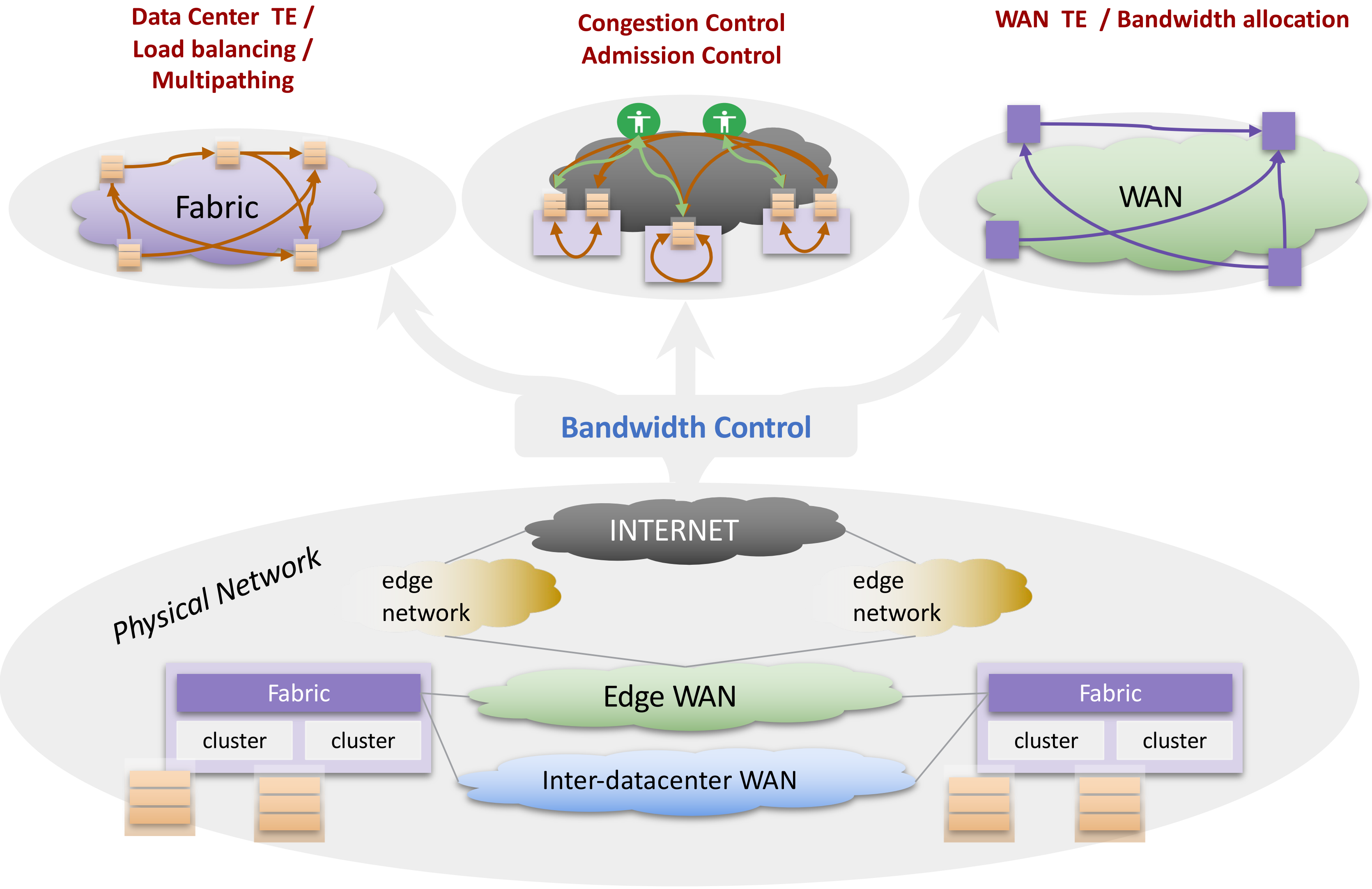
Congestion Signaling (CSIG)

Simple and Effective In-band Network Signals for Efficient Traffic Management in Datacenter Networks

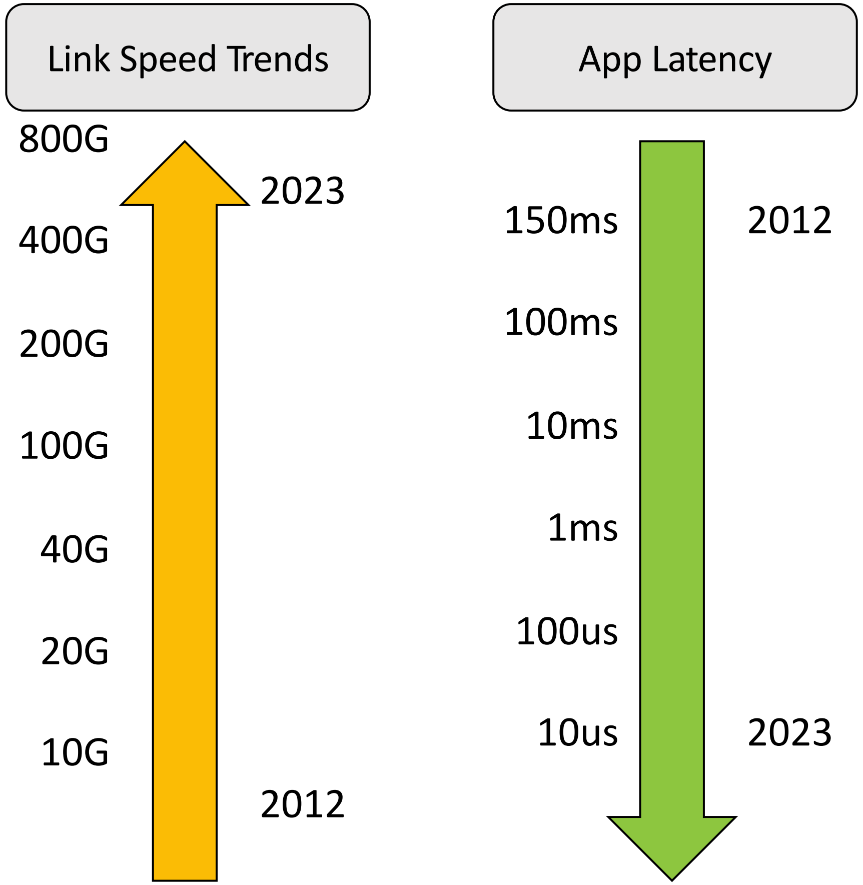
IEEE 802 Nendica Meeting
November 17, 2023

Ecosystem of Traffic Control Loops



Collection of control loops work in conjunction to achieve network efficiency and performance

Requirements for Network Efficiency



To sustain network efficiency, early and accurate visibility of **bottlenecks** is important for

- Robust congestion control
  - Requires real-time network telemetry signals from the fabric to address blind spots in end-to-end algorithms
- Traffic management
  - Requires visibility into transient congestion due to (milliseconds) bursts to guide traffic engineering, multipathing and load balancing techniques and accommodate bursts

In-band signaling over application data packets is required to enable fast response and flow attribution

Networking for ML Workloads

- Trend: More accelerators, more bandwidth per accelerator
  - Horizontal scaling → exercises shared data center network fabric
- Network efficiency is critical for AI/ML
  - Bursty traffic, large # of connections, flow collisions
  - Large-scale synchronized bursts stress the network unlike traditional compute / storage applications
  - Training → throughput *and* latency sensitive,
  - Inference / serving → latency sensitive
- Network performance directly impacts Training step time
  - Barrier collectives in training sensitive to 100p network latency
  - Throughput sensitive where compute/communication overlap.

AI/HPC Workloads Requirements

□ 2 or 3 stage CLOS or Dragonfly+ topologies

□ High bandwidth, high radix, ultra low latency switches

- AI Inference: Needs Latency SLA aimed at distributed models
- AI Training: Needs High bandwidth, low latency for small models

□ 256K+ network endpoints

□ Low tail latency and FCT

□ High performant transport in tandem with end-to-end congestion control

□ Network signals for optimal bandwidth utilization, flow ramp up and scheduling

Existing Technologies

□ HPCC++[1] use network INT metadata

□ SOLAR[2] use summary INT metadata

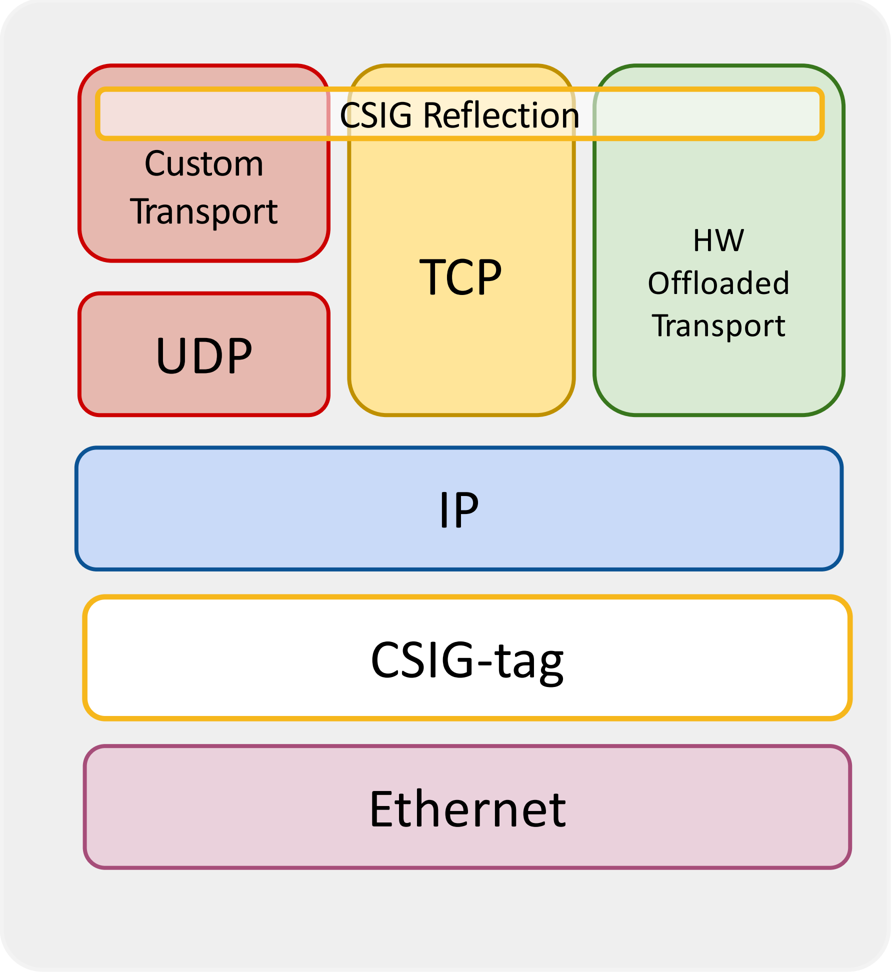
□ SWIFT[3] use RTT for delay-based congestion control

1 and 2 has overhead in terms of

- *Impacts PMTU*
- *Increased switch latency, power budget and hw complexity*
- *Doesn’t work very with tunneled domains. Extremely difficult to propagate INT metadata stack.*
- *Not very well suited for brownfield deployment. IPv6 Hop-by-Hop options are processed in slow path.*
- *Considerable overhead for small sized packets.*

[2] *https://dl.acm.org/doi/abs/10.1145/3544216.3544238*
[1] *https://dl.acm.org/doi/10.1145/3341302.3342085*

CSIG: Practical & Effective In-band Signaling protocol



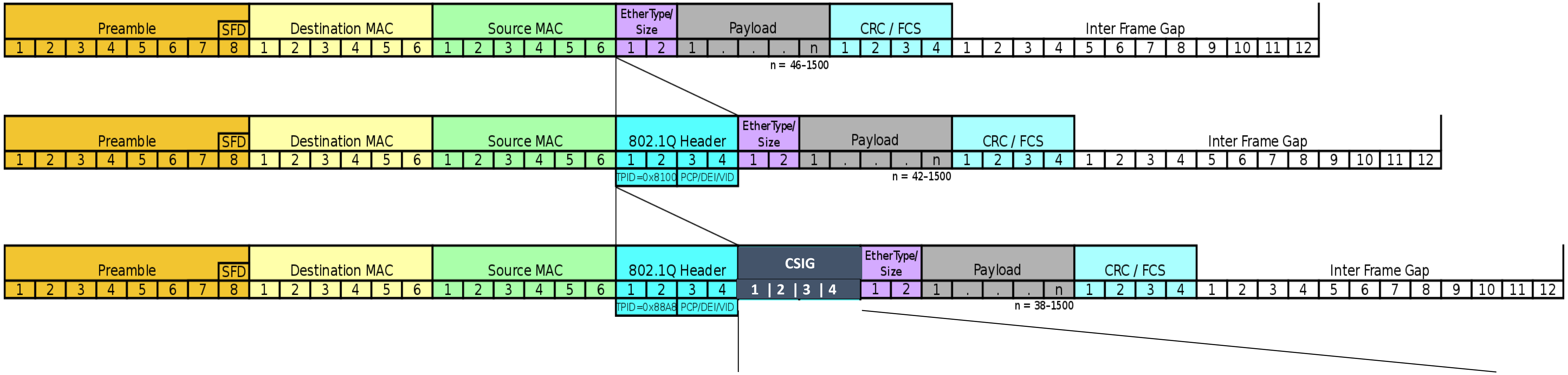
- Provides fixed-length simple summaries from the path bottlenecks
- Designed for Congestion Control, Traffic Management and Network debuggability use-cases
- Works with encrypted, tunnels, and tunnel in tunnel deployments
- Support modern AI, HPC oriented transports
- Designed for brownfield deployment with backward compatibility / interoperability
- Link to IETF Draft - https://datatracker.ietf.org/doc/draft-ravi-ippm-csig/

In-band Network Signals for Detecting Path Bottlenecks

Minimum Available Bandwidth: min(ABW)	The minimum available bandwidth in bps across all links on the packet path
Maximum Link Utilization: max(U/C) or min(ABW/C)	The maximum link utilization in percentage of link speed across all links on the packet path
Maximum Per-Hop Delay: max(PD)	The maximum per-hop delay across all hops in the packet path

…and potentially many more

CSIG Tag 4B Version



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         CSIG-ethtype            |  T  |R| Signal  |  Locator    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Byte 1-4

16 bits *- CSIG-ethtype: IEEE allocated code point for 4Byte CSIG frame*

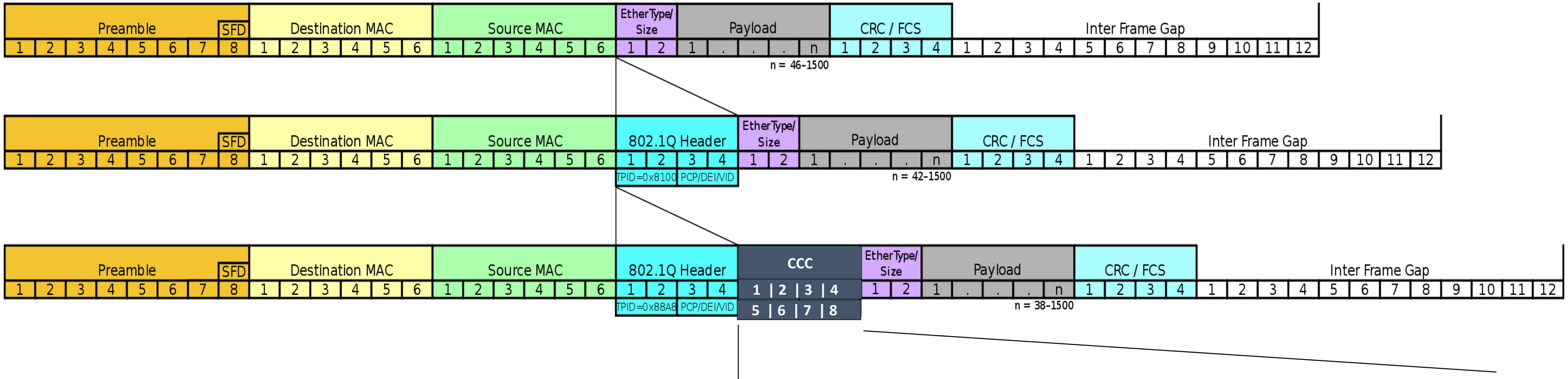
3 bits *- T: Signal Type (0:minABW, 1: minABW/C, 2:maxDelay)*

1 bit *- reserved*

5 bit *- S: Signal: Bucketed Signal Value (fully configurable)*

7 bits *- LM: Locator Metadata of bottleneck link (fully configurable per port information)[Port down notification back to sender]*

CSIG Tag 8B Version



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         CCC-ethtype           |            Locator            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   T   |              Signal Value             |    Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Byte 1-4

16 bits *- CSIG-ethtype: IEEE allocated code point for 8Byte CSIG frame*

16 bits *- LM: Locator Metadata of bottleneck link (fully configurable per port information)*

Byte 5-8

4 bits *- T: Signal Type (0:minABW, 1: minABW/C, 2:maxDelay)*

20 bits *- S: Signal: Quantized Signal Value*

8 bits *- Reserved*

8B: Fixed step function BW quantization

4B: Fully configurable 32 BW range buckets

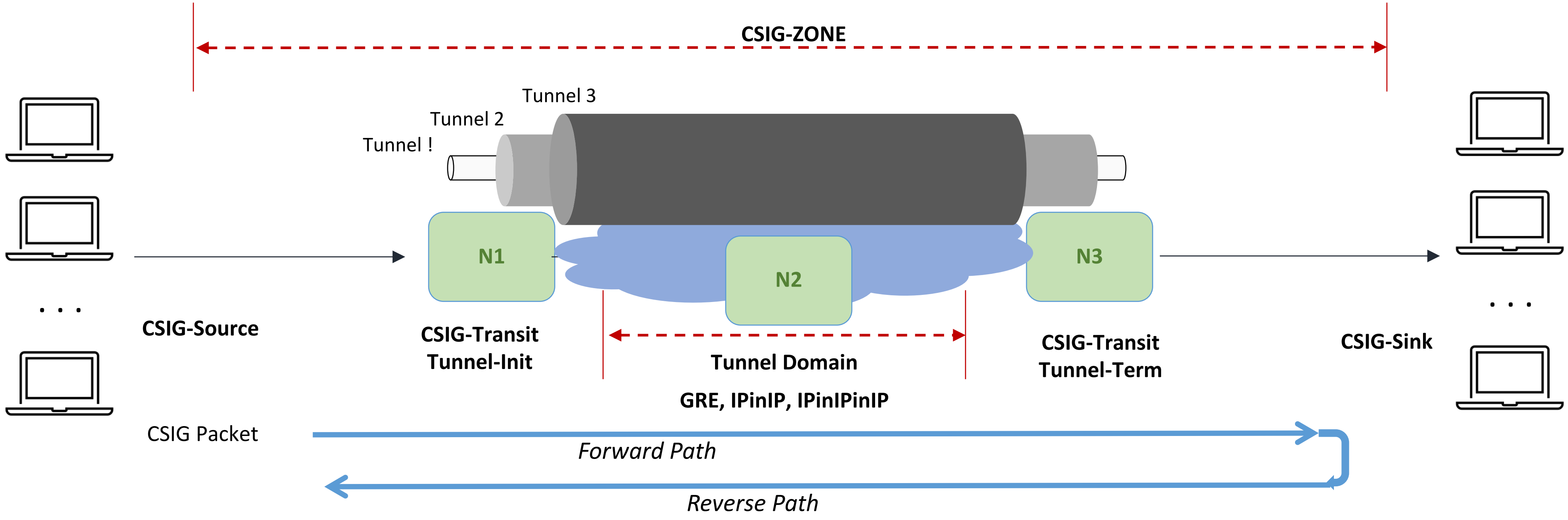
Ether Type Layering

ARPA : SA/DA/CCC-tag/ethtype/payload
802.1q : SA/DA/ vlan-tag/ CCC-tag/ethtype/payload
802.1ad : SA/DA/ vlan-tag/ vlan-tag/ CCC-tag/payload
802.1ae : SA/DA/Sec-tag/vlan-tag/ CCC-tag /Payload

Bridge Domain will forward the traffic and “MAY” update the CSIG Tag if bridging device is capable

Routed Domain will treat CSIG tag either as opaque tag for forwarding or will update the tag.

CSIG Life of a Packet

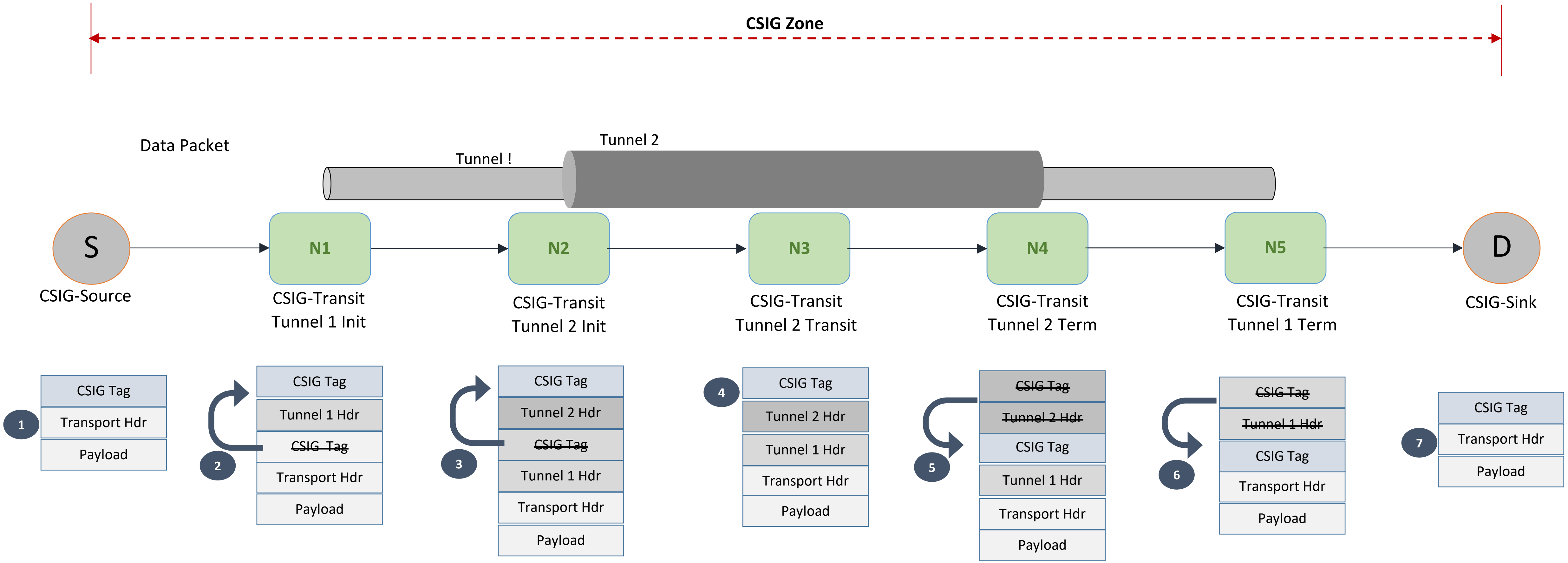


- *CSIG-Source: NIC sending CSIG packet (insert CSIG Header)*
- *CSIG-Transit: Network Device updating the CSIG fields in CSIG packet*
- *CSIG-Sink: NIC consuming the CSIG packet; will reflect ACK packet with CSIG data from incoming packet*

NIC end station is always the CSIG source and sink

Network device MAY be a CSIG sink when terminating a CSIG zone

CSIG Tag Propagation



1 *NIC Inserts CSIG Tag*

2 *N1 updates and copy CSIG Tag to tunnel 1*
N1 deletes incoming CSIG Tag from transport header

3 *N2 updates and copy CSIG Tag to tunnel 2*
N1 deletes incoming CSIG Tag from transport header

4 *N3 updates CSIG Tag;*

5 *N4 performs tunnel 2 decap and copies updated CSIG Tag in tunnel 1 header*

6 *N5 performs tunnel 1 decap and copies updated CSIG Tag to non tunneled packet transport header*

7 *NIC consumes CSIG Header*

CSIG Propagation in Tunneled Domain

CSIG packet may undergo multiple recursive tunnel domains. This means that the CSIG tag needs to be propagated up or down (encap or decap) when entering or exiting the tunnel domain. This is true for both L2 and L3 tunnels.

CSIG Tag Propagation:
Propagation can be configured at tunnel encap (UP) or decap (DOWN). Similar to MPLS TTL propagation.

Tunnel Encap:

- Update and Copy incoming CSIG tag to Tunnel L2 header.
- Delete incoming CSIG tag from payload.

Tunnel Decap:

- Update and Copy incoming CSIG tag to Payload L2 header
- Delete incoming CSIG tag along with tunnel header decap

Tunnel Turnaround (Tunnel Encap and Decap on the same node): Example MPLS VPN traceroute packets

- Collapsed CSIG tag update
- Update and Preserve incoming tag for Tunnel Turnaround.

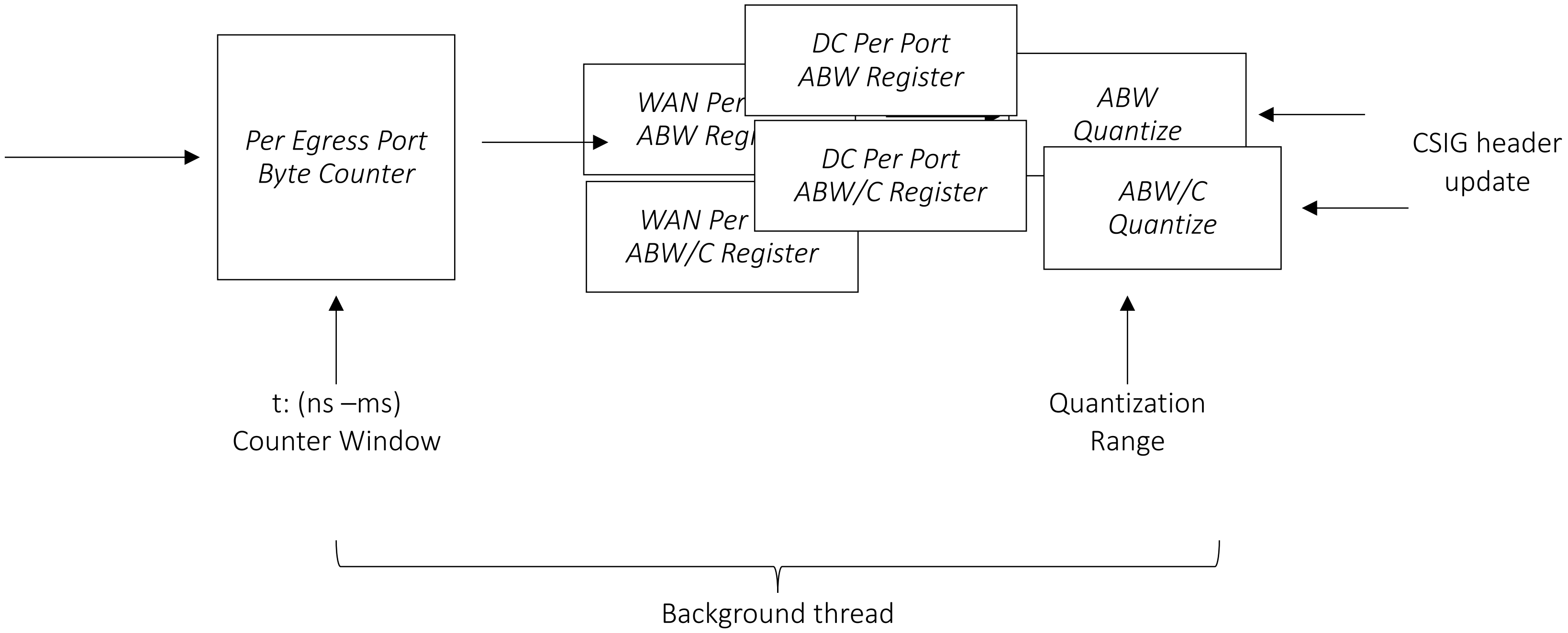
Switch Design Principles

HW friendly CSIG specification helps design high performance switch architecture

1. No impact on power budget
2. Line rate processing of signals
3. No impact on switch latency
   - *Fixed size CSIG header doesn’t need variable length parsing and editing*

4. No buffer overhead
   - Fixed size CSIG header (compared to INT variable sized metadata)

Pipeline Quantization Process



Per Egress Port Byte Counter

WAN Per ABW Regi

WAN Per ABW/C Register

DC Per Port ABW Register

DC Per Port ABW/C Register

ABW Quantize

ABW/C Quantize

CSIG header update

t: (ns –ms) Counter Window

Quantization Range

Background thread

BW Measure
Quantized Available Link BW Algorithm – ABW

$$m = Traffic\ in\ bits\ measure\ over\ a\ timer\ interval\ t$$
$$p = port\ speed (bps)$$
$$rate\ r = \frac{m}{t} (bps)$$
$$Availabe\ BW = (p\ - r)$$
$$Quantized\ Available\ BW: bbbbb\ (5bit\ value): Q(p - r)$$

$$pkt \rightarrow minBW = \min(Quantized\ Available\ BW, pkt \rightarrow minBW)$$

5 bits give 32 quantized values

All the network devices in a CSIG domain must be configured with the same value of t and quantization range

BW Measure
Quantized Available Link Load Algorithm – ABW/C

Compute % load observed for a given packet in a flow

$$m = Traffic\ in\ bits\ measure\ over\ a\ timer\ interval\ t$$
$$p = port\ speed$$
$$rate\ r = m/t$$
$$Consumed\ Load = \left({}^{r}/_{p}\right) * 100$$
$$Availabe\ Load = 100\left(1\ - \left({}^{r}/_{p}\right)\right)$$
$$Quantized\ Available\ Load{:}\ bbbbb\ (5bit\ value){:}\ Q(Available\ Load)$$

$$pkt \rightarrow minLoad = \min(Quantized\ Available\ Load, pkt \rightarrow minLoad)$$

5 bits give 32 quantized values

All the network devices in a CSIG domain must be configured with the same value of t and quantization range

Timers

We suggest CSIG Profile MUST support for BW measure computation

1. *Two timers, one for DC RTT and one for WAN RTT*
2. *Timer Granularity*
    - *1us, 10us, 100us, 1ms, 10ms, 100ms*
    - *Good to have 100ns mainly for future proofing*

Industry Support

Deployment

✓ Widely Deployed in Google Data Centers

Broadcom Support

✓ Tomahawk Series of Switches

✓ Jericho Series of Switches

Thank You