# Protocol Resilience

**Kireeti Kompella**
**Distinguished Engineer**
**Juniper Networks**

# Agenda

- Introduction
- Resilient architecture
- Improving data plane resilience
- Improving control plane resilience

# Robust IP Routing

- **Routers do *not* have to be flaky!**
- **Achieving 99.999% availability is definitely a challenge – but definitely achievable**
- **Achieving 5 9's is a requirement for many services (VoIP, VPNs, premium service, ...)**
  - **Wouldn't be bad for Public IP access as well**
- **We will see here some of the techniques that are being used to get there**

# Note

- This tutorial does not describe theoretical possibilities, but for the most part actual, implemented features
- To illustrate these concretely, specific examples from Juniper Networks' implementation are described
  - Other vendors have most of these features
- Juniper-specific information is highlighted with Slide Titles in Green


# IP Dependability

## IP Dependability

- Satisfies availability requirements
- Provides fault tolerance
- Ensures transparency to users
- Enables dependable services

Availability

Reliability

Recovery

**IP Dependability Addresses Many Dimensions**



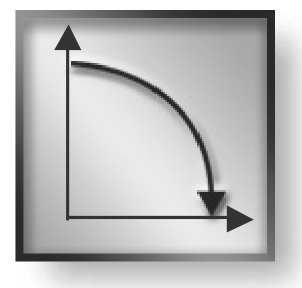


Juniper® NETWORKS


# Effects of Network Outage

- **Immediate Impact**
  - **Loss of Revenue**
  - **Repair Costs**
  - **SLA penalties**
  - **Dissatisfied customers**
  - **Project delays**
  - **Management distraction**
- **Long Term Impact**
  - **Damage to corporate brand**
  - **Customer churn, market share**
  - **Competition**
  - **Lawsuits**
  - **Lack of internal confidence**

**Financial**



**Market Share**

LIGHT READING

**Brand Damage**

# Strategy

- Robust protocol implementation
- Robust hardware
  - No single point of failure
  - Much to learn from CLASS 5 switches
- Respond quickly to data plane outages
- Respond gracefully to control plane outages
- Pause, re-evaluate, re-iterate

# Lessons Learned (M40)

- No redundancy: single Routing Engine
- Active backplane: single point of failure
- One switching plane: single point of failure
- Hot-swappable FPCs, but individual PICs not hot-swappable
- Other minor details

- Compare the M40 with the M40e!

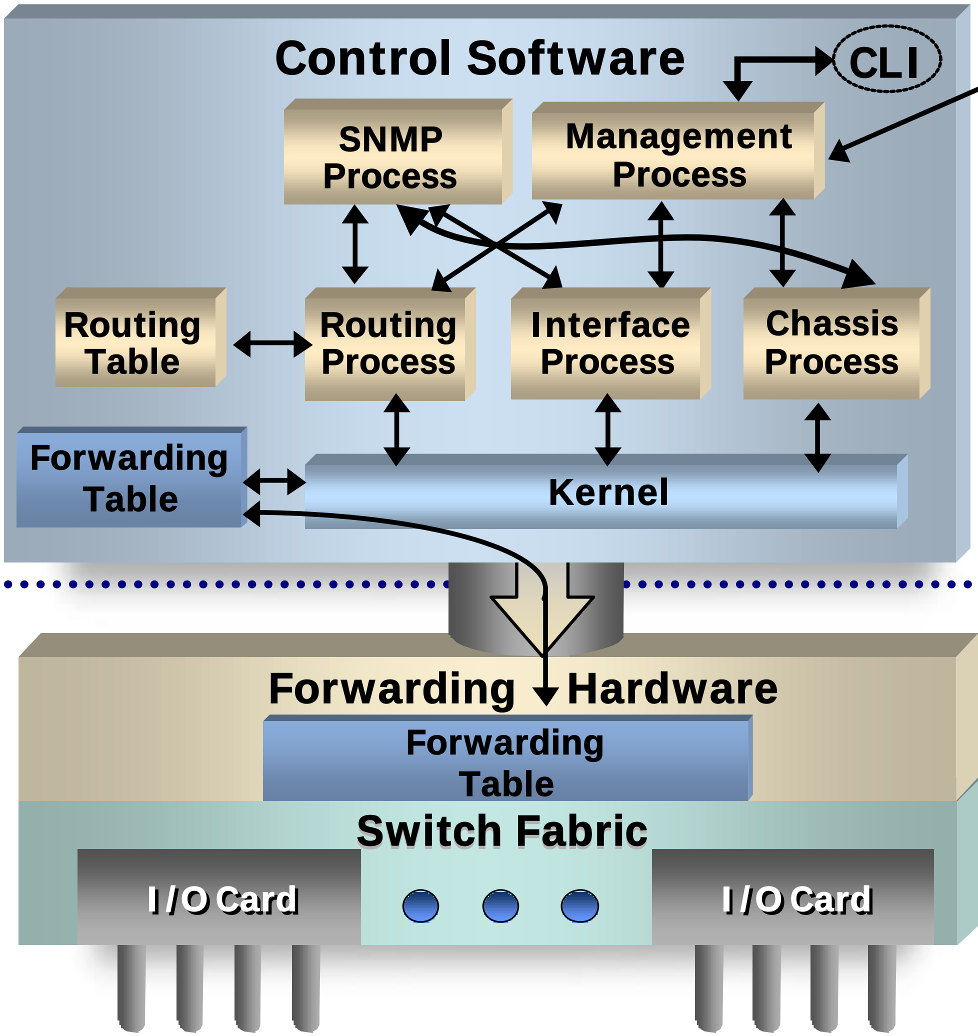

# Agenda

- Introduction
- Resilient architecture
  - Software design
  - Hardware resilience
- Improving data plane resilience
- Improving control plane resilience

# Resilient Architecture

## Control

**Routing Engine**




Modularization yields protection, fault isolation

## Forwarding

**Packet Forwarding Engine**

Separation of control and forwarding enables graceful restart techniques

# Software Design

- Purpose-built for
  - interoperability in heterogeneous environment
  - extensibility, easy addition of features
  - Reliability, fault isolation
- Lean and mean
  - Focus (e.g., on IP protocols)
- Modular architecture
  - Each module has independent resources


# No Software Modularization

- **One extreme: monolithic software**
  - **A bug anywhere can crash all software components**
  - **A bug in one component can affect another component (no fault containment)**
  - **A component can hog resources, affecting the performance of other components**
  - **A change in a component may have unpredictable results in the overall software**

# Full Modularization

- Every component is an "independent" module
  - Laudable goal, but …
  - unfortunately, components do need to interact
  - If interaction occurs often, it may have a high communication cost
  - Fine-grained locking and synchronization among modules also has a cost
- Secondary problem: mix-and-match of module versions can be a nightmare

# Tactical Modularization

- **Modularize those components that interact reasonably infrequently**
  - **Communication, locking and synchronization do not cost much**
- **Modularize on functional boundaries**
  - **Function separation usually Þ less interaction**
- **Modularize further when needed for some specific (tactical) reason**
  - **Performance, independence, …**

# Routing Protocol Daemon

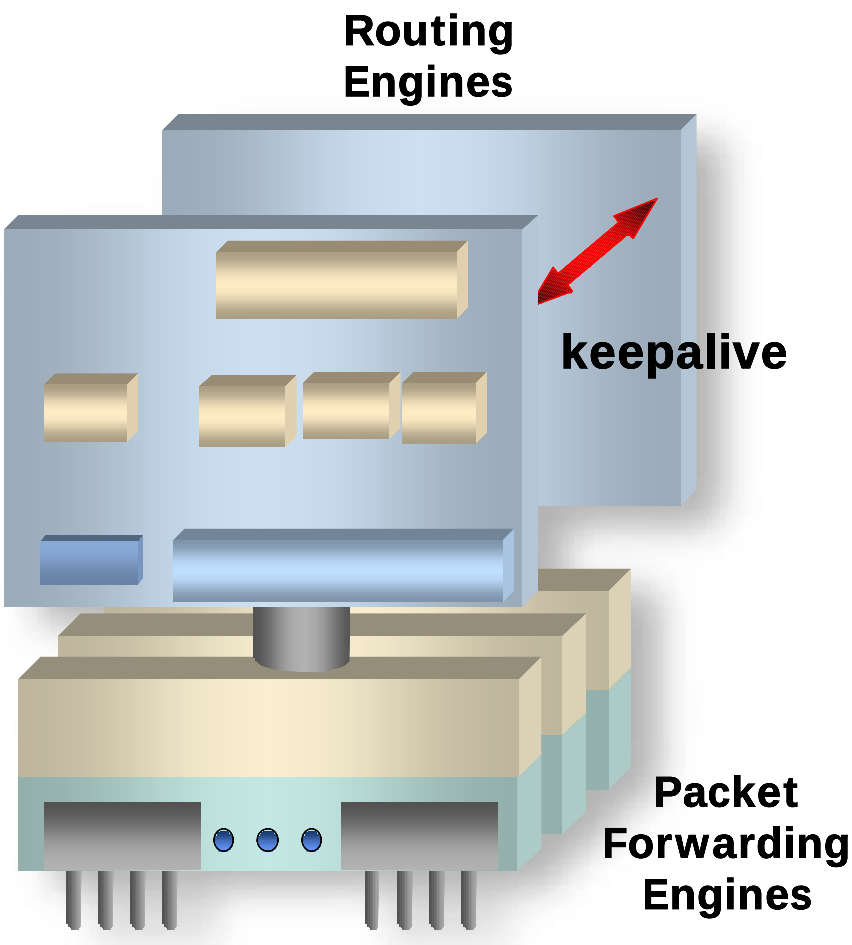
- RPD was a single software module
- Still mostly is …
- … but transport of IS-IS and OSPF packets is now handled by a different module
  - Achieves a measure of independence between IS-IS/OSPF and other routing protocols
  - Especially useful for the “hello” packets

# Hardware Resilience

increasing down time

- Redundant units
  - Automatic failover
  - Hot, warm or cold restart of software
- Hot swappable components
- Field-replaceable units

# Control Plane Failover



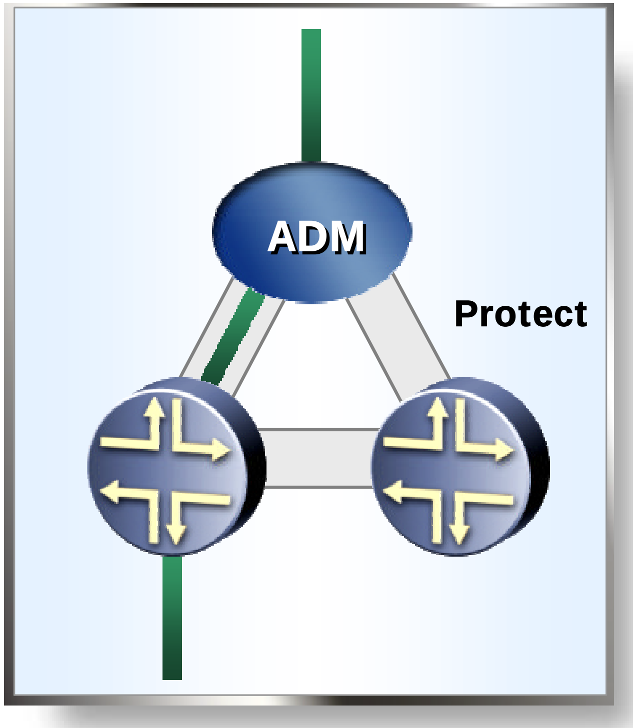

- Redundant Routing Engines with keepalives running between them
- Automatic failover to secondary, followed by cold, warm or hot restart
- Configuration synchronized between RE's

# Agenda

- Introduction
- Resilient architecture
- Improving data plane resilience
  - Layer 1 protection
  - VRRP
  - Faster layer 3 convergence
  - MPLS protection
- Improving control plane resilience

# SONET / SDH Protection

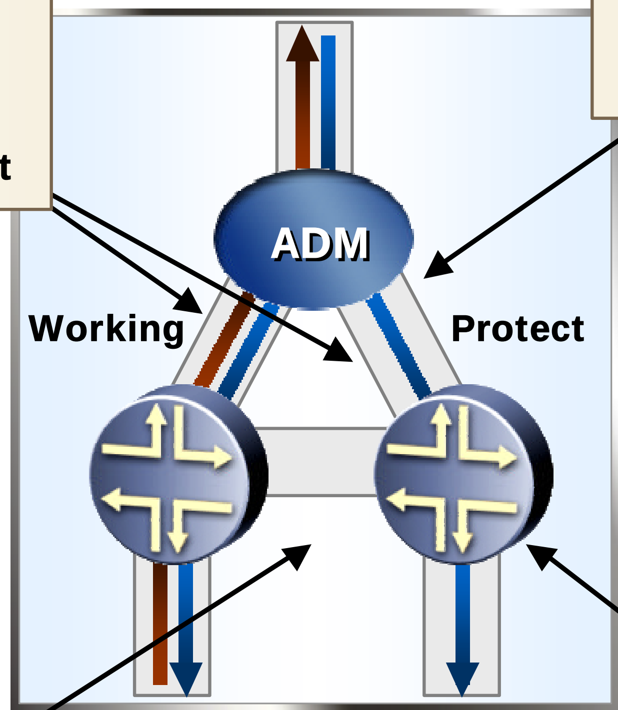
- SONET APS & SDH MSP
  - Redundant routers share uplink
- Rapid circuit failure recovery
  - Used on router-to-ADM links
  - 50 ms at physical layer
  - Layer 3 protocol convergence longer
- Interoperable with standard ADM
- Working & protect circuits
  - May reside on different routers
  - May reside on same router



# Dual Router APS Operation

Inbound traffic sent over both working and protect circuits; outbound traffic sent only over working circuit

ADM signals status over the protect circuit via K1/K2 overhead bytes



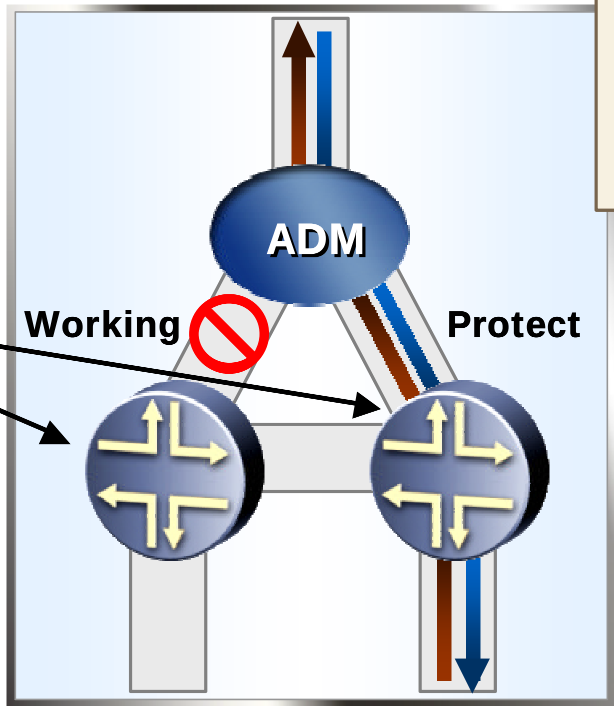

Separate reachability protocol used to signal between the two routers

Router on protect circuit signals back the desired protect/working configuration

# Dual Router APS Operation

If the ADM detects a fault in the working circuit, it starts receiving data over protect circuit instead and signals change to router. (Router can also request ADM to switch to protect circuit.)

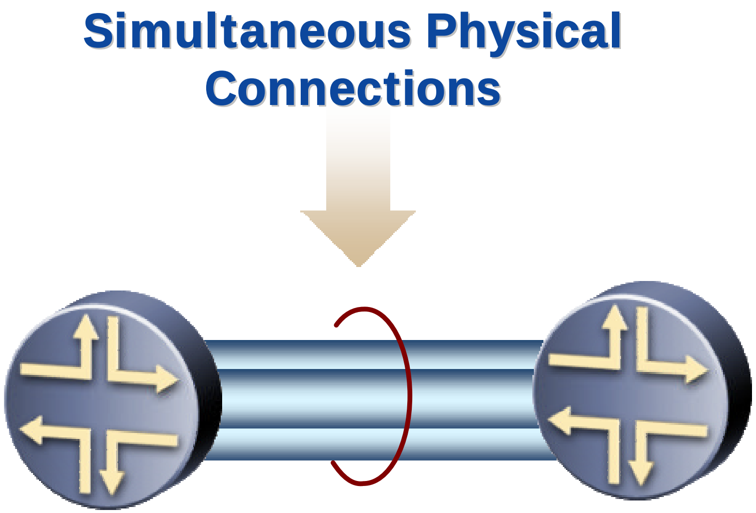
Once the circuit has switched, routing protocols converge on new topology




Note: Both revertive and non-revertive modes are supported for switching back when working circuit is available again

# Link Redundancy

- **Dependable Links**
  - **Link failure does not affect forwarding**
  - **Load redistributed among other members**
- **Various mechanisms**
  - **MLPPP – T1/E1 bonding**
  - **802.3ad – Ethernet aggregation**
  - **SONET/SDH aggregation**

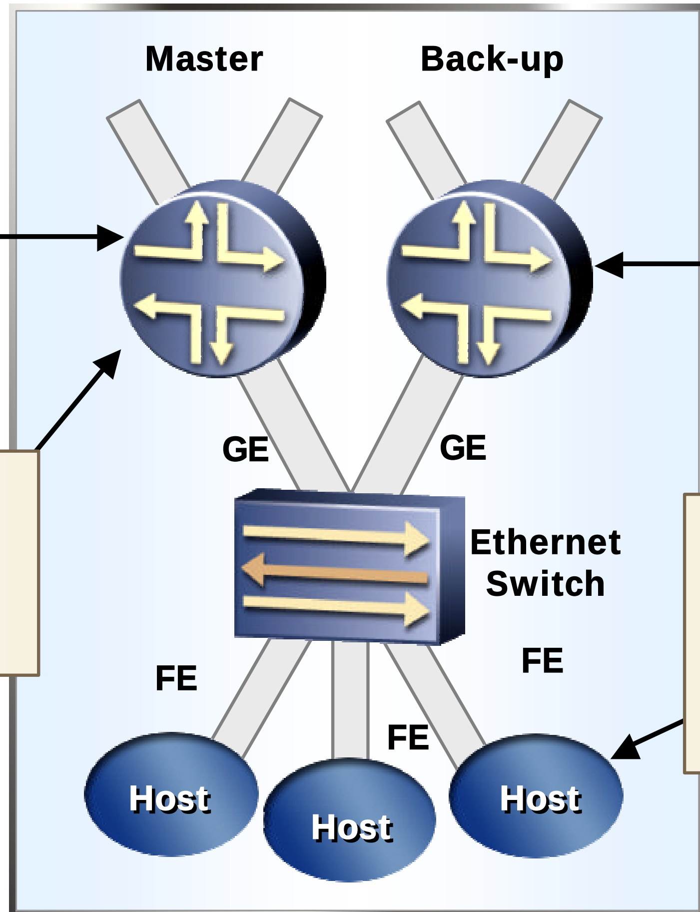


# Virtual Router Redundancy

- Redundant default gateways - VRRP (RFC 2338)

Multiple routers on the subnet negotiate who will be “Master” and own the Virtual Router IP Address.

Master sends periodic hellos to communicate alive state.

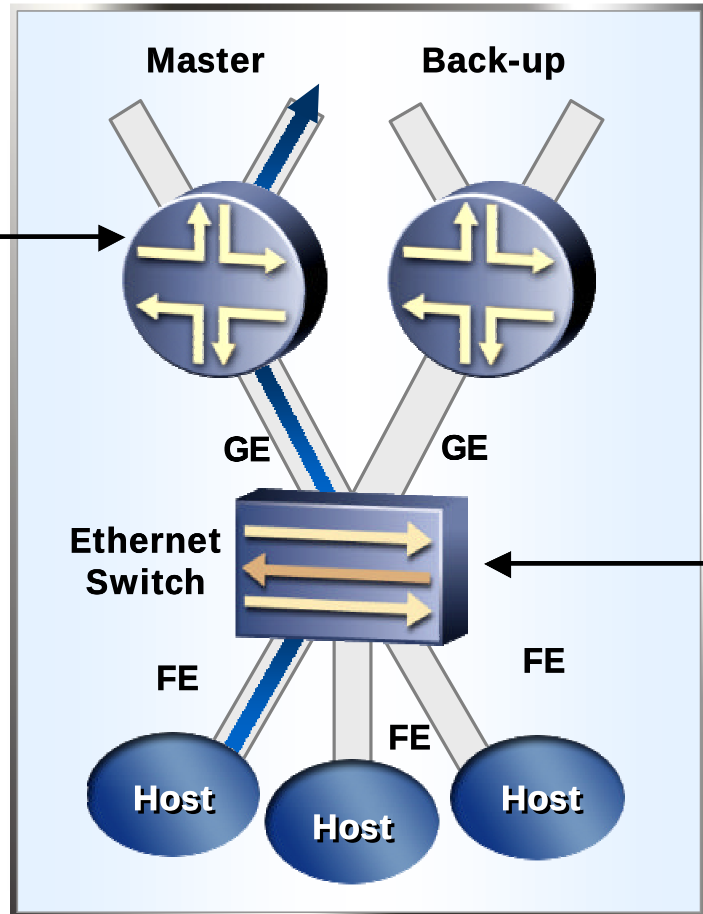



All other routers are backups. Backup priority is configurable

Hosts are preconfigured with Virtual Router IP address as default for traffic exiting the LAN

# VRRP Operation (1)

The owner of Virtual IP Address (ie “Master”) responds to ARP requests from hosts. Sends Virtual Router MAC Address.

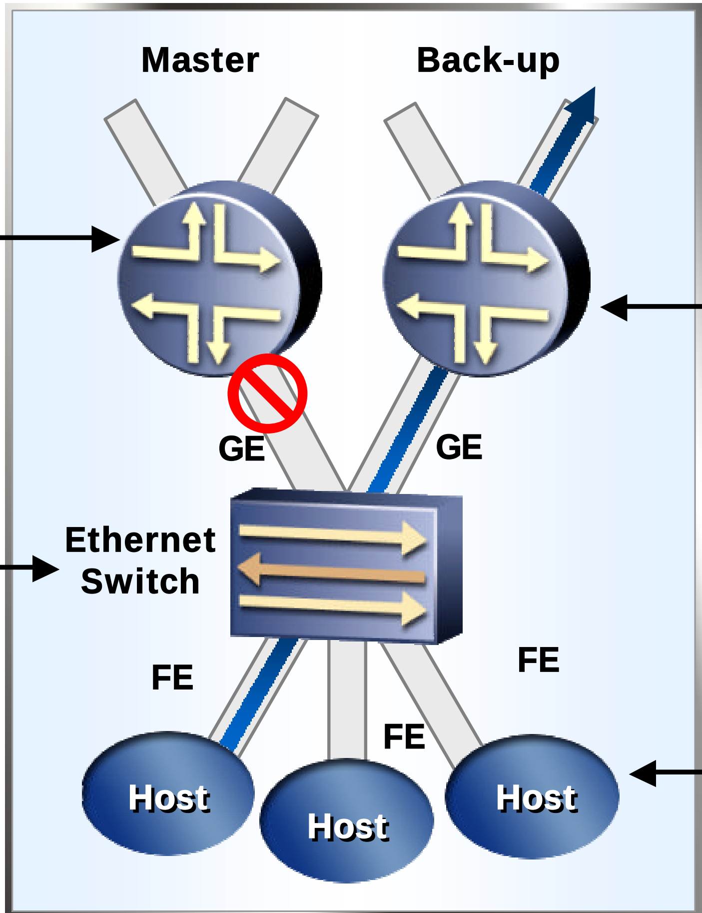



Switch learns location of Virtual Router MAC Address and forwards outbound traffic in that direction.

# VRRP Operation (2)

If current master fails, a backup router will detect condition and become the new Master and owner of Virtual Router IP and MAC address. Note: No change to Virtual Router MAC address.

Switch learns location of new Virtual Router and forwards traffic accordingly.




New master sends a “gratuitous ARP” to allow the switch to learn new location of Virtual MAC address.

Change of Virtual Router is transparent to hosts.

# Faster Router Convergence

- Faster convergence improves Network Dependability
  - Separation of control & forwarding planes is key
  - Protocol expertise is key

| Features | Benefits |
|---|---|
| High Priority Flooding for LSPs that trigger an SPF | ◆ Timer reduced from 100 to 20msec<br>◆ Faster propagation of major changes |
| Quick SPF Scheduling | ◆ Reduces time from 7 sec to 50 msec<br>◆ Speeds calculation of optimum path |
| RIB/FIB Enhancements | ◆ Faster updates of routes that share nexthops |
| E-BGP Peer Groups | ◆ Shared updates to peers with common policies<br>◆ Better handling of subsetting of a group |

# ISIS/OSPF Convergence

- Current implementation:
  - 5s hold-down timer between SPF computations
  - 100ms transmit interval between LSA updates
- ISIS LSPs that cause a router to trigger an SPF locally are put in a high priority queue for flooding
- Performs up to 3 SPFs back-to-back (separated by the spf-delay), then hold-down (5 secs)
- Configurable spf-delay instead of being hard coded to 1 sec (default is 1 sec)
  - The range for spf-delay is between 50 and 1000 msec


# Faster ISIS SPF

kireeti@pro1-a# edit protocols isis

[edit protocols isis]

kireeti@pro1-a# set interface so-0/0/0 ?

Possible completions:

lsp-interval — Interval between LSP transmissions (milliseconds)

kireeti@pro1-a# set spf-delay ?

Possible completions:

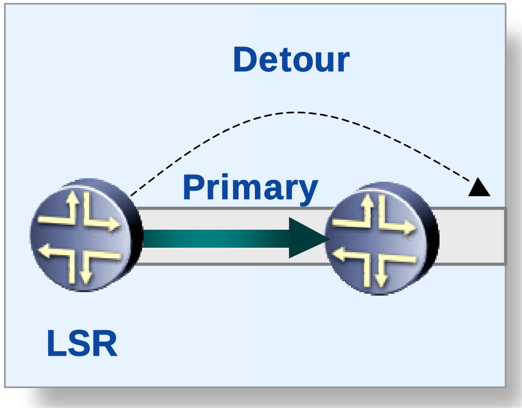
<spf-delay> — Time to wait before running an SPF (milliseconds)


# Faster OSPF SPF

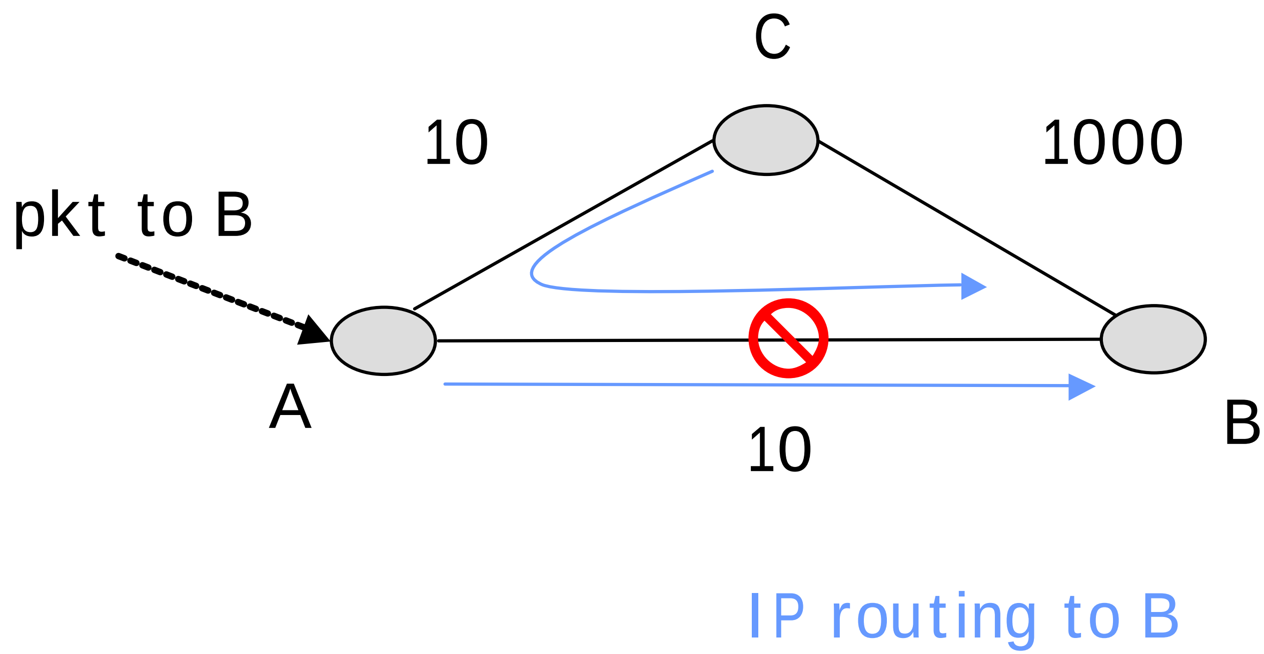
```
kireeti@pro1-a# edit protocols ospf
[edit protocols ospf]
kireeti@pro1-a# set spf-delay ?
Possible completions:
  <spf-delay>         Time to wait before running an SPF
                      (milliseconds)
kireeti@pro1-a# edit protocols ospf3
[edit protocols ospf3]
kireeti@pro1-a# set spf-delay ?
Possible completions:
  <spf-delay>         Time to wait before running an SPF
                      (milliseconds)
```

# MPLS-based Protection

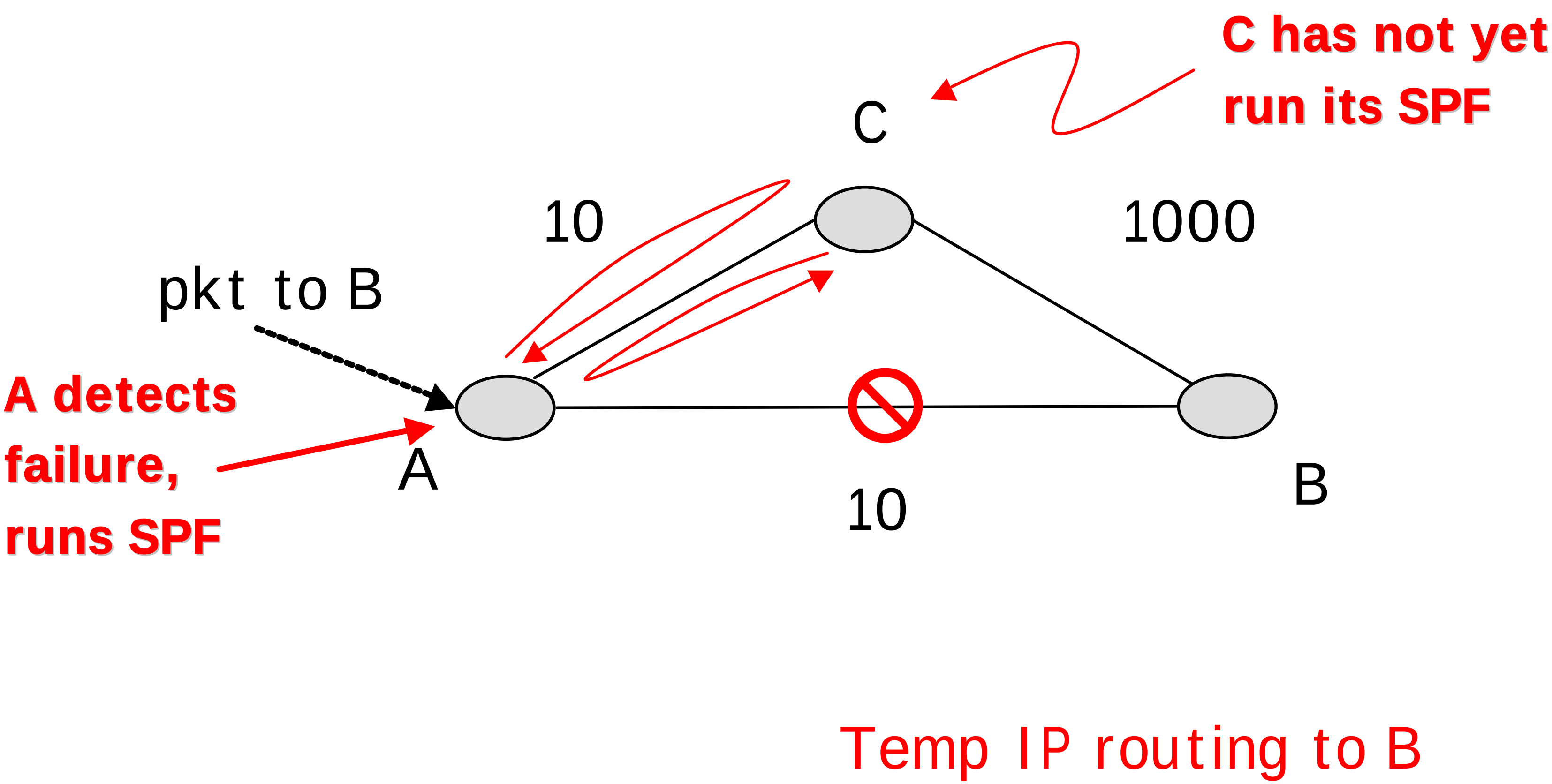
- Increasing demand for “APS/MSP-like” redundancy
  - Cost of APS/MSP protection
  - Convergence time
- Solution: MPLS protection
  - Secondary LSP paths
  - LSP Protection Fast Reroute
  - Link Protection Fast Reroute



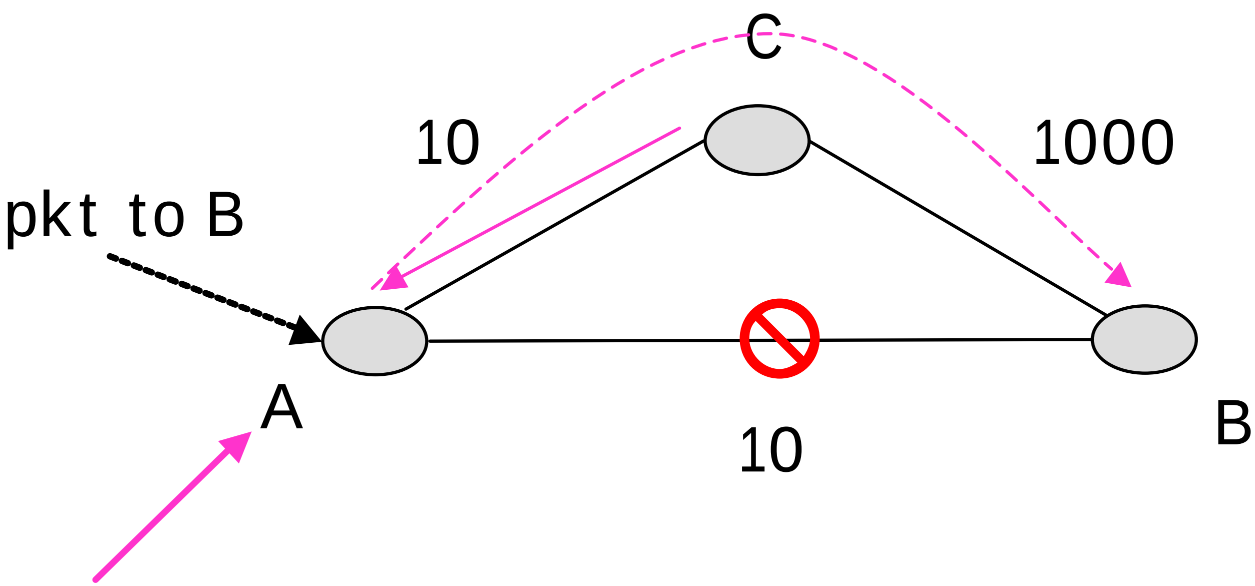
# MPLS vs. IP Convergence



# MPLS vs. IP Convergence



# MPLS vs. IP Convergence




**A detects failure, reroutes pkts to B into LSP (works independent of B)**

MPLS detour to B

# RSVP-TE LSPs

- An RSVP-TE LSP is usually point-to-point, and usually has constraints
  - Bandwidth, colors, ERO (Explicit Route Object)
- An RSVP-TE LSP is a container: a tunnel from A to B for traffic of a certain class
  - The LSP may have multiple instantiations, each with its own route
  - Each such instantiation is called a path in Juniper terminology

# Secondary LSP Paths

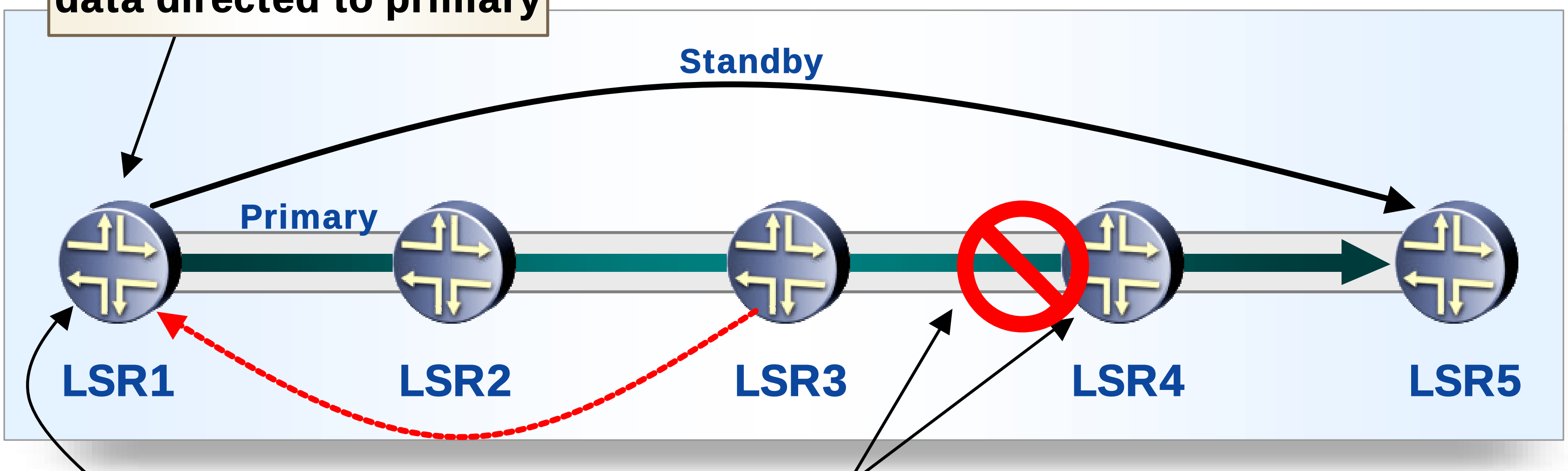
- Primary path is the preferred path to set up and use for data transmission
- Secondary paths are alternatives, to be used when the primary fails
  - Standby secondary paths are pre-signaled
- Switching to secondary requires propagating error to head end, which must signal secondary, then change data flow
  - Switching to standby requires propagating error to head end – no signaling needed

# Signaling Errors in RSVP-TE

- RSVP-TE uses standard RSVP messages to signal errors (PathErr, ResvErr)
  - These go hop-by-hop to head end or tail end
  - Each hop must process error, then forward it
- Thus, response time to an error depends on distance in hops from head end
- New RSVP message in GMPLS spec called Notify that is sent directly to head end
  - Response time is detection time + transmission time + forwarding time

# Using Secondary LSP Paths

Head end of LSP, with primary and standby; data directed to primary




Head end gets error message, switches data flow to standby

When an error occurs here (either link or LSR4), upstream LSR (LSR3) detects the error and signals it back to the head end

# Configuring LSP Paths

\# Define LSP foo with primary path bar

kireeti@pro1-a# edit protocols mpls

[edit protocols mpls]

kireeti@pro1-a# set label-switched-path foo <details of lsp foo>

kireeti@pro1-a# set label-switched-path foo primary bar

kireeti@pro1-a# set path bar <details of path bar>

\# Add secondary path fubar to LSP foo

kireeti@pro1-a# set label-switched-path foo secondary fubar

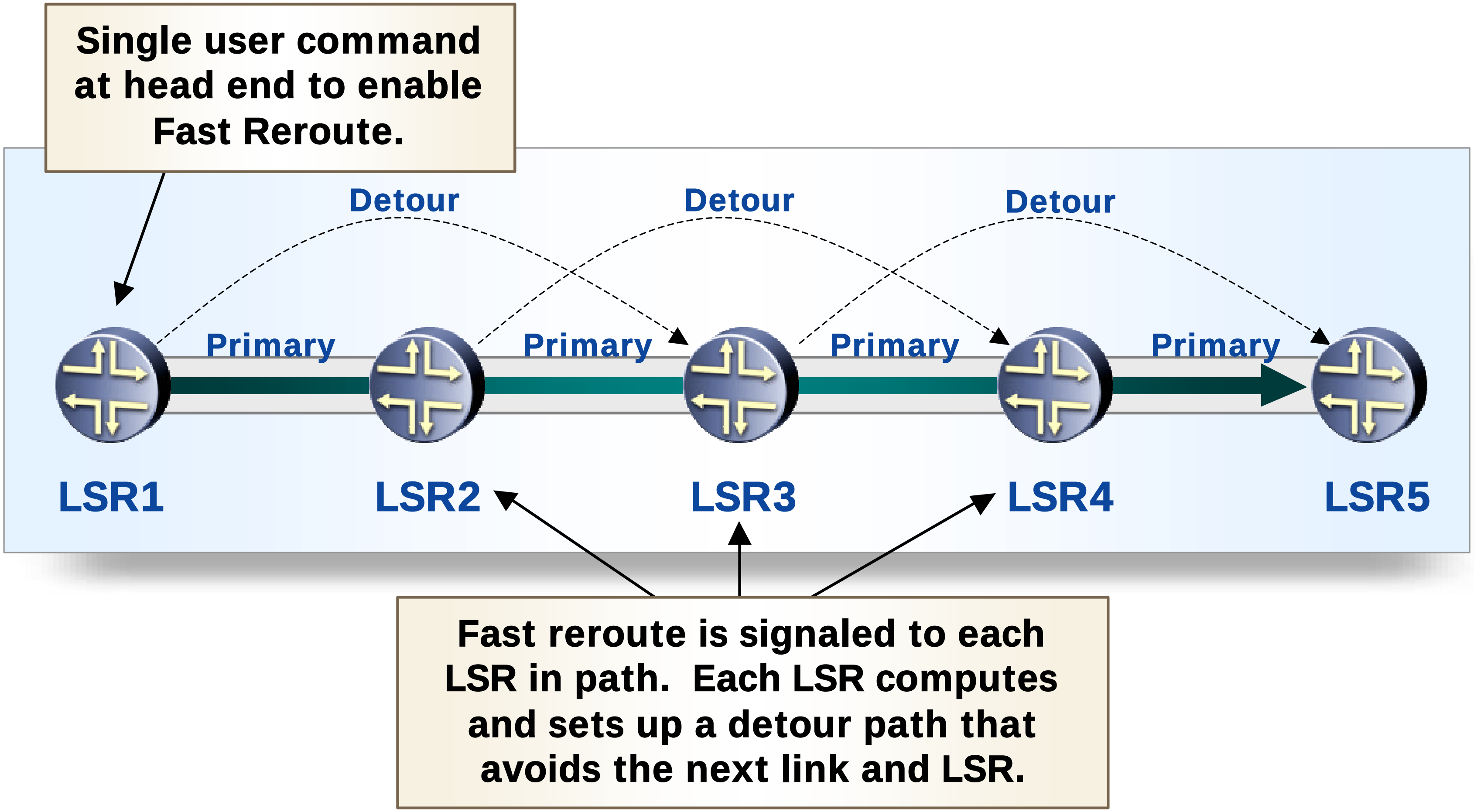
\# Add standby path fubar2 to LSP foo

kireeti@pro1-a# set label-switched-path foo secondary fubar2 standby

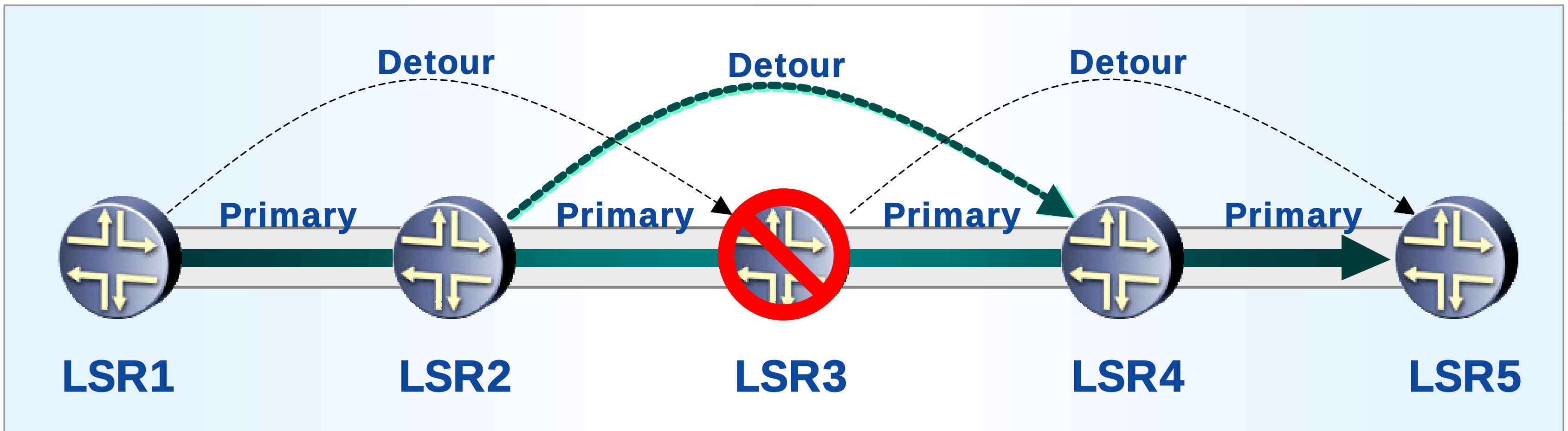
# Two Styles of Fast Reroute

- LSP Protection Fast Reroute
  - Each LSP is protected by an individual detour
  - 1:1 style protection
  - Better bandwidth guarantees
- Link Protection Fast Reroute
  - Each link protected by a bypass LSP
  - Each LSP on that link protected by bypass LSP
    - Uses label stacking
  - Bandwidth guarantees harder to meet
  - Fewer overall LSPs

# LSP Protection FRR Operation



# LSP Protection FRR Operation

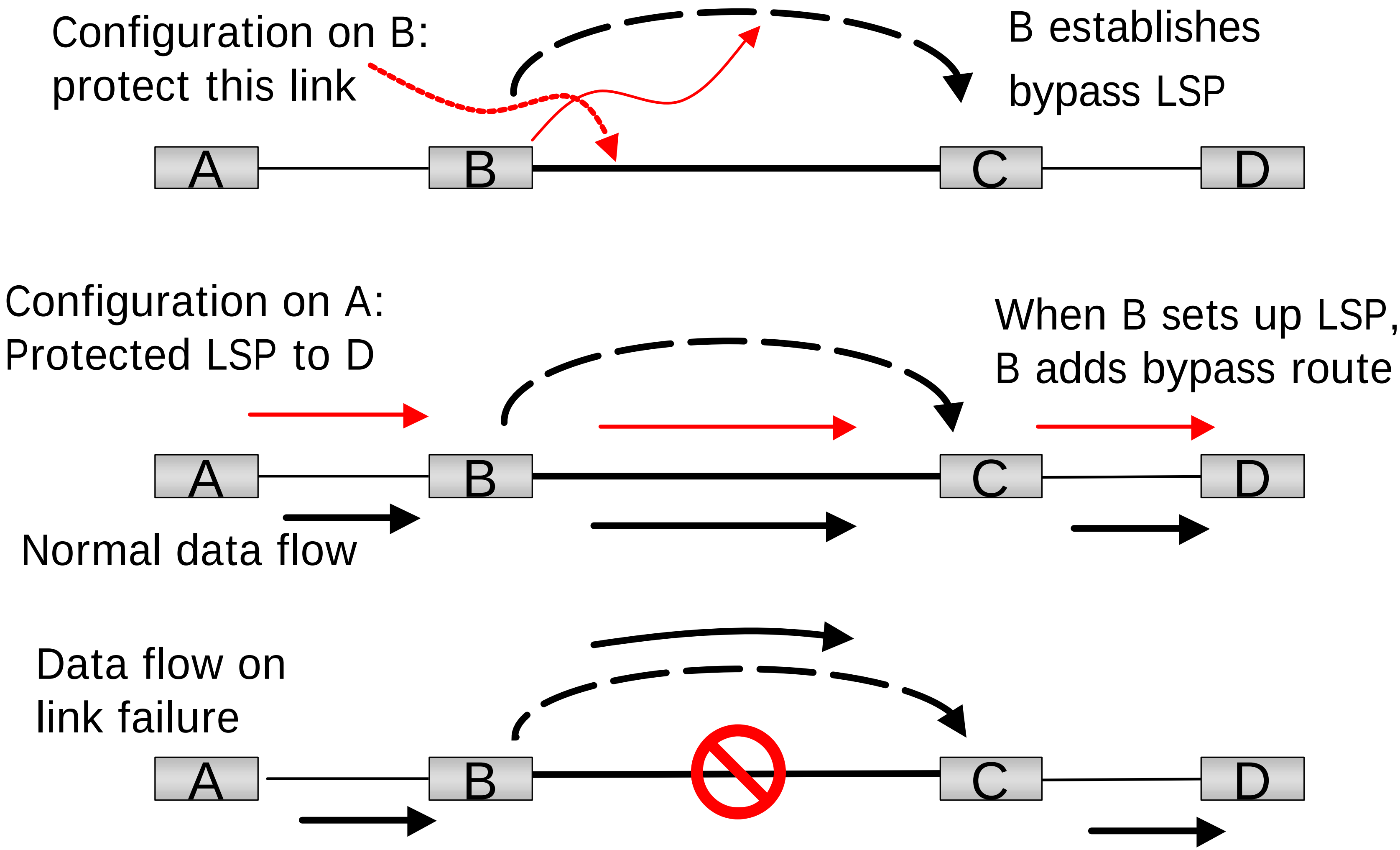



- Upstream LSR detects quickly that an interface in an LSP has gone down and reroutes via detour
- LSR also sends an error back to the head end to switch to a backup path – detour generally only used for a few seconds

# Link Protection FRR Operation

- **Bypass LSP**
  - LSP used to protect a set of LSPs on a protected link
  - During link failure, packets use bypass LSP via label-stacking
- **Protected LSP**
  - LSP being protected; associated with a bypass LSP at a protected link
- **Backup LSP**
  - LSP established after link outage; used to maintain the protected LSP over a bypass LSP

# Link Protection FRR Operation



# Configuring Link Protection

**# Protect interface so-0/0/0 on pro1-b**

kireeti@pro1-b# edit protocols rsvp

[edit protocols rsvp]

kireeti@pro1-b# set interface so-0/0/0 ?

Possible completions:

...

➤ link-protection        Protect traffic with a label-stacked LSP

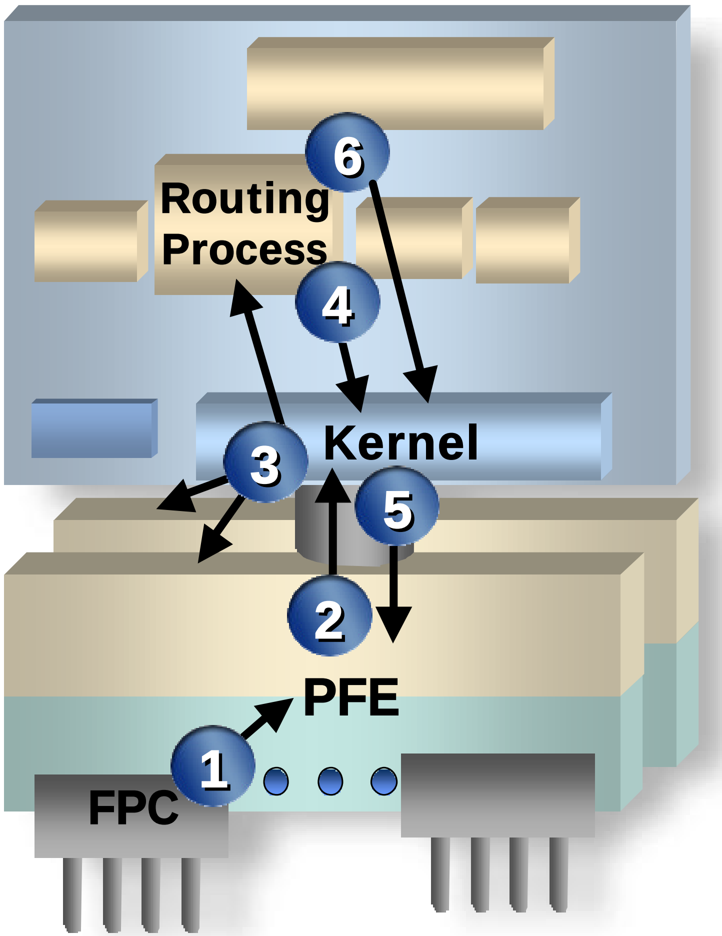
**# Establish a link-protected LSP called foo on pro1-a**

kireeti@pro1-a# edit protocols mpls

[edit protocols mpls]

kireeti@pro1-a# set label-switched-path foo link-protection

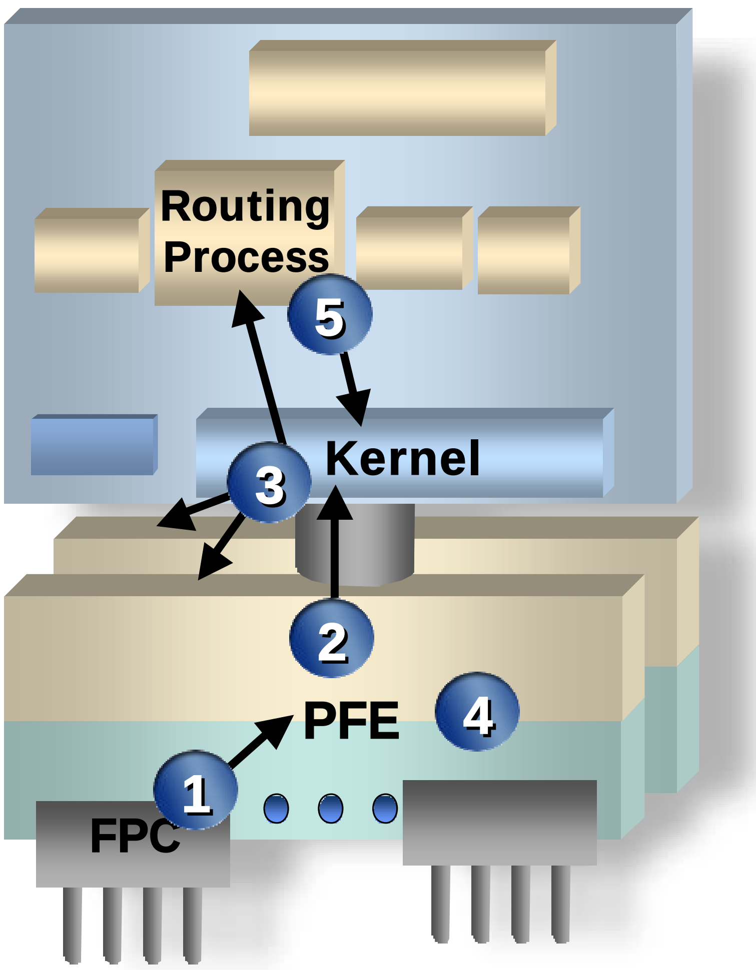
# Fast Reroute




1. FPC informs PFE that the interface X went down
2. PFE passes that message on to the kernel
3. The kernel informs RPD and other PFEs that interface X went down
4. RPD sends backup routes to the kernel
5. The kernel sends updates to the PFE
6. RPD computes new path

Knowledge of backups is in RPD

# Express Fast Reroute




1. **FPC informs PFE that interface X went down**
2. **PFE passes message on to kernel**
3. **Kernel informs RPD and other PFEs that interface X went down**
4. **PFE sees interface X is down and switches to backup routes**
5. **RPD computes new path**

Knowledge of backups is in PFEs


# Agenda

- Introduction
- Resilient architecture
- Improving data plane resilience
- Improving control plane resilience
  - Graceful restart

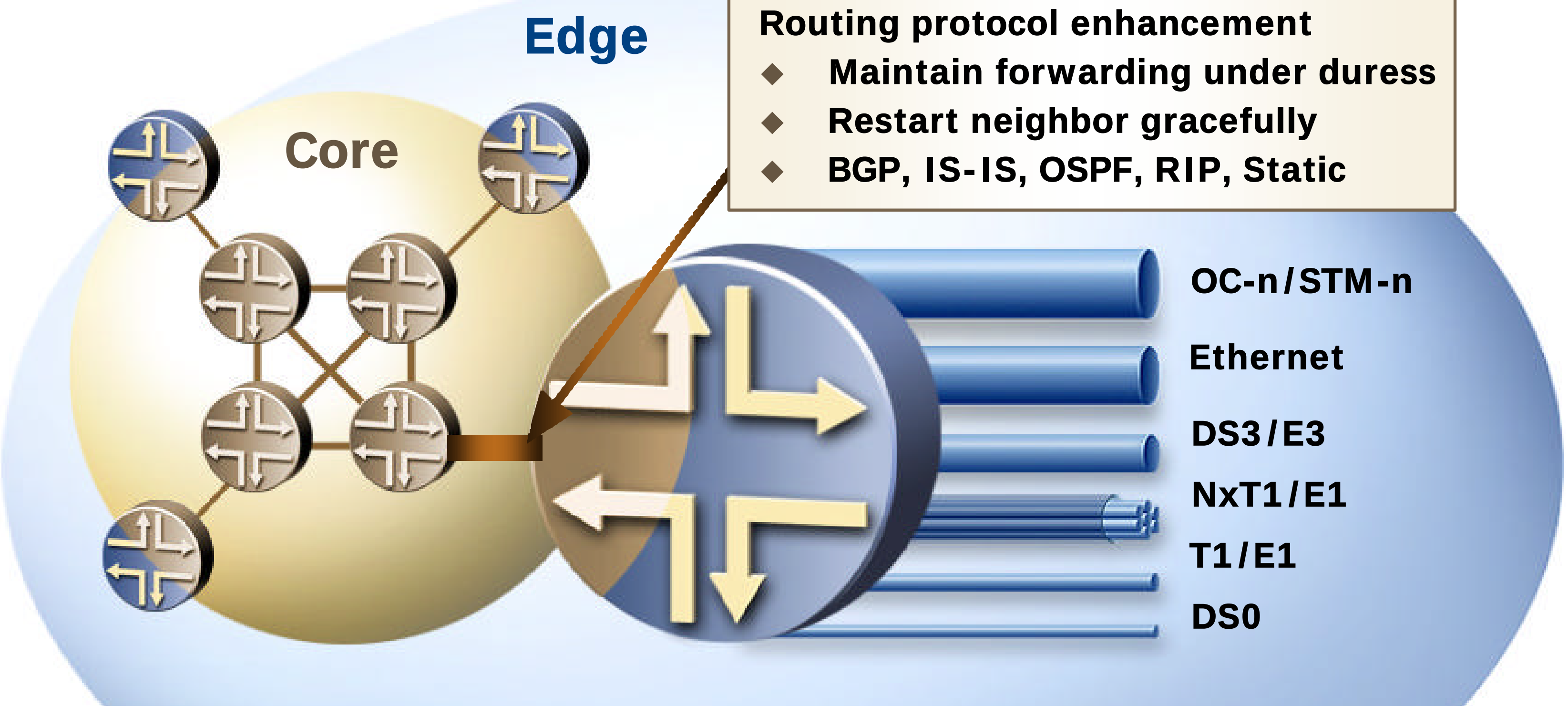


Juniper NETWORKS

# Reliability Paradox

- Software is buggy …
  - … so add lots more code to make it reliable!
  - (applies to a lesser extent to hardware as well)
- General purpose High Availability is hard to achieve (and expensive!)
- For routing protocols, though, the problem is easier
  - Saving state is mostly not necessary, as one's peers usually have the most up-to-date state
  - This assumes cooperation from one's peers

# Protocol Graceful Restart

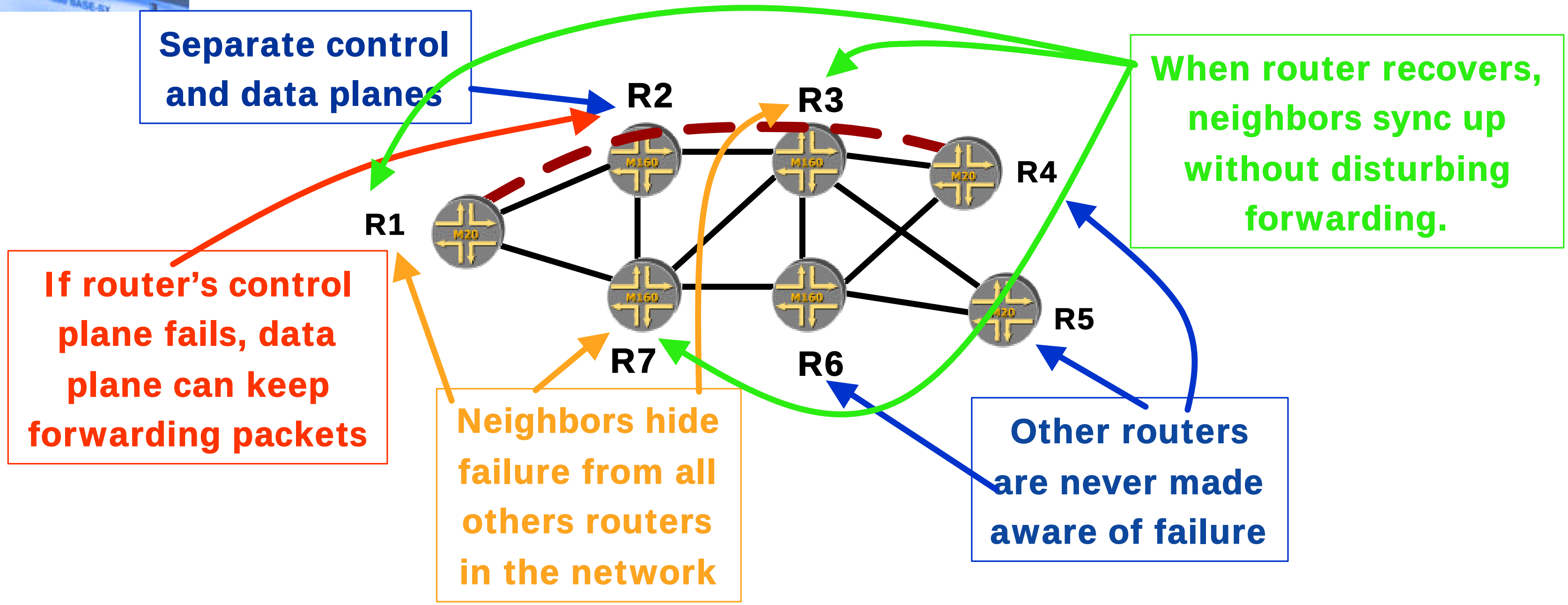



**Routing protocol enhancement**

- Maintain forwarding under duress
- Restart neighbor gracefully
- BGP, IS-IS, OSPF, RIP, Static

- Protocol extensions distinguish between control, data plane failure
- Protects against Routing Protocol Module failure
- Failure invisible to everyone but peers
- Stepping stone for non-stop forwarding

# Protocol Graceful Restart




- **New concept only feasible when router has separate control and data planes**
- **Basic problem is not recovering state, but avoiding any drastic action when you learn neighbor has gone down**
  - **Tearing down adjacency**
  - **Ripping out routes**

# Basic Issue

- Router X goes down
  - Peers may not yet realize this
- Router X comes back up, sends hello (IS-IS, OSPF) or resets TCP connection (BGP, LDP)
  - At this point, peers do know
  - Peers (or X itself) inform rest of routers not to use X, so all routers recompute routing
- Router X re-establishes synchronization
  - All other routers start swinging routes back

# Solution

- If X can in fact maintain *forwarding* state across failure, keep the fact that X crashed a family secret: don't tell rest of network
  - X must indicate its ability to do graceful restart
- When X comes back up, help X to retrieve *protocol* state and achieve synchronization
  - IS-IS, OSPF: give X the LSDB (as usual)
  - BGP: send X the routes you normally would
- When X is back in sync, X updates forwarding table to clean out stale info

# Goal

- Goal here is not to eliminate network redundancy – still need alternative paths
- Goal here is to eliminate
  - Control plane churn as everyone in the network responds to one router restarting
  - Data plane churn as traffic shifts, then shifts back
  - Predictability of traffic patterns, delay, jitter, ...

# Concerns

- Backward compatibility with non-graceful restart capable routers
  - X may not be graceful restart capable
  - X's helpers may not be graceful restart capable
- X may have failed in a way that doesn't preserve forwarding state (lost power)
- X may not recover (caught fire)
- X (or helper) may start graceful restart procedures but never complete

# Concerns

- X has stale routes – could lead to routing loops
  - Alternative: blackhole traffic while resyncing
  - Escape clause in OSPF: abort Graceful Restart if topology changes during the process
- X must be careful not to send intermediate updates, as this defeats the purpose
  - E.g., establish all adjacencies before sending out link state update
  - Ideally, advertisement is identical to pre-restart
  - Inter-protocol dependencies

# GR-aware Protocols

- Graceful restart mechanisms defined for IS-IS, OSPF, BGP, LDP, RSVP-TE
  - Two mechanisms for OSPF
    - However, majority seem to have converged on one
  - Two mechanisms for LDP
    - One requires check-pointing and saving state
    - The other is more in line with other GR mechanisms
- Some new protocols are thinking in terms of Graceful Restart from the beginning
  - E.g., Link Management Protocol

# BGP Graceful Restart

- Router that is restarting called the "Restarting Speaker"
- A peer of Restarting Speaker that is Graceful Restart capable is called a "Receiving Speaker"
- BGP graceful restart is done on a per-RIB (AFI+SAFI) basis – one may restart one RIB gracefully, and another not

# Restarting Speaker

- Sends new capability indicating that it is Graceful Restart capable (*before* restarting)
- After restarting, checks whether forwarding state was preserved (assume so); marks all routes in the forwarding table as stale
- Sends new capability again, indicating that forwarding state was preserved
- Establishes sessions with all peers

# Restarting Speaker

- Receives routes from each peer, followed by a marker saying “Done!” (End-of-RIB)
- (Waits for IGP to converge, do SPF)
- Performs path selection, updates forwarding state, removes any remaining stale routes
- Advertises routes to peers, followed by End-of-RIB marker
- Drops back to normal operation

# Receiving Speaker

- Detects peer is down, closes old session (does *not* send a Notification)
- Marks forwarding state received from old peer session as stale if peer had previously advertised being GR-capable
- When new session comes up, check that peer actually preserved forwarding state – if not, delete stale routes from peer
- Send routes to peer, followed by End-of-RIB marker

# Receiving Speaker

- Wait for Restarting Router to send its routes, updates forwarding state
- When End-of-RIB marker is received, removes all remaining stale routes
- Drops back to normal operation

# OSPF “Hitless” Restart

- Hitless == graceful
- Restarting router == restarting speaker
- Helper == receiving speaker

# Restarting Router

- Generate a "Grace-LSA" on all interfaces
  - Before restarting, if planned restart
  - After, if unplanned.  In this case, Grace-LSA MUST be sent before any hello
- Mark OSPF forwarding entries as stale
- Do *not* flush any received, self-generated "pre-restart" LSAs.  Do *not* generate LSAs
- Send hellos, acquire database as usual
- When all adjacencies are up, exit GR mode

# Restarting Router Done

- Generate router and network LSAs based on current info
- Do SPF, update forwarding based on the results, flush remaining stale routes
- Generate other LSAs
- Flush any remaining pre-restart LSAs, including Grace-LSAs

# Helper Operation

- If Grace LSA received from a neighbor, keep neighbor's state as Full (for a while)
- Help neighbor acquire database (as usual)
- Meanwhile, continue to generate LSAs based on current information
- When done (e.g., restarting router flushed Grace-LSA), re-originate all LSAs based on current state

# Inter-protocol Timeline

- All protocols execute their graceful restart procedures
- IGPs acquire LSDB, do SPF and are done
- LDP needs IGP SPF for route selection
- BGP needs to get “End-of-RIB” from all peers, get IGP SPF, and do path selection. Then, BGP can advertise and install routes
- RSVP operates independently (may need TE database from IGPs)
  - Note: No RSVP Graceful Restart at ingress (yet)

# Graceful Upgrade

- Initial target of graceful restart: controlled use for scheduled maintenance
- Shorter service affecting windows
  - Maintenance window may remain as long, but service disruption is shorter
- As experience and confidence increases, the graceful restart concept may be used for unplanned outages

# IS-IS Graceful Restart

```
kireeti@pro1-a# edit protocols isis
[edit protocols isis]
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  disable              Disable graceful restart
  helper-disable       Disable graceful restart helper capability
  restart-duration     Maximum time for graceful restart to finish
                       (seconds)
```


# OSPF Graceful Restart

```
kireeti@pro1-a# edit protocols ospf
[edit protocols ospf]
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  disable            Disable OSPF graceful restart capability
  helper-disable     Disable graceful restart helper capability
  notify-duration    Time to send all max-aged grace LSAs
                     (1..3600 seconds)
  restart-duration   Time for all neighbors to become full
                     (1..3600 seconds)
```

# LDP Graceful Restart

```
kireeti@pro1-a# edit protocols ldp
[edit protocols ldp]
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  disable              Disable graceful restart
  helper-disable       Disable the graceful restart helper capability
```


# BGP Graceful Restart

```
kireeti@pro1-a# edit protocols bgp
[edit protocols bgp]
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  <[Enter]>            Execute this command
  disable              Disable graceful restart
  restart-time         Restart time used when negotiating with a
                       peer (1..600)
  stale-routes-time    Maximum time for which stale routes are
                       kept (1..600)
```

# RSVP Graceful Restart

```
kireeti@pro1-a# edit protocols rsvp
[edit protocols rsvp]
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  disable              Disable RSVP graceful-restart capability
  helper-disable       Disable graceful restart helper capability
```

# Graceful Restart of RIP

- No protocol changes
- However, can still apply graceful restart techniques to RIP
  - Need to configure this
  - Also need to configure timeout

# RIP Graceful Restart

```
kireeti@pro1-a# edit protocols rip
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  <[Enter]>          Execute this command
  disable            Disable graceful restart
  restart-time       Time after which RIP is declared out of
                     restart (1..600)
kireeti@pro1-a# edit protocols ripng
kireeti@pro1-a# set graceful-restart ?
Possible completions:
  <[Enter]>          Execute this command
  disable            Disable graceful restart
  restart-time       Time after which RIPng is declared out of
                     restart (1..600)
```

# Graceful Restart Summary

| | BGP | ISIS | OSPF |
|---|---|---|---|
| **Purpose** | Continue forwarding (PFE) during a restart of routing (RE) | | |
| **Changes** | Graceful Restart Capability<br>Per-peer configuration | New "re-start" option (TLV) in IIH PDU | New link-local (type 9) opaque-LSA called "Grace-LSA" |
| **Timers** | Restart timeout, stale routes timeout, path selection timeout | Restart timeout | Restart timeout (Grace Period) |
| **IETF Drafts** | *Graceful Restart Mechanism for BGP*<br>draft-ietf-idr-restart-05<br>draft-ietf-mpls-bgp-mpls-restart-00 | *Restart Signaling for ISIS*<br>draft-shand-isis-restart-01 | *Hitless OSPF Restart*<br>draft-ietf-ospf-hitless-restart-02 |

# Thank You!

http://www.juniper.net

kireeti@juniper.net