) across the restart of the control plane (control component)
- Localize the knowledge that the node's control plane restarts to only the routing peers of the restarting node
- On the routing peers of the restarting node preserve routing information associated/received from the restarting node across the restart of the control plane of the restarting node
- Restarting node (re)learns its routing information from its routing peers

Graceful Restart - How ? (cont.)

- Graceful restart mechanisms are protocol specific:
  - BGP – see draft-ietf-idr-restart-05.txt
  - ISIS – see draft-ietf-isis-restart-01.txt
  - OSPF – see draft-ietf-ospf-hitless-restart-02.txt
  - LDP – see draft-ietf-mpls-ldp-restart-06.txt
  - BGP/MPLS – see draft-ietf-mpls-bgp-mpls-restart-02.txt
  - RSVP - draft-ietf-mpls-generalized-rsvp-te-09.txt
  - RIP – already build in !!!
- No preservation of any of the protocol-related state across the restart on the restarting node
  - For all of the above protocols !!!
- Graceful restart of control plane is more than graceful restart of individual components/protocols !!!
  - need to take into account interaction/dependencies among the protocols at the time of restart

Graceful Restart – results:

- **No disruption in the data path on the restarting node**
  - **Due to restarting node preserving its forwarding component**
- **No disruption in the data path on the routing peers of the restarting node**
  - **Due to the routing peers of the restarting node preserving routing information associated/received from the restarting node across the restart of the control plane of the restarting node**
    - **Implies that the routing peers don't modify their forwarding state in response to the restart of the control plane of the restarting node**
- **No disruption of the data path elsewhere**
  - **Due to nodes other than the routing peers of the restarting node being unaware of the restart of the control plane of the restarting node**

Graceful restart – results (cont.):

- **No change in the traffic pattern**
  - **preserves steady state traffic pattern**
  - **no impact on jitter, latency, packet ordering, route optimality**
- **Improved control plane scalability**
  - **By limiting the scope of the nodes that are aware of the restart to only the routing peers of the restarting node**
- **Improved control plane convergence/adaptability**
  - **By limiting the scope of the nodes that are aware of the restart to only the routing peers of the restarting node**

Graceful Restart and control plane memory corruption

- Software bugs in OSPF, ISIS, BGP, LDP, RSVP are causes of control plane memory corruption
- Control plane memory corruption (due to software bugs) is one of the reasons for control plane restart
- Graceful restart does NOT preserve (corrupted) control plane memory
  - Graceful restart preserves only the forwarding state
  - Graceful restart does NOT preserve protocol-related (e.g., OSPF, ISIS, BGP, LDP, RSVP) state on the restarting node
- ⇨ Control plane memory corruption is cleared after graceful restart

Why not replicate protocol-related state (e.g., BGP, OSPF, ISIS, etc…)

- Because replicating protocol-related state adds non-negligible amount of complexity to the control plane
  - significantly more than what is required to support graceful restart
  - more code means more probability of software bugs -> replicating protocol-related state is more likely to be the cause of control plane failure than graceful restart
- Because one of the reasons for control plane restart is the corruption of the protocol-related state (due to software bugs)
- The replicated state is not “immune” to memory corruption due to software bugs
  - The same software bugs could corrupt both the “primary”, as well as the replicated state
- In contrast, by re-creating protocol related state from scratch graceful restart clears corrupted state – eliminates the cause for the restart

Why not replicate protocol-related state (cont.)

- Replicating protocol-related state may be appropriate when handling planned restart (e.g., software upgrades)
- Replicating protocol-related state is NOT appropriate for handling restart due to software bugs (unplanned restart)
- In contrast, graceful restart is appropriate for both planned restart, as well as unplanned restart (restart due to software bugs)

What about routers that can't preserve forwarding state ?

- **Implementing a subset of the graceful restart is useful, even if a router can't preserve its forwarding state across the restart of its control plane**
  - **Enables such routers to take advantage of the neighbors that can preserve forwarding state across the restart**
    - **CE routers need not be able to preserve forwarding state to take advantage of the graceful restart capabilities available on the service provider routers (PE and P)**

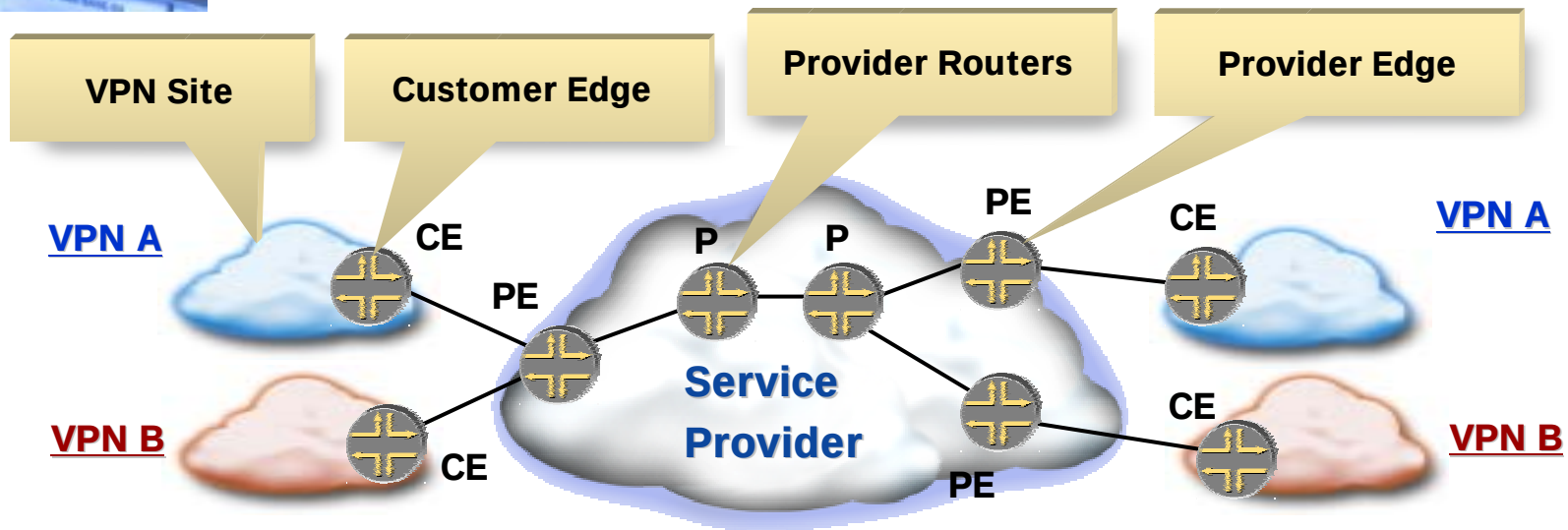
Potential Drawbacks

- If routing changes during the time it takes to restart the control plane, then there is a potential for transient forwarding loops
  - The loops will last as long as it would take the (restarting) control plane to get up to date routing information
  - Having secondary (standby) RE helps
    - Reduces the time to restart the control plane
  - Only a subset of the services whose data path traverses through the restarting node may be impacted
    - Impact only the services affected by the routing changes AND whose data path traverses through the restarting node
- An alternative (recalculating routes as the control plane goes down and then comes back) may result in transient forwarding loops as well
  - Impacts all the services whose data path traverses through the restarting node
    - May impact other services as well (due to shifting traffic)

Handling Control Plane Faults: Summary

- Minimalistic approach
  - With respect to the amount of state that has to be preserved (as requires to preserve only the forwarding state)
  - With respect to the additional complexity introduced by the graceful restart
    - No need to preserve any of the protocol-related state across the restart
- Focus: improving service availability in the presence of control plane restart
  - Does not deal with service disruption due to other faults (e.g., link failures, misbehaved CEs, etc…)
- Not a perfect solution
  - Folks who are interested in a perfect solution should look elsewhere
- But a useful tool to deal with a specific set of problems

Agenda



Service provider offers 2547 VPNs, VPLS, Layer 2 VPNs, and Internet

- Protection against misbehaved customers (CEs)
- Handling control plane faults within the Service Provider network (P and PE routers)
- Handling data plane faults within the Service Provider network
- Summary

Types of Data Plane faults

- Link failure
- Node forwarding plane failure
  - E.g., line card failure, switching fabric failure
  - Partial failures possible (e.g., a particular line card failed while the switching fabric and the rest of the line cards on a given node are ok)

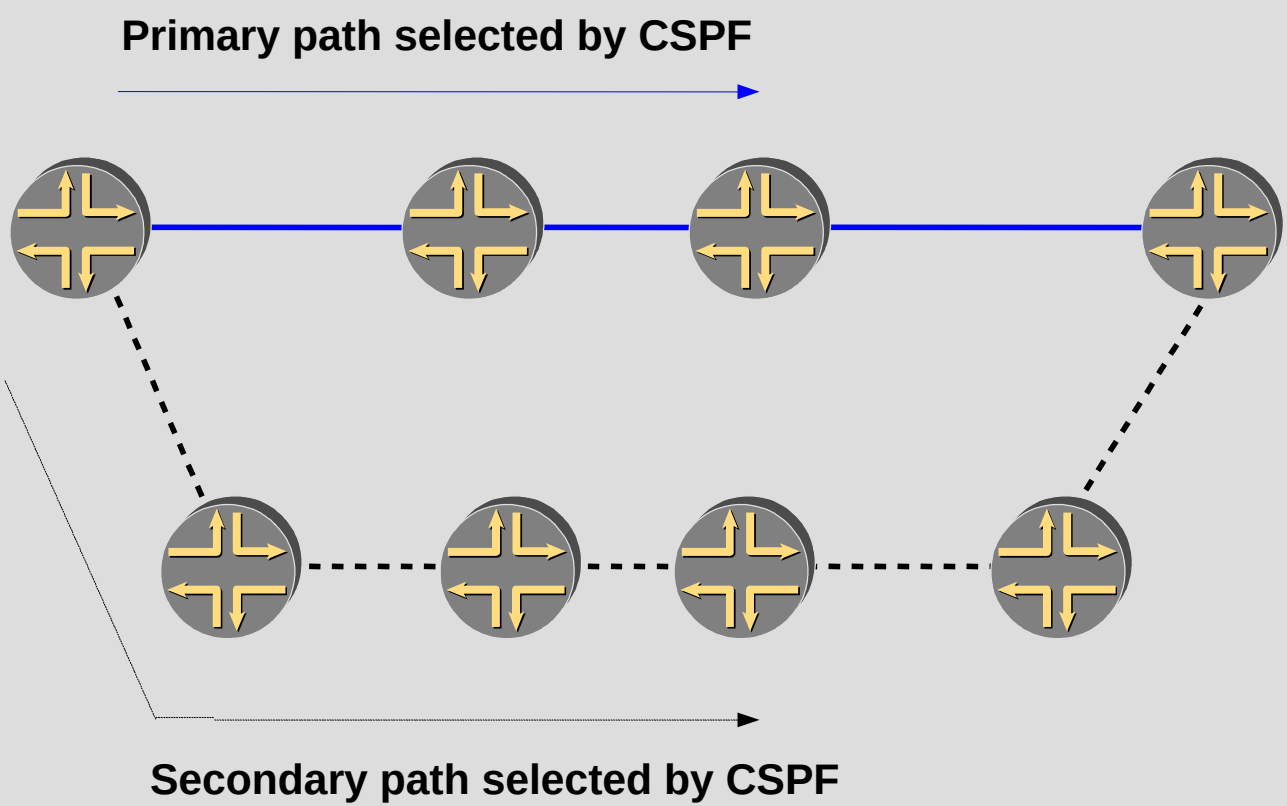
MPLS-based mechanisms

- Path protection (aka Secondary LSP)
- Local 1:1 protection (aka LSP Protection Fast Reroute)
- Local 1:N protection (aka Facility-based Fast Reroute)

Path Protection

- An LSP may have multiple paths
- Primary path is the preferred path to set up and use
- Secondary paths are alternatives, to be used when the primary fails
  - Usually node/link disjoined from primary
    - The level of overlap between the primary and the secondary could be controlled
- Two types of secondary paths:
  - Standby secondary paths are pre-computed and pre-signaled
  - Non-standby secondary paths are pre-computed, but not pre-signaled
- Standby secondary may result in wasting resources
  - Resources reserved for standby secondary are reserved all the time, yet used only when the primary fails
  - Resource waste could be reduced (ala GMPLS shared mesh restoration)
    - Work in progress within GMPLS

Example: Path Protection



Recovery Speed with Path Protection

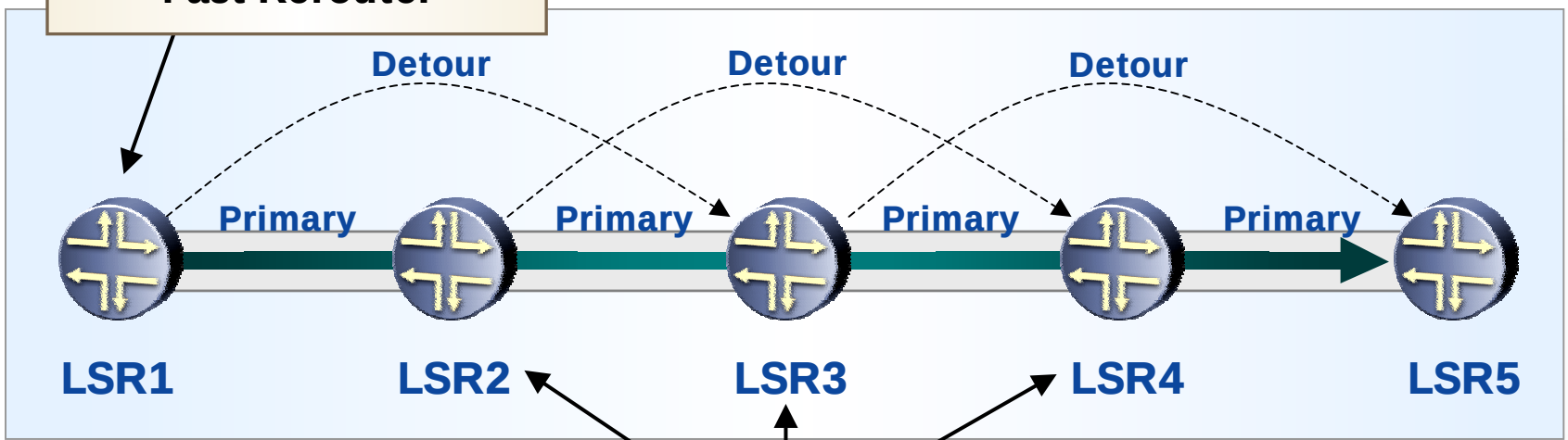
- Switching to secondary (either standby or non-standby) requires propagating error to the head-end of the LSP
  - RSVP Path Error or Resv Tear message
- In addition, switching to (non-standby) secondary requires signaling secondary after the failure of the primary
- Secondary path may be pre-computed
  - Eliminates the need for Constrained SPF (CSPF) at the time of failure
- Handles both link failure and node forwarding plane failure
  - Except for the failure of the forwarding plane at the head-end or tail-end nodes
- Packet loss occurs until LSP is redirected by head-end LSR

MPLS-based mechanisms

- Path protection (aka Secondary LSP)
- Local 1:1 protection (aka LSP Protection Fast Reroute)
  - Protects against both link failure and node forwarding plane failures
- Local 1:N protection (aka Facility-based Fast Reroute)

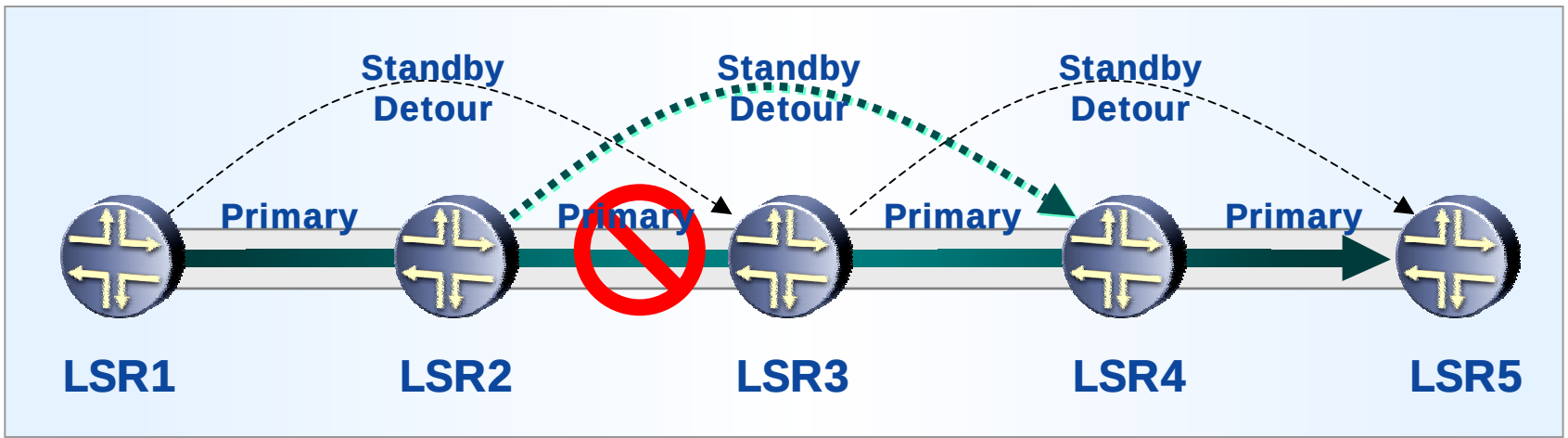
Local 1:1 Protection Operation

Single user command at head end to enable Fast Reroute.



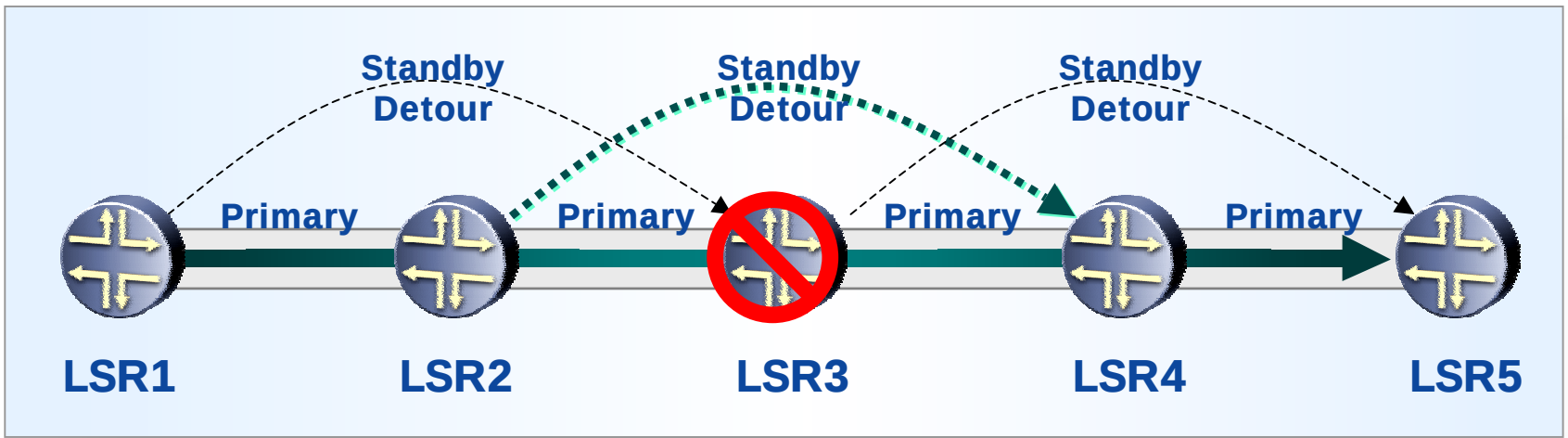
- **Fast reroute is signaled to each LSR in the path**
- **Each LSR computes and sets up a detour path that avoids the next link and next LSR**
- **Each LSR along the path uses the same route constraints used by head-end LSR to ensure constraint-based routing requirements are maintained**

Local 1:1 Protection Operation: Link Failure



- LSR2 detects that an interface in an LSP has gone down and reroutes via standby detour
  - Recovery time is limited by the time to detect the failure
    - Comparable to SONET APS
- Packet loss is minimized to the unlucky few that were transiting at the time of failure

Local 1:1 Protection Operation: Node Failure

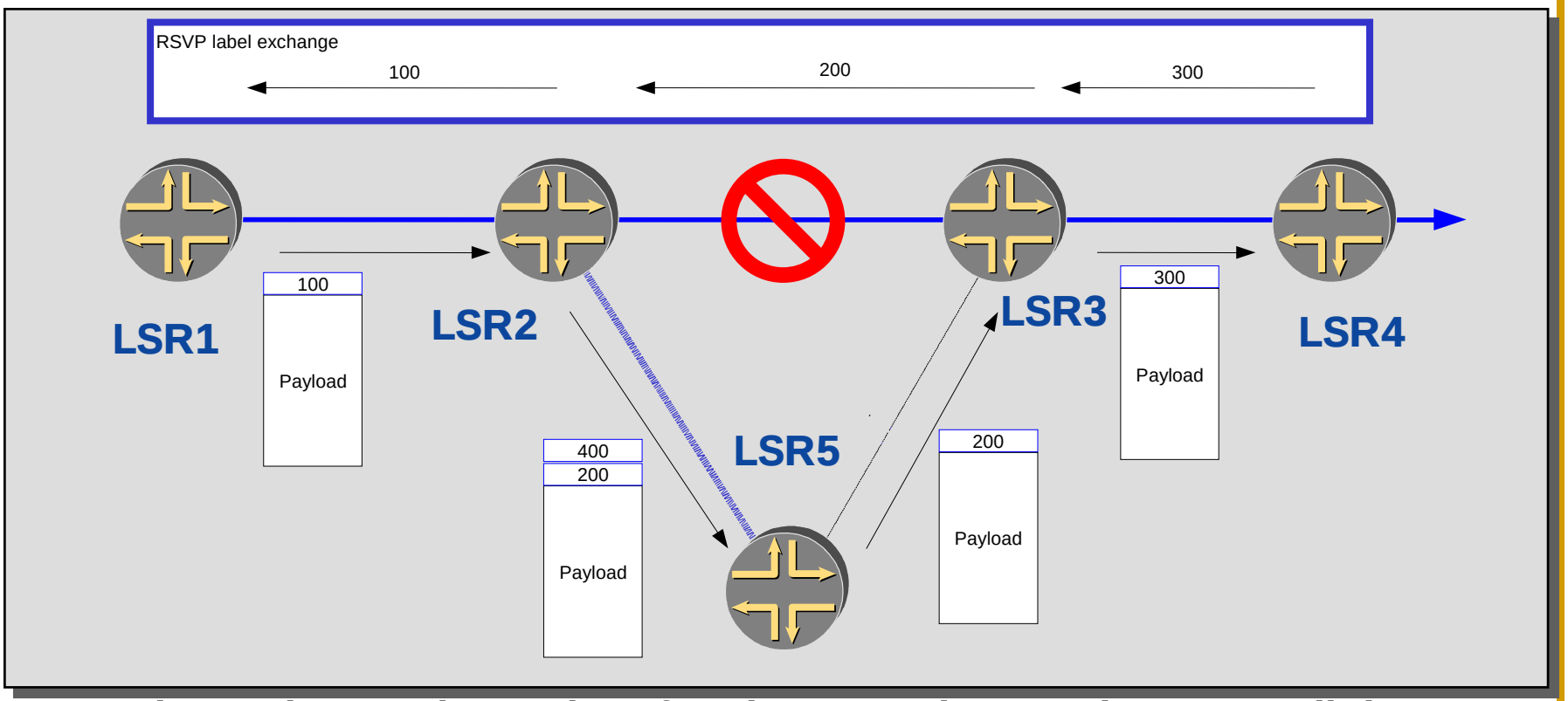


- LSR2 detects that neighbor's (LSR3) forwarding plane has gone down and reroutes via standby detour
  - Recovery time is limited by the time to detect the failure
- Packet loss is minimized to the unlucky few that were transiting at the time of failure

MPLS-based mechanisms

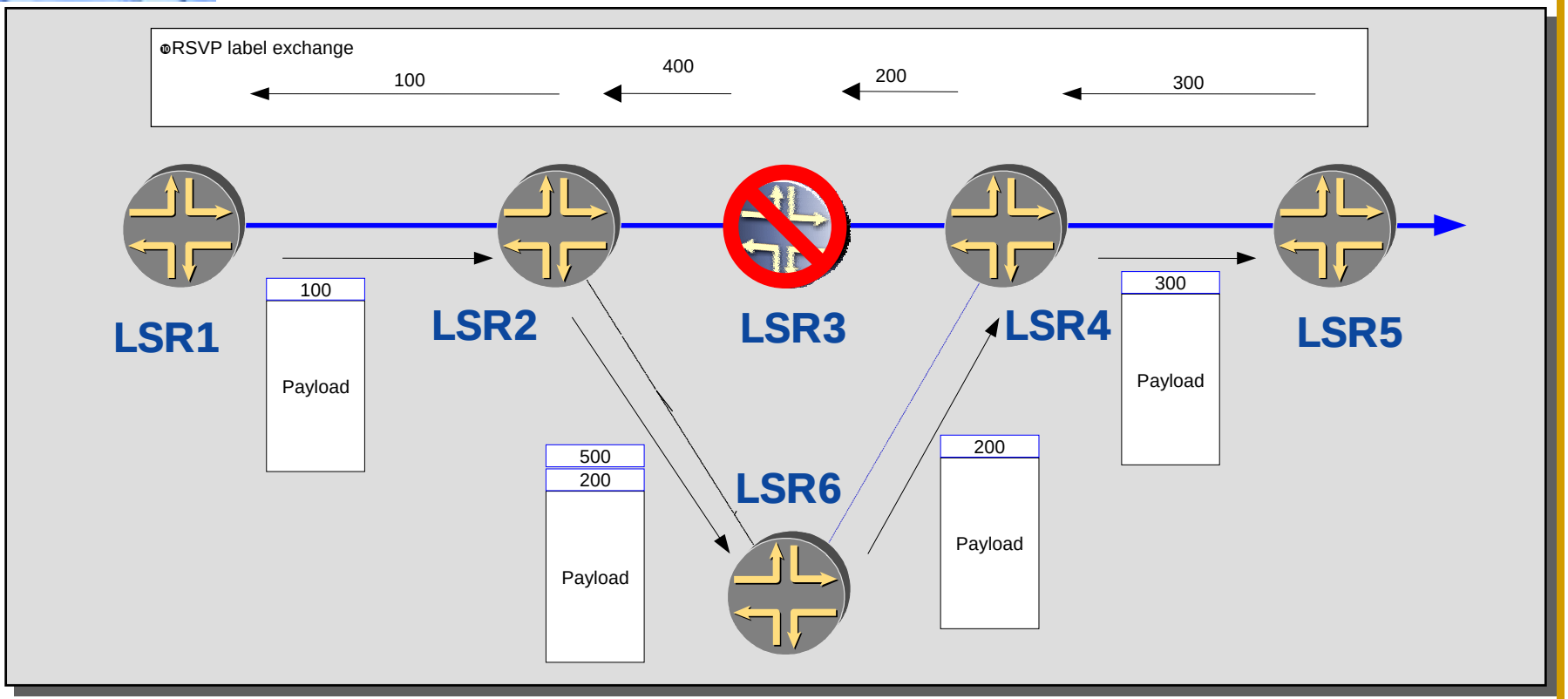
- Path protection (aka Secondary LSP)
- Local 1:1 protection (aka LSP Protection Fast Reroute)
- Local 1:N protection (aka Facility-based Fast Reroute)
  - 1:N Link Protection Fast Reroute
    - Protects only against link failure
  - 1:N Node Protection
    - Protects against both link failure and node forwarding plane failure
      - Using both Node Protection and Link Protection doesn't make sense

1:N Link Protection Fast Reroute



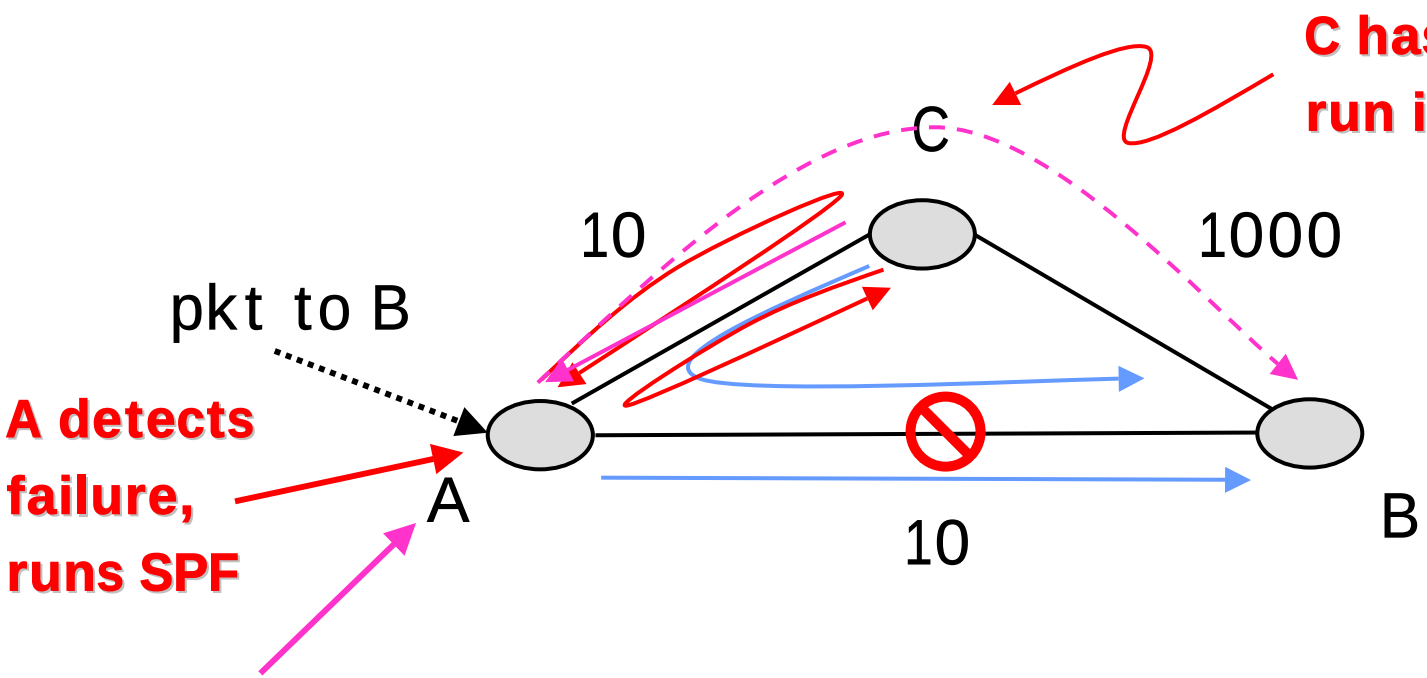
- Each LSR detects that an interface has gone down and reroutes all the Protected LSPs traversing the interface via the Bypass LSP
  - Recovery time is limited by the time to detect the failure
- Packet loss is minimized to the unlucky few that were transiting at the time of failure

1:N Node Protection Fast Reroute



- Each LSR detects that an interface has gone down and reroutes all the Protected LSPs traversing the interface via the Bypass LSP
  - Recovery time is limited by the time to detect the failure
- Packet loss is minimized to the unlucky few that were transiting at the time of failure

MPLS Fast Reroute vs IP



IP routing to B

New IP routing to B (transient forwarding loop)

MPLS detour to B (no loops)

Local 1:1/1:N Protection vs Graceful Restart

- In principle Local 1:1/1:N Protection could be applied to handle control plane restart, BUT
- Local 1:1/1:N Protection could (temporary) introduce jitter, while Graceful Restart has no impact on jitter
- Local 1:1/1:N Protection could (temporary) introduce out of order delivery, while Graceful Restart does not
- Local 1:1/1:N Protection could (temporary) introduce suboptimal routing, while Graceful Restart does not (as it does not alter routing)
- Local 1:1/1:N Protection does not handle PE control plane restart, while Graceful Restart does

Conclusion: Graceful Restart is more suitable than Local 1:1/1:N Protection for handling control plane restart

Data plane faults: Summary

- **Protecting against link failure:**
  - **Use MPLS 1:N Link Protection, or MPLS 1:N Node Protection, or MPLS 1:1 Protection**
  - **Doesn't help with faults on CE-PE links**
- **Protecting against node forwarding plane failure:**
  - **Use MPLS 1:1 Protection, or MPLS 1:N Node Protection**
  - **Doesn't help with forwarding plane failure on PEs**
- **MPLS Local 1:1/1:N Protection is not a substitute for Graceful Restart when handling control plane faults**
- **Not a perfect solution**
  - **Folks who are interested in a perfect solution should look elsewhere**
- **But a useful tool to deal with a specific set of problems**

Summary

- **Handling misbehaved CEs:**
  - **Rate limiting CE to PE control traffic**
  - **Limit memory usage on PEs by enforcing upper bound on the size of individual VRFs/VPLS Forwarding Table**
- **Handling control plane faults:**
  - **Graceful restart of control plane**
- **Handling data plane faults:**
  - **MPLS Path Protection**
  - **MPLS Local 1:1 Protection**
  - **MPLS Local 1:N Protection Protection**

Combined together enable to improve service availability for Provider Provisioned VPNs

Juniper® NETWORKS

Thank you!

http://www.juniper.co.jp
http://www.juniper.net