



Telemetry-Tuned Weighted SRv6

Fine-grained application-aware congestion mitigation.

Shashwitha Puttaswamy
George Washington University

Mariam Kiran
Oak Ridge National Laboratory

Problem Statement

Different applications share limited capacity,but need different path properties.

Shortest-path and equal-cost routing do not, by themselves, account for every application’s performance requirements

The challenge: Adapt path selection to application needs and changing congestion across shared network links.

Why existing mechanisms leave a gap

Transport congestion control
TCP and ECN regulate the sending rate, but do not select the SRv6 path for each class.

Per-hop traffic treatment
DiffServ controls queue behavior. Path selection remains a separate decision.

Explicit paths need adaptive allocation
SRv6 supplies forwarding paths. The controller must decide how each class uses them.

Paper aims to solve

Connect application requirements to routing

Use trusted classes to select the appropriate SR Policy for each destination.

Respond to changing network conditions

Adjust segment-list weights using current telemetry and class-specific objectives.

Respect shared capacity

Coordinate allocations across classes and control how policy changes reach the headend.

Proposed method: TTW-SRv6

Function	Mechanism	Output
Classification	Trusted ingress and domain DSCP	Application-class identity
Path enforcement	SR Policy and selected SID list	Explicit SRv6 forwarding path
Adaptation	Telemetry and class-specific scoring	Updated segment-list weights
Safety	Shared-capacity and stability checks	Controlled policy installation

Weights guide per-flow path selection at the headend.

Step	Action	Purpose
1	Classify and mark traffic	Establish a trusted application class
2	Bind the class to an SR Policy	Select validated forwarding options
3	Observe candidate paths	Measure congestion and availability
4	Score paths for each class	Reflect application requirements
5	Check and install weights	Respect capacity and limit churn

Step 1: trusted traffic classification

Class	Flow behavior	Domain DSCP
C0	Low latency	EF / 46
C1	Priority	CS5 / 40
C2	High throughput	AF31 / 26
C3	Long-living flows	AF21 / 18
C4	Short-living flows	AF11 / 10
C5	Intent-based flows	Custom / 12
C6	Default flows	BE / 0

Step 2: class-specific SR Policy selection

Match the trusted class and destination

Select the configured SR Policy at the ingress headend.

Use validated segment lists

Each SID list represents an eligible forwarding option toward the endpoint.

Apply relative path weights

The headend selects a list and uses its SRH instructions to forward the traffic.

Step 3: observe candidate paths

Measurement	Meaning for path selection
Delay and jitter	Transit time and timing variation
Packet loss	Delivery reliability
Link utilization	Pressure on the most loaded link
Residual bandwidth	Capacity available for additional traffic
Availability	Whether the path can still carry traffic

Step 4: evaluate paths by application need

Class	Main routing preference
C0: Low latency	Low delay and jitter
C1: Priority	Reliable delivery
C2: High throughput	Available bandwidth
C3: Long-living	Stable path allocation
C4: Short-living	Shorter estimated completion time
C5: Intent-based	Explicit application constraints
C6: Default	Balanced network conditions

Step 5: check and install policy weights

Check shared-link capacity

Evaluate the combined allocation across classes before installing new weights.

Stabilize the update

Use smoothing, dwell time, hysteresis and rate limiting to reduce repeated changes.

Keep a feasible fallback

Quantize supported weights, remove failed paths and retain rollback capability.

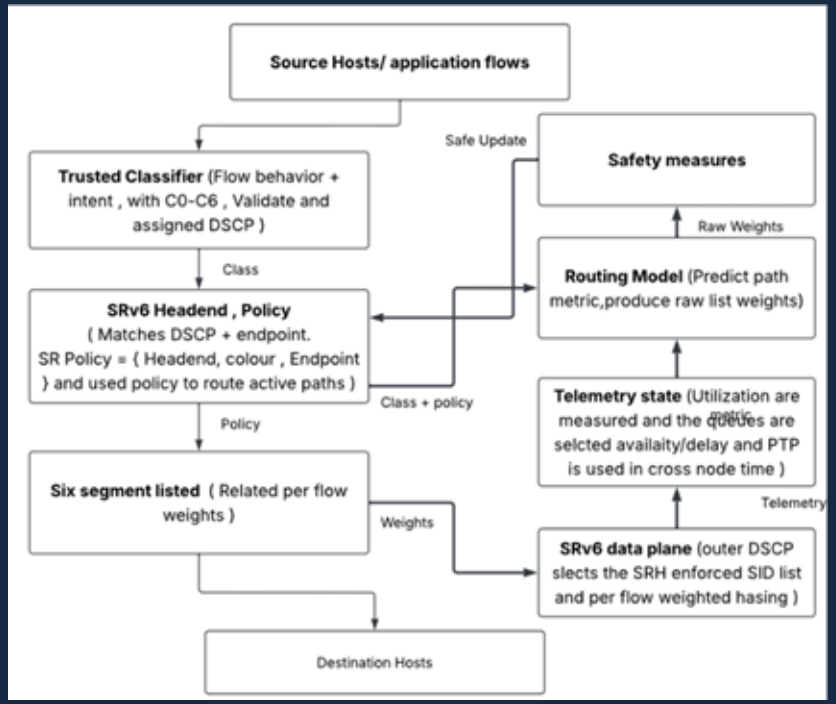
Telemetry and policy control loop

Forwarding

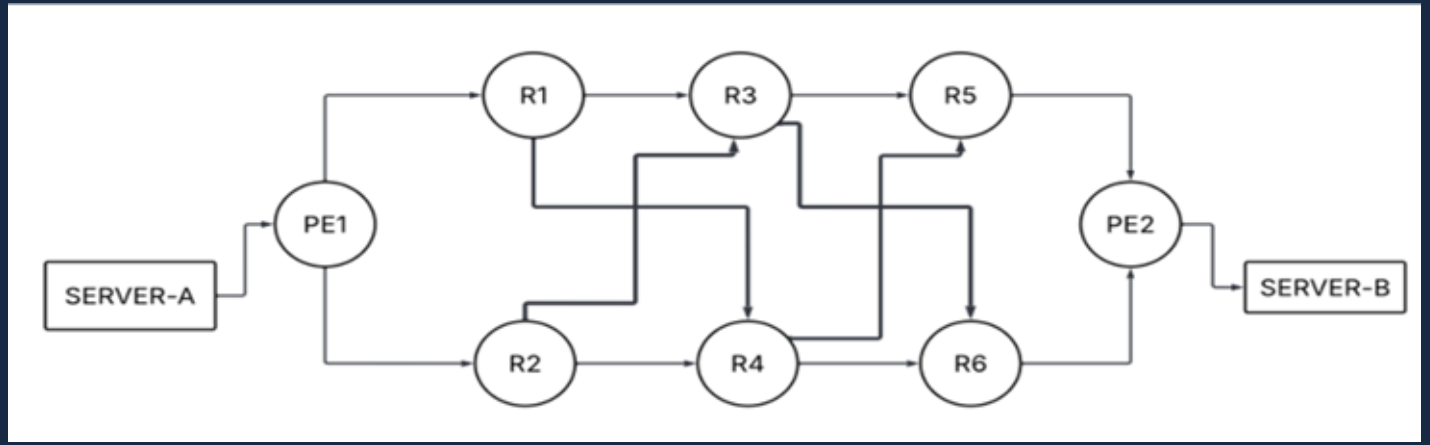
The trusted class selects a policy. The chosen SID list determines the path.

Feedback

Measurements inform the next controlled weight update.



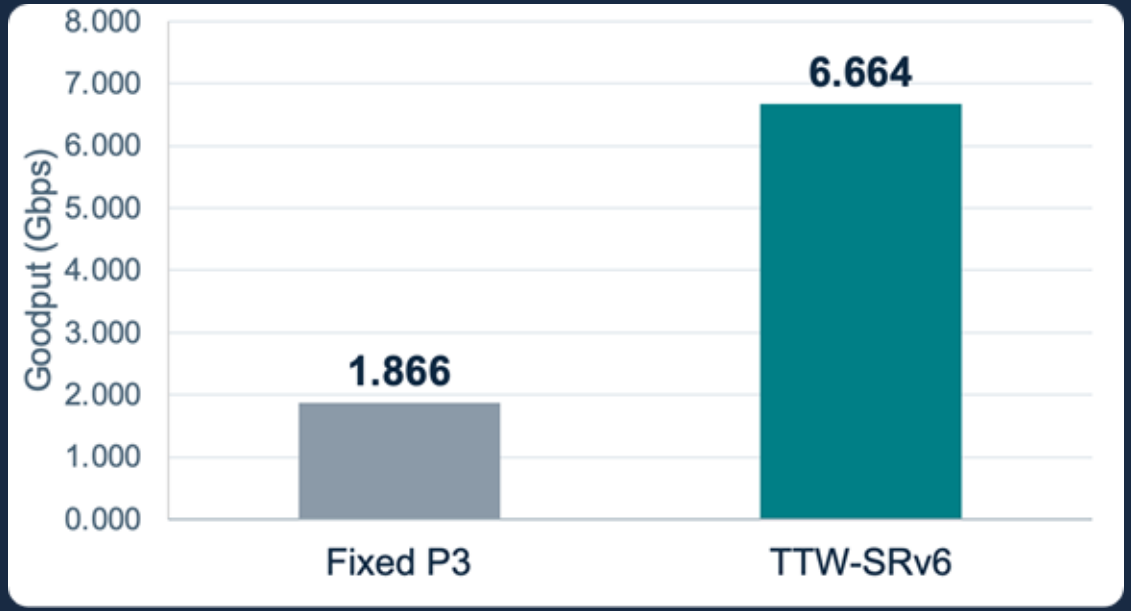
Evaluation setup



Seven classes and six validated paths

TTW-SRv6: adaptive weights across P1–P6

C4 delivers 3.57× the baseline goodput



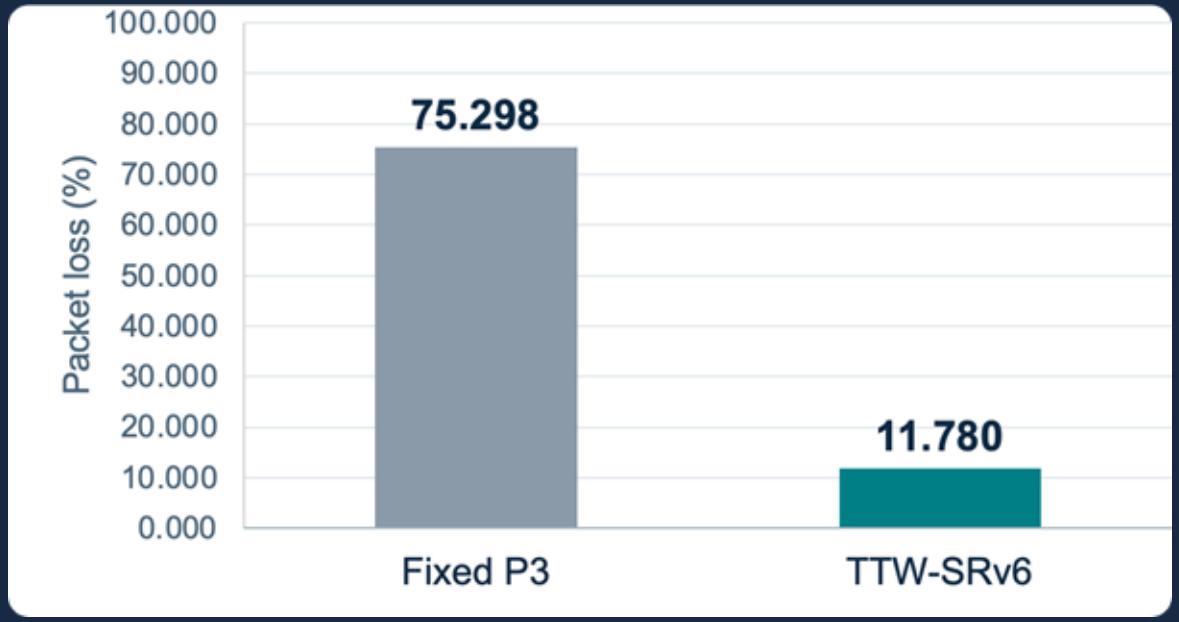
257.1% higher

4.798 Gbps more useful data delivered

Higher is better

Adaptive allocation improves delivery in the reported C4 scenario.

Packet loss falls sharply, but remains high



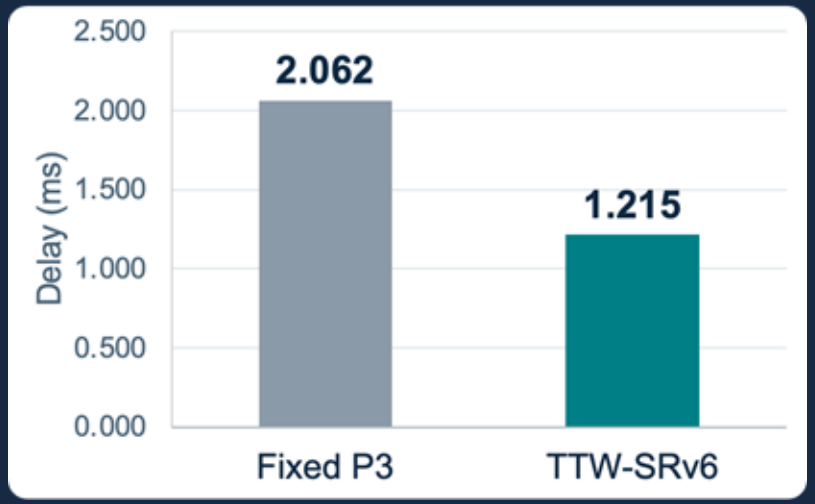
84.4% lower

63.518 percentage points lower

11.780% remains

Congestion is reduced. The remaining loss is still substantial.

C4 sees lower delay and less timing variation





Mean delay: 41.1% lower

Mean jitter: 45.0% lower

Reported results: C4 short-living traffic

Metric	Fixed P3	TTW-SRv6	Reported change
Mean delay (ms)	2.062	1.215	41.1% lower
Mean jitter (ms)	0.160	0.088	45.0% lower
Packet loss (%)	75.298	11.780	84.4% lower
Goodput (Gbps)	1.866	6.664	257.1% higher

Higher goodput means more useful data delivered per second.

Congestion response

Observe the bottleneck
The R1–R3 link shows rising load and queue pressure while alternatives retain headroom.

Shift path preference
Decrease weights on affected paths and favor suitable alternatives through R4–R6.

Check the effect across classes
Evaluate shared-link load before installation, then continue monitoring the network.

Results

Better delivery in the reported scenario

C4 gains goodput while delay, jitter and packet loss decrease.

Congestion remains

Packet loss is still 11.780%, so adaptive allocation does not eliminate overload.

The baseline limits the conclusion

The numerical evidence compares C4 with fixed P3, rather than optimized ECMP or SR-TE.

Implementation considerations

Forwarding-platform capabilities

Segment-list limits, weight granularity, hashing behavior, and update latency affect deployment.

Controller resilience

Telemetry validation, retained policy versions, and rollback support safe operation.

Further evaluation

Test broader baselines and workloads, class-level outcomes, and behavior on real hardware.


Thank
You