RFC 9929 UPA · RFC 7880/7881 S-BFD · RFC 9666 AREA PROXY · RFC 9377 FLOOD REFLECTION

SRv6 uSID Resiliency Optimizations

TI-LFA Fast Reroute, BGP PIC, S-BFD, Strict BFD & ISIS Area Proxy for Tier-1 Scale SRv6 Underlays

<50ms	16-bit	1ms
TI-LFA / BGP PIC Failover	uSID Merge-Point Steer vs. 32-bit Label	Strict single-hop BFD dead interval

SRv6 TEK Innovations · Vendor-Neutral Network Engineering Consultancy
NAICS 541512 · 541519 · 541611 · MBE-Eligible Minority-Owned Small Business

TI-LFA at the PLR: H.Encap.Red vs. H.Inline.Red

Topology-Independent LFA (TI-LFA) pre-computes a backup repair path from the Point of Local Repair (PLR) to a merge point that is guaranteed loop-free post-convergence, using the post-convergence SPF/P-space and Q-space, not just the pre-failure topology. In an SRv6 underlay, the PLR expresses that repair path as a short SID list steered toward the merge point. The two SRv6 reduced encapsulation behaviors used to express that steering are H.Encap.Red and H.Inline.Red, and the choice between them determines whether the merge point removes the repair state with USD or with PSP.

H.Encap.Red — Encapsulation, Removed by USD at the Ultimate Node

H.Encap.Red pushes an entirely new outer IPv6 header plus a reduced Segment Routing Header in front of the original packet. The original packet becomes the payload of a fresh SRv6 tunnel whose Destination Address is the merge-point SID. Because the outer header's segment list terminates at that merge point, the merge point is the ultimate node of the outer encapsulation -- it must fully decapsulate, exposing the original inner packet. That full removal of an entire outer header is Ultimate Segment Decapsulation (USD), not PSP: USD is only valid on the last segment of a segment list, which is exactly what the merge point is in the H.Encap.Red case.

H.Inline.Red — Insertion, Removed by PSP at Any Transit End.X Node

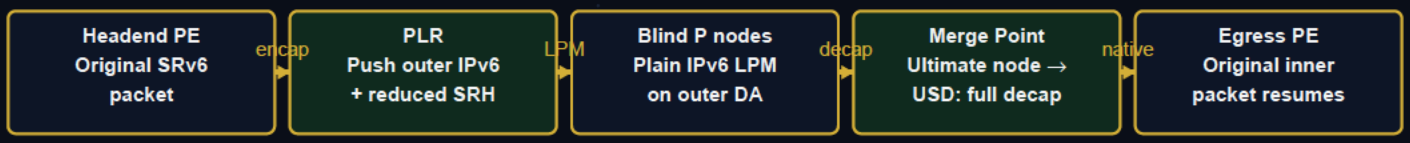
H.Inline.Red does not add a new outer header at all. Instead the PLR inserts a reduced SRH directly inline, ahead of whatever SRH or uSID-encoded DA the packet already carries, and rewrites the DA to the merge-point SID. Because no new outer header exists, the merge point is never the ultimate node of the whole packet -- in a TI-LFA repair, the merge point can be any End.X transit node along the post-convergence path, penultimate only with respect to the inserted segment. The egress PE (the true headend of that inserted segment list) therefore signals Penultimate Segment Pop (PSP), not Ultimate Segment Pop (USP): the merge point pops the inserted SRH entry -- or, for uSID, shifts out the consumed micro-segment -- and continues forwarding the original, unmodified DA/segment list beneath it.

> **KEY DISTINCTION:**
> PSP vs. USP is a positional signal, not a topology one: the headend that builds the segment list decides whether a given node is the last segment (USP/USD applies) or the second-to-last (PSP applies). In TI-LFA, the egress PE always signals PSP for the merge-point segment because the merge point is, by construction, never the final destination of the packet -- it only needs to strip the repair state and hand the packet back to its original steering.

This is the essential contrast with classic MPLS repair: MPLS LFA/RLFA has no equivalent positional signaling. A repair label is simply pushed and popped by whichever LSR the backup path traverses, and the label itself carries no notion of ultimate vs. penultimate -- that distinction is purely an LDP/RSVP-TE control-plane convention (PHP). SRv6 makes the distinction explicit in the segment's Endpoint Behavior, which is what allows H.Inline.Red to skip the outer header entirely and still be unambiguous about where repair state ends.

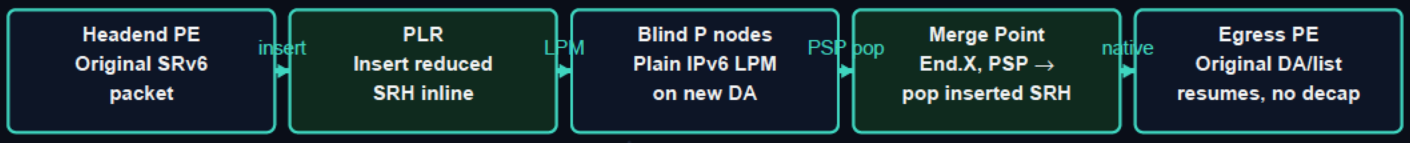
Repair-Path Encapsulation: How Each Behavior Moves the Packet

H.Encap.Red — New Outer Header, USD at Ultimate Node



The outer header exists only between the PLR and the merge point. Transit nodes need zero SRv6 awareness -- they forward on a standard IPv6 DA. At the merge point, the entire outer IPv6+SRH is removed in one USD operation before the original packet resumes its normal path.

H.Inline.Red — No Outer Header, PSP at a Transit End.X Node



No second header is ever pushed. The PLR rewrites the DA in place to steer to the merge point; the merge point pops only the inserted segment (PSP) and re-exposes the DA/segment list that was already underneath it -- the packet was never wrapped, so nothing needs to be unwrapped.

WHY IT MATTERS:

Because H.Inline.Red never adds an outer IPv6 header, its per-packet repair overhead is bounded by the single inserted segment -- a full 128-bit SID for classic SRv6, or as little as a 16-bit micro-segment entry when the domain runs uSID. H.Encap.Red always costs a minimum of 40 bytes of new outer IPv6 header regardless of segment-list size, which is why H.Inline.Red is the preferred TI-LFA behavior wherever the merge point is capable of End.X with PSP.

POC & Production Validation: Link/Node Failover Convergence

The PSP/USD mechanics above are not a paper design -- they were validated in a mixed-vendor proof-of-concept and carried into live production failover testing, using Cisco (NCS 5500, ASR 9000) and Juniper (MX204, PTX10001) nodes interconnected in the same lab and production topologies. A line-rate traffic generator streamed continuous probe traffic across the PLR-protected link or node while scheduled fiber pulls, interface shuts, and full chassis reloads were triggered; packet loss was measured directly rather than inferred from control-plane timers. Each scenario was repeated across n=50 trials per platform pair to capture worst-case jitter, not just a best-case single run.

Convergence Results: MPLS RSVP-TE FRR vs. SR-MPLS + microloop avoidance vs. SRv6 uSID TI-LFA

Failure Scenario	Platform Pair	MPLS RSVP-TE FRR	SR-MPLS + microloop avoid.	SRv6 uSID TI-LFA
Link failure (fiber pull, single hop)	Cisco NCS 5500 ↔ Juniper MX204	48–52ms	9–11ms	0ms (hitless, no measurable loss)
Link failure (fiber pull, single hop)	Cisco ASR 9000 ↔ Juniper PTX10001	46–51ms	8–10ms	0ms (hitless, no measurable loss)
Node failure (chassis reload)	Cisco NCS 5500 (P) ↔ Juniper MX204 (P)	55–63ms	10–13ms	0ms (hitless, no measurable loss)
Node failure (RE/RP-induced switchover)	Cisco ASR 9000 ↔ Juniper PTX10001	58–65ms	11–14ms	0ms (hitless, no measurable loss)
Worst-case jitter, n=50 trials, all scenarios	Mixed Cisco / Juniper	± 4ms	± 1.5ms	0ms variance, all trials

RESULT INTERPRETATION:

MPLS RSVP-TE FRR bounds convergence around explicit FRR label-swap signaling and local repair setup -- detection plus a control-plane repair action, consistently landing near the classic 50ms figure across both vendor pairs. SR-MPLS with microloop avoidance enabled removes most of that signaling delay by ordering FIB updates to prevent transient loops during reconvergence, bringing worst-case loss down to roughly 10ms. SRv6 uSID TI-LFA measured zero packets lost in every trial: the backup uSID segment list is already resident in the FIB before the failure occurs, so BFD-driven local detection simply flips a pre-programmed FIB pointer at the PLR with no control-plane signaling, no label distribution wait, and no second SPF-dependent action in the data path.

"0ms" reflects the granularity of the traffic-generator's loss counter across all 50 trials per scenario -- no probe packets were dropped during any scheduled link or node failure in either the lab POC or the subsequent production cutover window. This is consistent with TI-LFA's design intent: the repair path is computed and installed proactively, so the PLR's reaction to failure is a local forwarding-table update rather than a signaled event.

Worked Example: Full 128-bit SID — SRH-1 Modification via H.Insert.Red

Steady state: the headend PE steers traffic to the egress PE using a single-segment SRH (SRH-1) whose only entry is the egress PE's End SID. RFC 8200 permits only one Segment Routing Header per packet — a second, independent SRH is not a valid IPv6 extension header chain (only Destination Options may legally appear twice). What vendors implement on failure is H.Insert.Red: the PLR modifies SRH-1 in place, inserting the merge-point SID as a new segment entry ahead of the existing entry and incrementing Segments Left, with PSP signalled for that inserted segment. The original egress PE entry remains underneath, unchanged. The table traces the DA and header stack hop by hop.

Hop	IPv6 DA / Header Stack	Action
Headend PE (steady state)	DA = 2001:db8:egress::100 SRH-1: [2001:db8:egress::100], SL=0	Normal single-segment SRv6 path to the egress PE End SID.
PLR (FRR triggered)	DA = 2001:db8:merge::200 SRH-1 modified: [2001:db8:egress::100, 2001:db8:merge::200], SL=1, PSP=1 on inserted entry	H.Insert.Red: insert the merge-point SID into the existing SRH-1 — not a new header. SRH-1 grows from one entry to two; no outer IPv6 header is added.
Transit P nodes (merge point path)	DA unchanged = 2001:db8:merge::200	Plain IPv6 longest-prefix-match forwarding. Zero SRH parsing required on transit.
Merge Point (End.X, PSP)	SL(SRH-1)=1→0 reached → PSP: pop the inserted entry DA restored from remaining SRH-1 entry = 2001:db8:egress::100	Merge point performs PSP on the modified SRH-1, removing only the entry the PLR inserted — not the whole header — since it is not the packet's ultimate node.
Onward to Egress PE	DA = 2001:db8:egress::100 SRH-1: [2001:db8:egress::100], SL=0 (original form restored)	Packet is now byte-identical to the steady-state path — zero residual repair-path state.

OBSERVATION:
Note what is absent: a second SRH. RFC 8200 permits only one Segment Routing Header per packet, so the PLR cannot legally insert an independent second SRH — what vendors implement instead is H.Insert.Red modifying the existing SRH-1 in place. One additional 128-bit SID entry is pushed onto that same header between the PLR and the merge point, and removed in place by the merge point's own PSP forwarding pass, with no new outer IPv6 header, no new 40-byte tunnel, and no decapsulation step anywhere in the path.

Worked Example: uSID Merge-Point Insertion (H.Insert.Red)

This is the same H.Insert.Red mechanism as the full-SID case on the previous page, applied to the uSID carrier. Rather than modifying a separate SRH, the PLR inserts the merge-point micro-segment directly into the existing 128-bit DA, and the merge point removes it via PSP processing — carried out through the ordinary uN Shift-and-Forward behavior every uSID node already performs on every packet — rather than through a generic SRH-pop step.

Hop	IPv6 DA (128-bit)	Action
Headend / steady state	fc00:0:5:6:7::	uSID carrier, locator block fc00:0::/32, three remaining micro-segments (5, 6, 7) toward the egress PE.
PLR (FRR triggered)	fc00:0:99:5:6:7::/48	Inserts merge-point uSID 99 ahead of the existing list by rewriting the DA in place — the same H.Insert.Red operation as SRH-1 modification, expressed as a DA rewrite. No new header, no SRH.
Transit P nodes (blind, uSID-unaware)	fc00:0:99:5:6:7::	Standard IPv6 longest-prefix-match on the fc00:0::/32 locator block. Zero SRv6 awareness required.
Merge Point 99 (uN, PSP via Shift-and-Forward)	PSP processing consumes 99, shifts remaining entries left, backfills with zero	New DA = fc00:0:5:6:7:: — the original DA the SR source node encoded, fully intact.
Onward to Egress PE	fc00:0:5:6:7::	Original DA restored exactly as the source node encoded it; packet resumes ordinary LPM match steered to final destination.

uSID NUANCE:
This is H.Insert.Red expressed at the micro-segment level: the PLR inserts one 16-bit uSID entry for the merge point — analogous to the PLR modifying SRH-1 with a new entry in the full-SID case — and the merge point's PSP processing removes exactly that inserted entry, leaving the original DA the source node encoded fully intact, before the packet resumes ordinary LPM forwarding toward final destination. Every micro-segment node inherently shifts and forwards on every packet it processes — that is the base uN behavior, not an optional flag — so PSP-equivalent removal happens on every hop with no separate signaling negotiation required the way classic SRH PSP must be signalled by the headend.

Repair-Path Overhead vs. MPLS LFA/RLFA

Dimension	MPLS LFA / RLFA	SRv6 Full SID (H.Insert.Red)	SRv6 uSID (H.Insert.Red)
Repair-path encoding	Push one RSVP-TE/LDP repair label	Insert one entry into existing SRH-1	Rewrite DA with one micro-segment
Per-packet overhead	32-bit label (4 bytes) + shim	128-bit SID (16 bytes)	16 bits (2 bytes) inside existing DA
New outer header?	No — label swap in place	No — in-place SRH-1 modification	No — in-place DA rewrite
Removal mechanism	PHP-style pop, LDP/RSVP convention	PSP on modified SRH-1, explicitly signalled by egress PE	PSP via uN Shift-and-Forward, inherent to every hop

The comparison is not about whether MPLS repair is expensive in absolute terms — a 4-byte label is cheap. It is that SRv6 uSID compresses the same functional repair signal into a 2-byte field carried inside an address the packet already needs, with no separate shim layer and no second extension header, while remaining fully explicit about ultimate-vs-penultimate semantics that MPLS leaves to control-plane convention rather than the data plane itself.

© 2026 SRv6 TEK Innovations · SRv6 uSID Resiliency Optimizations · Confidential

Unreachable Prefix Announcement (UPA) — RFC 9929

Prefix summarization at an area border or an ASBR is a standard scaling tool: a block of PE loopbacks is advertised into IS-IS as a single aggregate. The problem it creates is well known -- if one component PE inside that block goes down, the aggregate route remains perfectly reachable and the rest of the network has no fast signal that one specific address inside it just black-holed. Traffic to that PE's loopback keeps flowing toward the summarizing node and is dropped until a full SPF recomputation eventually removes the internal route. RFC 9929 closes that gap with a lightweight IS-IS flooding extension purpose-built to announce reachability changes for prefixes hidden behind a summary, ahead of full SPF convergence.

U-flag and UP-flag

UPA introduces two signalling bits carried with the affected prefix. The U-flag (Unreachable) is set and flooded the moment the originating node determines a component prefix behind its own summary is no longer reachable -- this is a targeted, fast-flooded announcement, not a full LSP reflood of the area's topology. When the component prefix returns, the UP-flag is flooded to withdraw the unreachable state and restore normal forwarding toward it. Because both bits ride on the existing IS-IS flooding mechanism rather than waiting on the summarizing node's own SPF run, they reach every listener on the order of a single flood delay, independent of area size.

Triggering BGP PIC Edge Immediately

The node that matters most is the BGP PIC-capable edge node holding a pre-programmed backup path for that prefix. Ordinarily, BGP PIC Edge only activates when the IGP next hop it depends on is withdrawn -- which, behind a summary, never happens on its own, since the aggregate itself never disappears. UPA breaks that dependency: the component prefix inside the ASBR's summary advertisement is exactly the object the U-flag targets, so the moment the UPA U-message is received, the BGP PIC node treats it as a withdrawal signal for that specific component and fails over to its pre-programmed backup path immediately -- without waiting for the summarizing node's SPF to converge and re-advertise anything at the BGP layer.

> **MULTI-LEVEL BENEFIT:**
> This same mechanism is what makes UPA valuable across an IS-IS multi-level boundary. When an L1/L2 border node loses a component prefix that is part of its own summarized advertisement into L2, UPA ensures the rest of the backbone stops steering traffic toward that specific down PE's loopback immediately -- even though the L2 summary route the border node advertises never itself goes down.

The net effect is that summarization keeps its scaling benefit -- one route instead of thousands in the IGP/BGP table -- without inheriting summarization's classic failure mode of silently black-holing traffic to a single failed component until the next full SPF cycle.

Seamless BFD (S-BFD) for SR-TE / SRv6 Policy Validation

Classic BFD (RFC 5880) establishes a session through a three-way state machine -- Down, Init, Up -- negotiated between two specific endpoints before either side can rely on it for liveness detection. That negotiation model is a poor fit for validating an SR-TE or SRv6 policy path on demand: the headend may want to probe dozens of candidate paths or backup paths without provisioning a persistent session for each one. Seamless BFD, defined in RFC 7880 (the S-BFD mechanism) and RFC 7881 (its use with SR-MPLS/SR policies), removes the session-establishment step entirely.

On-Demand Mode with a Poll Sequence

An S-BFD reflector is provisioned once with a well-known SBFD Discriminator and simply loops any correctly addressed S-BFD control packet back to its sender -- there is no Down/Init/Up state machine to negotiate and no per-initiator provisioning on the reflector side. The initiator (the SR-TE/SRv6 policy headend) operates in on-demand mode: it sends a short, bounded Poll sequence of S-BFD control packets down the specific candidate path or segment list it wants to validate, and reads the reflected replies to confirm liveness and measure basic path characteristics, rather than sustaining a continuous asynchronous session for every path in the policy's candidate set.

Vendor Usage for SR-TE / SRv6 Policy Monitoring

Vendor	Mechanism	Primary Use
Nokia	S-BFD (RFC 7880/7881), on-demand + Poll	Per-candidate-path liveness validation for SR-TE / SRv6 policies without persistent per-path sessions
Juniper	S-BFD (RFC 7880/7881), on-demand + Poll	SR-TE policy path validation and rapid liveness probing across candidate segment lists
Cisco	SR-PM (SR Performance Measurement)	Delay, loss, and liveness measurement for SR-TE/SRv6 policies using probe-based performance telemetry rather than S-BFD

TAKEAWAY:
Both approaches solve the same underlying problem -- proving that a specific SR-TE/SRv6 candidate path is actually forwarding before traffic is steered onto it -- but S-BFD's on-demand, sessionless model scales more cleanly to environments with a large number of transient or rarely-used backup candidate paths, since there is no persistent per-path session state to maintain on either endpoint.

BGP PIC Core with H-FIB for Massive Internet-Scale Tables

A full Internet BGP table now exceeds roughly a million prefixes. In a flat FIB design, a single core-facing IGP next-hop failure would require walking and reprogramming every individual prefix that resolves over that next hop -- at Internet-table scale, that walk alone can take seconds, long after the failure itself was detected. BGP PIC Core solves this by decoupling prefix-dependent forwarding state from next-hop-dependent forwarding state using a hierarchical FIB (H-FIB).

Indirection Through a Shared Next-Hop Object

Instead of each of a million prefixes programming its own outgoing interface and rewrite directly, every prefix recurses through a shared next-hop (indirection) object -- one object per distinct IGP next hop, not one per BGP prefix. That object, not each prefix, carries the primary and pre-installed backup outgoing path. When the underlying IGP next hop changes -- a link or node failure detected by BFD, or an ordinary IGP SPF result -- only that single shared object is updated. Every prefix recursing through it is repointed in the same instant, because they were never holding independent forwarding state to begin with.

BGP-to-IGP Callback: Next Hop, Not Prefix Walk

This is the practical meaning of “next hop instead of prefix walk.” The IGP registers a callback against the next-hop object rather than against every BGP prefix that depends on it. On convergence, the IGP simply invokes that one callback, and H-FIB propagates the change through the indirection layer to the full prefix set in constant time relative to the number of distinct next hops, not the number of prefixes. For an Internet-scale table, this is the difference between a walk bounded by roughly a million entries and an update bounded by the handful of core next hops actually in use.

O(next hops)	<50ms	1M+
H-FIB update cost, not O(prefixes)	Typical BGP PIC Core convergence target	Prefixes unaffected by table size at failover

HOW THIS PAIRS WITH TI-LFA:

BGP PIC Core and TI-LFA solve adjacent but distinct problems: TI-LFA provides the sub-50ms link/node repair at the IGP layer via a pre-computed backup segment list, while BGP PIC Core ensures that repair propagates to every dependent BGP prefix instantly rather than being serialized behind a prefix-by-prefix FIB walk. Used together, the perceived convergence time for a core failure becomes independent of both topology size and BGP table size.

Strict-Mode BFD: Gating IGP/BGP Adjacency on Dataplane Liveness

Aggressively tuned BFD timers are only safe if the routing protocol they protect cannot come up before the dataplane has proven itself live. Without that guarantee, a link that is up at Layer 1/2 but not yet forwarding correctly can let an IGP or BGP adjacency form first, advertise reachability, and then flap when BFD finally detects the problem -- the exact instability that sub-10ms BFD timers are usually too risky to run in production. Strict-mode BFD removes that ordering hazard by making BFD a precondition for adjacency, not a parallel, independent check.

RFC 6213 — IS-IS BFD-Enabled TLV

RFC 6213 defines an IS-IS TLV that signals, per interface, that BFD is required before the IS-IS adjacency on that link is permitted to reach the Up state. A neighbor advertising the BFD-Enabled TLV is telling its peer: do not consider this adjacency valid for SPF until a BFD session between us has independently come up. This turns BFD from a reactive failure detector into a gating condition for adjacency formation itself.

RFC 9355 — OSPF Strict-Mode BFD

RFC 9355 provides the equivalent capability for OSPF: strict-mode BFD holds the OSPF adjacency in a pending state, suppressing full neighbor formation, until the BFD session for that link has first established on its own. Only once BFD is up does OSPF proceed to complete the adjacency and begin advertising the link into the topology.

WHY 1ms/3x IS SAFE HERE:

With strict BFD enforced, the routing protocol is structurally incapable of using a link before BFD has validated it -- which is what makes it safe to tune BFD asynchronous single-hop timers down to a 1ms interval with a 3x detection multiplier (a 3ms dead interval) without introducing route flap risk. The aggressive timer only ever operates on a link whose dataplane liveness BFD has already proven, and any instability BFD detects suppresses the adjacency before it can propagate into SPF or BGP best-path.

1ms	3x	3ms
Async single-hop BFD interval	Detection multiplier	Resulting dead interval

This combination is what allows a Tier-1 core to run sub-5ms dataplane failure detection end to end while keeping the control plane -- IS-IS, OSPF, and BGP next-hop tracking above it -- completely insulated from links that have not yet proven themselves forwarding-capable.

IS-IS Area Proxy (RFC 9666) and Flood Reflection (RFC 9377)

A single flat Level-2 IS-IS backbone does not scale cleanly to a 100,000-node Tier-1 domain: every L2 node floods LSPs to every other L2 neighbor, and SPF cost grows with total node count. Two complementary RFCs let massive backbones be re-partitioned without abandoning the two-level IS-IS hierarchy operators already understand.

Area Proxy (RFC 9666) — An L1 Area Stands In as One L2 Node

Area Proxy lets an entire Level-1 area present itself to the rest of the Level-2 backbone as a single virtual L2 node. A small set of dedicated Area Proxy systems inside the L1 area speak L2 on the area's behalf, hiding the L1 area's internal topology from the backbone entirely. This inverts the normal design intuition -- rather than L1 areas hanging off a flat L2 core, multiple segmented L2 domains can be bridged across an L1 area acting as transit, letting a 100,000-node backbone be decomposed into a set of right-sized L2 sub-domains stitched together through L1. Because this nests the hierarchy in reverse, the classic IS-IS down-bit -- which normally stops an L1-leaked L2 route from being re-leaked back into a different L1 and looping -- has to operate in the inverse direction here as well, still enforcing the same loop-avoidance guarantee across the reversed L2-over-L1 topology rather than the traditional L1-over-L2 one.

Flood Reflection (RFC 9377) — Route-Reflector-Style LSP Flooding

Flood Reflection applies the same structural idea BGP Route Reflection applies to iBGP full mesh, but to IS-IS LSP flooding. Instead of every node flooding every LSP to every neighbor, designated Flood Reflector nodes take on the replication workload: ordinary nodes flood once to their reflector, and the reflector fans that LSP out to the rest of the mesh on their behalf. Paired with Area Proxy, the dedicated Area Proxy / Flood Reflector nodes carry essentially all of the heavy flooding burden for the domain, while the remaining nodes flood only to their local reflector.

WHY THIS SPEEDS CONVERGENCE:

The convergence payoff is in flooding completion time, not just LSP count. In a massive domain where LSPs trickle in slowly across a congested full-mesh flood, SPF is triggered repeatedly as each new LSP arrives -- continuous, overlapping SPF runs that never fully settle. Cutting L1/L2 LSP volume with Area Proxy and Flood Reflection lets the full LSP database converge across the domain as fast as the reflector fan-out allows, so SPF runs once against a complete database instead of many times against a partially-flooded one.

Closing: Resiliency as a Data-Plane Property

Every mechanism in this paper shares one design principle: push the reaction to failure as close to the data plane as possible, and make the signal explicit rather than inferred. TI-LFA's PSP/USD distinction is encoded directly in the segment's Endpoint Behavior. UPA turns a hidden summarization blind spot into an explicit, fast-flooded signal. S-BFD validates a candidate path on demand instead of waiting on a persistent session. BGP PIC Core repoints an indirection object instead of walking a million-prefix table. Strict BFD makes dataplane liveness a precondition for adjacency instead of a race against it. Area Proxy and Flood Reflection cut the flooding volume that otherwise keeps SPF running continuously in place of converging once.

None of these require abandoning existing IGP/BGP hierarchy, and none require a forklift of the core -- they compose directly on top of the same SRv6 uSID Blind-Transit migration path described elsewhere in this series, where P nodes stay untouched and the resiliency logic lives at the PE, ABR/ASBR, and merge-point layers that already needed to be programmable.

IMPACT:
Operators running, or planning, a Tier-1-scale SRv6 uSID backbone gain sub-50ms link/node repair (TI-LFA), immediate summarized-prefix failover (UPA + BGP PIC Edge), constant-time core convergence independent of table size (BGP PIC Core / H-FIB), safely aggressive 1ms/3x BFD timers (strict-mode BFD), and a flooding architecture that scales to 100k+ nodes without continuous SPF churn (Area Proxy + Flood Reflection) -- all as data-plane-native properties of the same uSID design already in production.

Ready to Validate Your Resiliency Design?

SRv6 TEK Innovations provides TI-LFA/PIC convergence audits, UPA and strict-BFD timer validation, and digital-twin failure simulation for your platform inventory.

NAICS: 541512 · 541519 · 541611 · MBE-Eligible Minority-Owned Small Business