A Day in the Life of an L2/L3 TSN Data Packet.

Norman Finn
Version 2

Mar. 5, 2014

This presentation

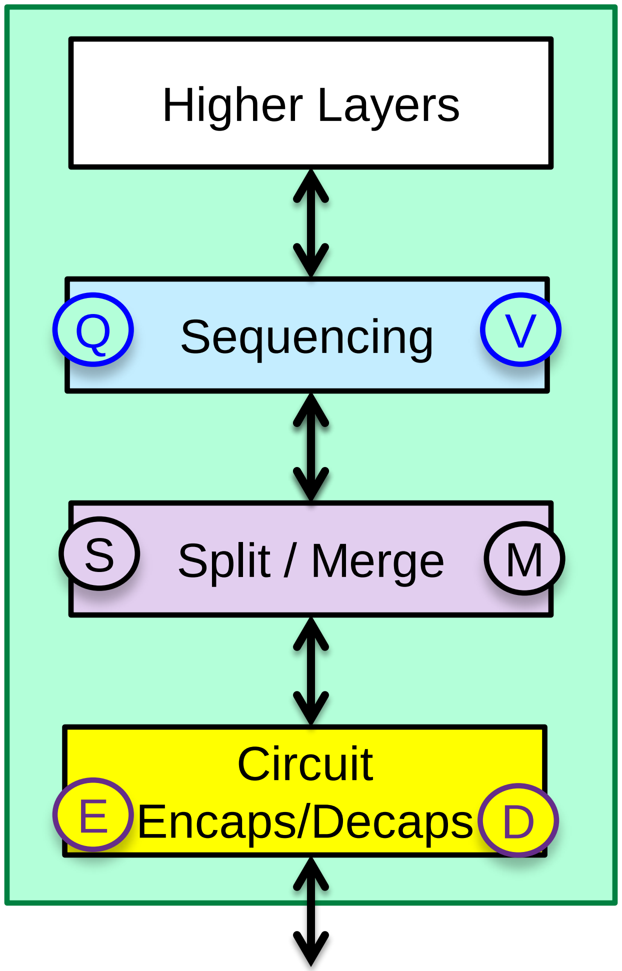
- This presentation, tsn-nfinn-Day-In-The-Life-0214-v02 is an annex to a two-part presentation.
- Part 1, tsn-nfinn-L2-Data-Plane-0214-v04, introduces concepts on which these presentations depend.
- Part 2, tsn-nfinn-L3-Data-Plane-0214-v03, is concerned with Layer 3 issues.
- See also cb-nfinn-How-Many-VLANs-0214-v02.

Outline

1. A very brief introduction, using concepts introduced in the preceding decks, followed by 6 “A Day In The Life Of A Packet” case studies.
2. Case 1: End-to-end Sequenced TSN encaps.
3. Case 2: Mixed L2/L3 IPgram pseudowire encaps.
4. Case 3: IPgram pseudowire to Sequenced TSN Stitching.
5. Case 4: IEC 62439-3 HSR or PRP encaps.
6. Case 5: End-to-end Ethernet-over-XYZ tunnels.
7. Case 6: IP Multicast encaps.
8. A one-slide summary of conclusions is given.

Layering

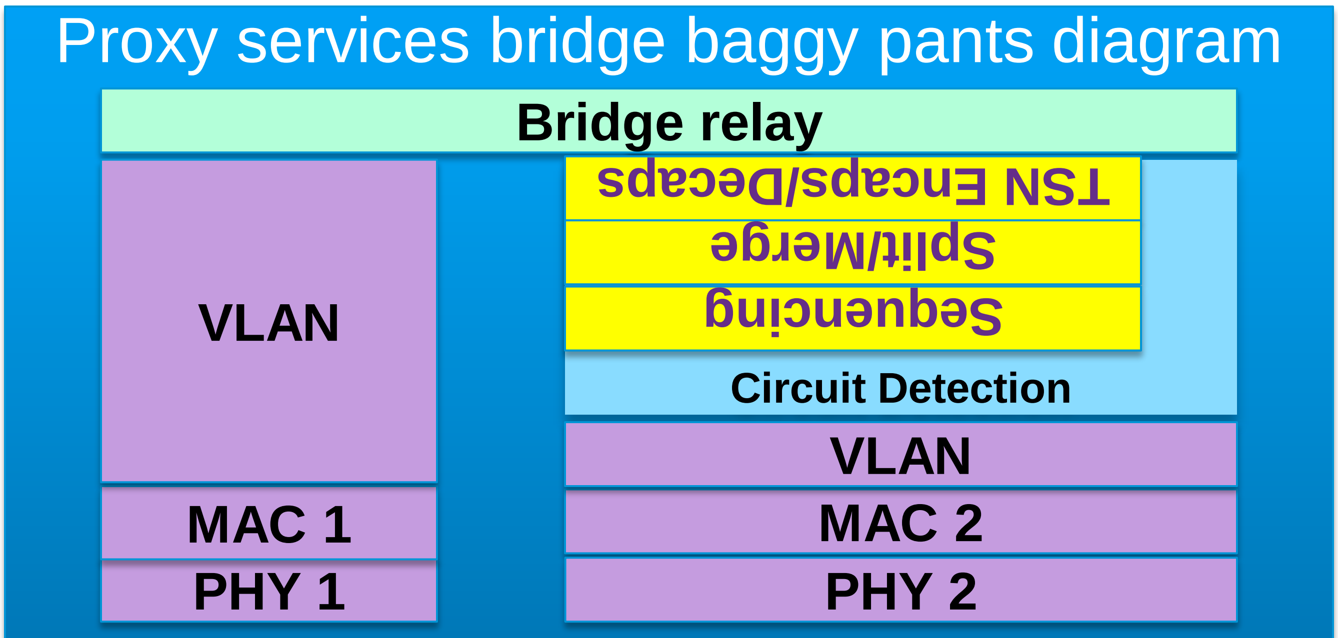
Layer reminder (from L2-Data-Plane)



- Higher Layers work as always.
- "Sequencing" numbers packets Q, and discards duplicates V.
- Split S /Merge M has one circuit ID above and two below its layer.
- Circuit Encaps E /Decaps D marks individual circuits.

Proxy bridge stack (from L2-Data-Plane)

- This is the stack for a bridge that proxies for a non-TSN client, e.g. Bridge 8 in the following examples:

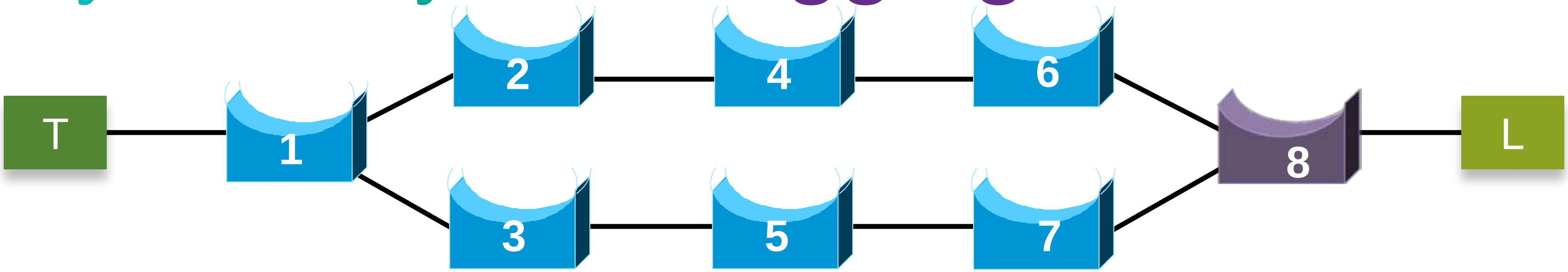


Case 1:
Layer 2 end-to-end
Sequenced TSN tagging

Sequenced TSN tagging

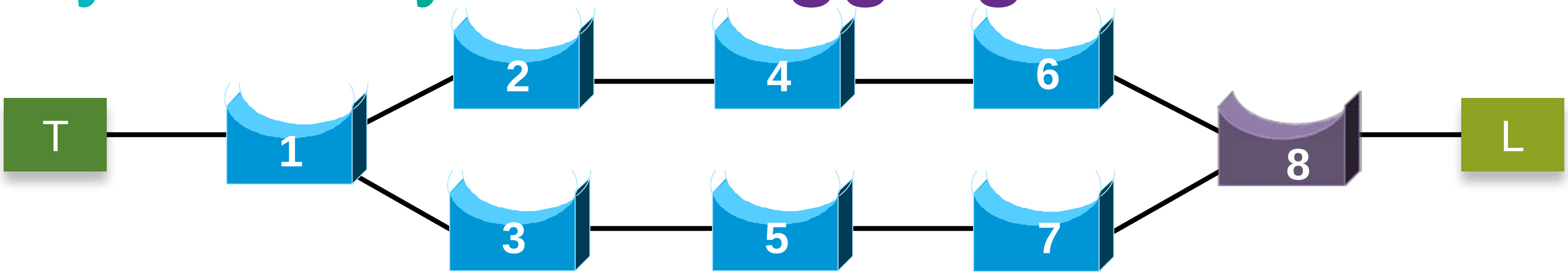
- Top-down view

Layer 2 only: **TSN tagging**



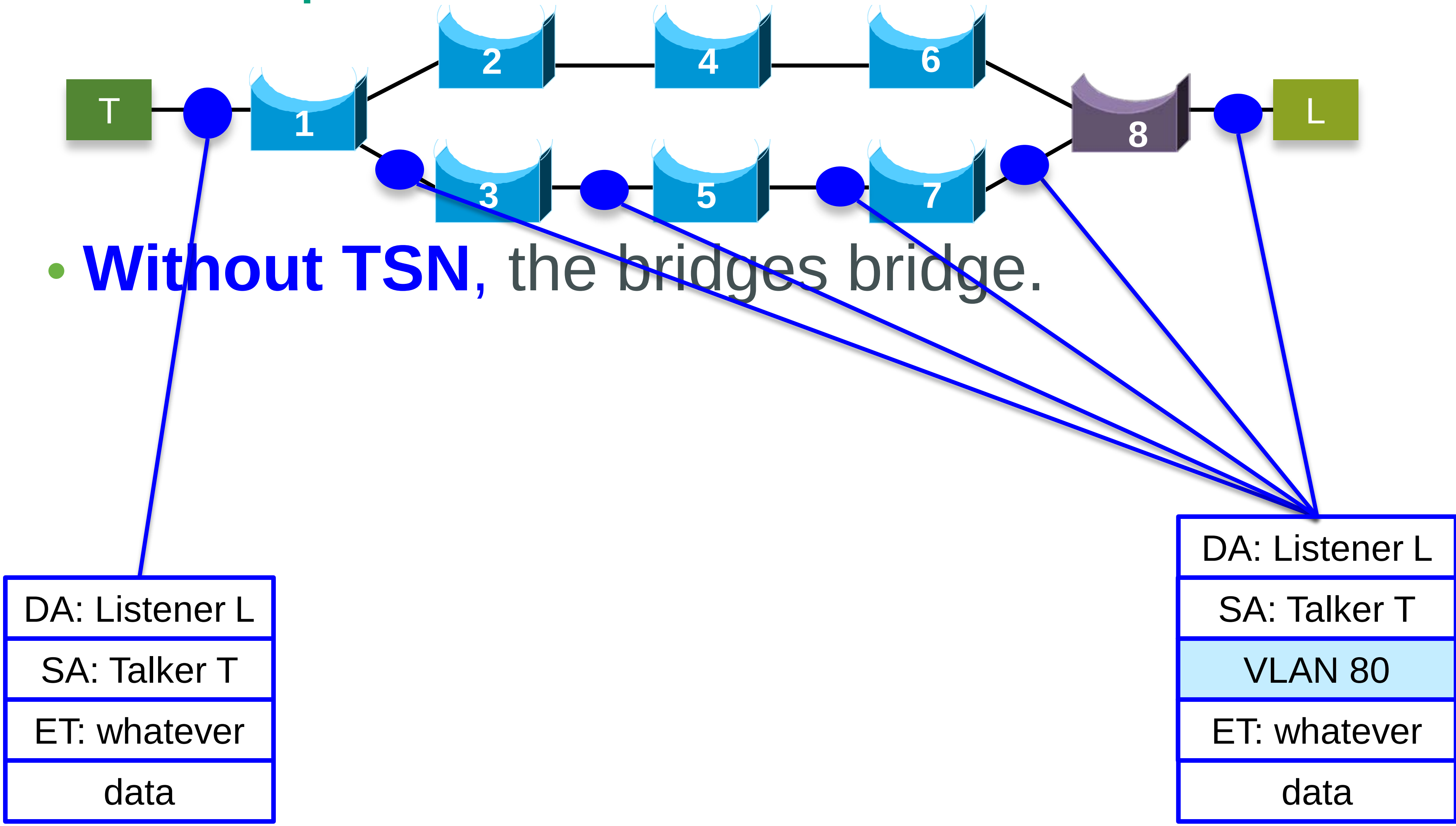
- Given that introduction, let us examine the simplest case: end-to-end connectivity through a Bridged LAN.

Layer 2 only: **TSN tagging**



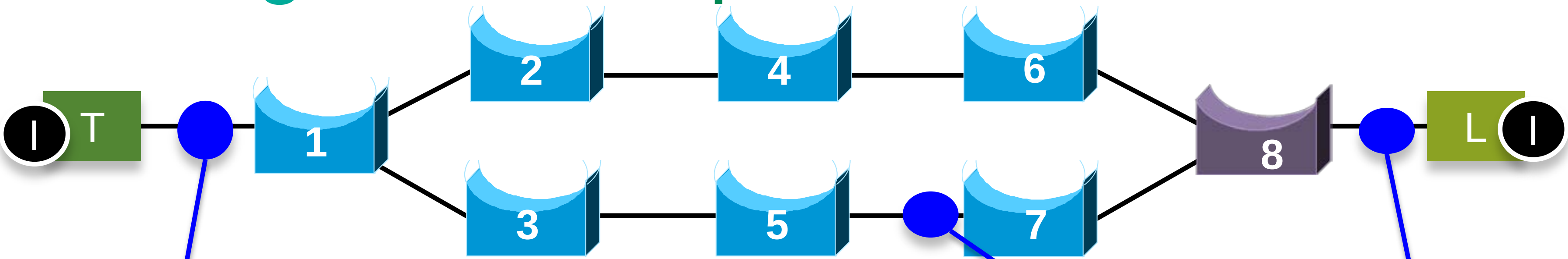
- Talker is TSN-aware, Listener is not.
- Talker is **not** VLAN-aware, Listener **is** VLAN-aware.

Natural packets



- **Without TSN**, the bridges bridge.

Peering relationships

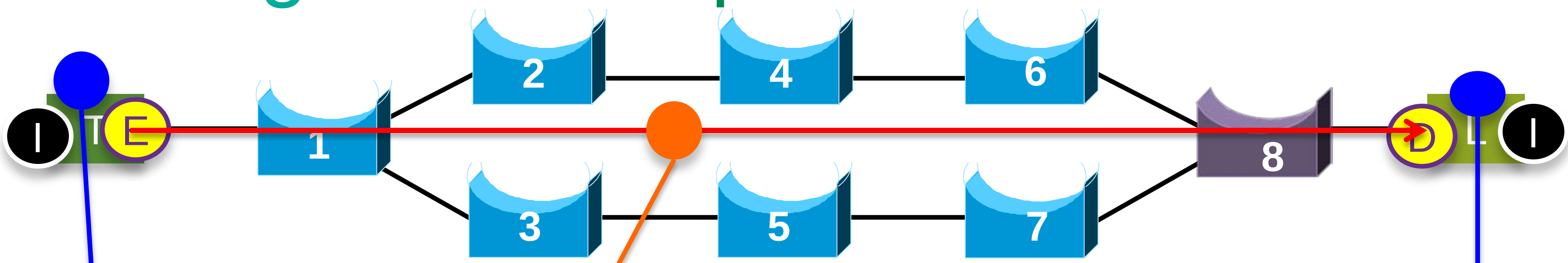


- Talker T and Listener L have a higher layer relationship. I

DA: Listener L
SA: Talker T
ET: whatever
data

DA: Listener L
SA: Talker T
VLAN 80
ET: whatever
data

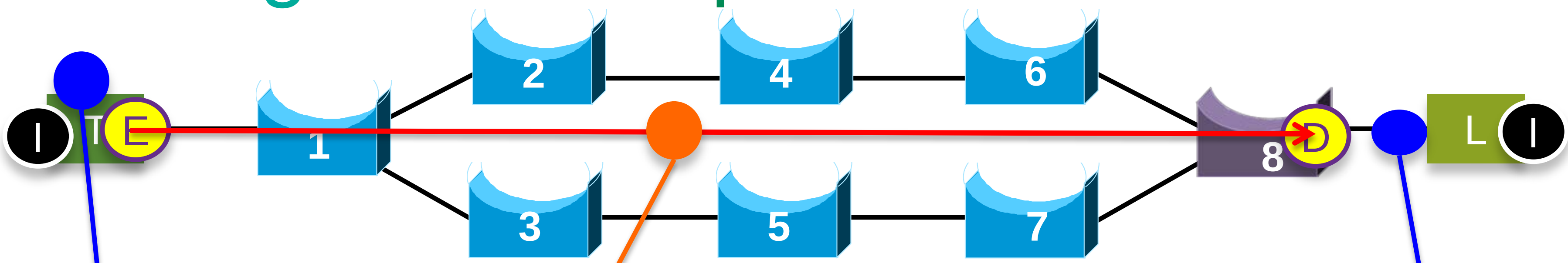
Peering relationships



- The operator wants Talker T and Listener L to have a TSN **circuit** relationship (E), (D), (734[99]) so that they can get the TSN QoS. (The bridges need the circuit ID in order to provide the TSN QoS.)

DA: Listener L	DA: TSN 734	DA: Listener L
SA: Talker T	SA: T	SA: Talker T
circuit_ID	VLAN tag 99	VLAN 80
ET: whatever	ET: whatever	ET: whatever
data	data	data

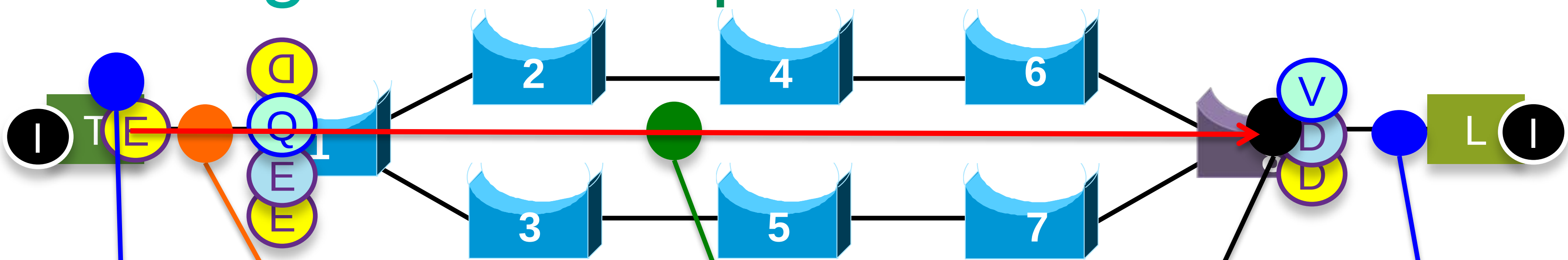
Peering relationships



- But, the Listener is TSN-unaware, so Bridge 8 has to provide the TSN Circuit Decaps (D) as a **proxy service**.

DA: Listener L	DA: TSN 734	DA: Listener L
SA: Talker T	SA: T	SA: Talker T
circuit_ID	VLAN tag 99	VLAN 80
ET: whatever	ET: whatever	ET: whatever
data	data	data

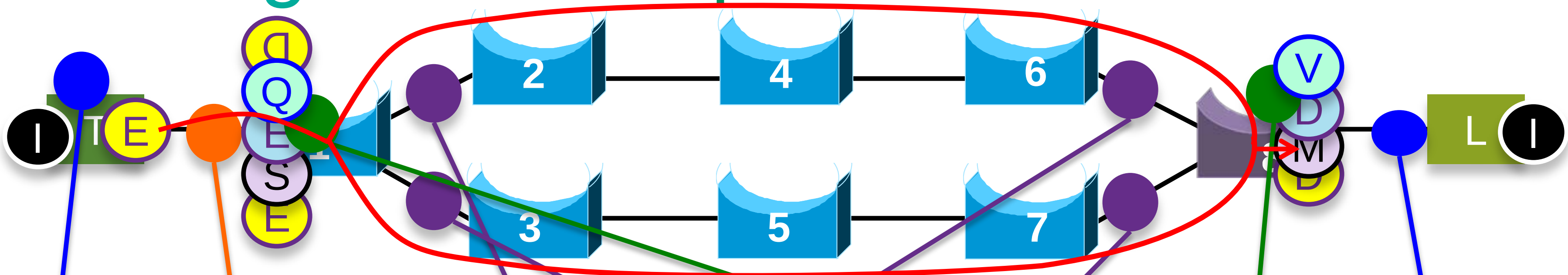
Peering relationships



- The operator wants **Sequencing** Q V proxied for T and L by Router 1 and Bridge 8, and seq. ecaps E D.

		DA: TSN 734	DA: Listener L	
		SA: T	SA: Talker T	
DA: Listener L	DA: TSN 734	VLAN tag 99	VLAN 80	DA: Listener L
SA: Talker T	SA: T	ET: TSN Seq	circuit_ID	SA: Talker T
circuit_ID	VLAN tag 99	Sequence #	seq_number	VLAN 80
ET: whatever	ET: whatever	ET: whatever	ET: whatever	ET: whatever
data	data	data	data	data

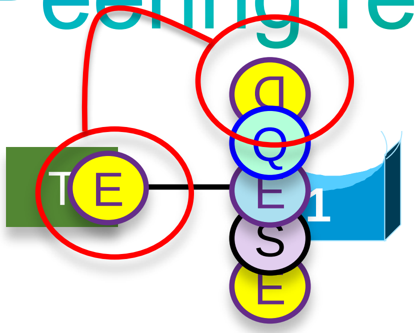
Peering relationships



- We want Split (S) and Merge functions (M), for seamless redundancy where the **circuit** bifurcates.

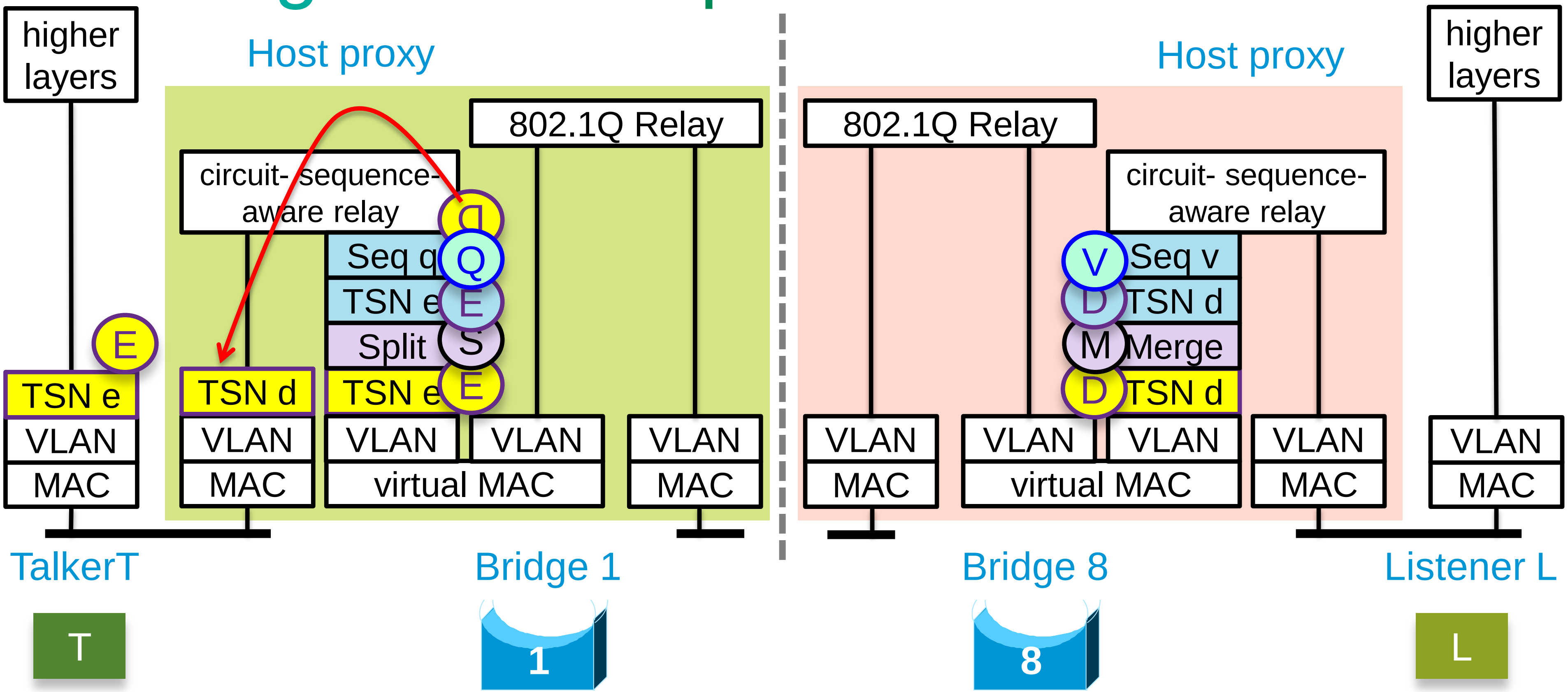
		DA: TSN **7840**	DA: TSN **12**	DA: L	
		SA: T	SA: T	SA: T	
DA: Lis	DA: TSN 734	VLAN tag **23**	VLAN tag **50**	vlan_ID 80	DA: L
SA: Tal	SA: T	ET: TSN Seq	ET: TSN Seq	circuit_ID	SA: T
circuit_	VLAN tag 99	Sequence #	Sequence #	Sequence #	VLAN 80
ET: wha	ET: whatever	ET: whatever	ET: whatever	ET: whatever	ET: whatever
data	data	data	data	data	data

Peering relationships



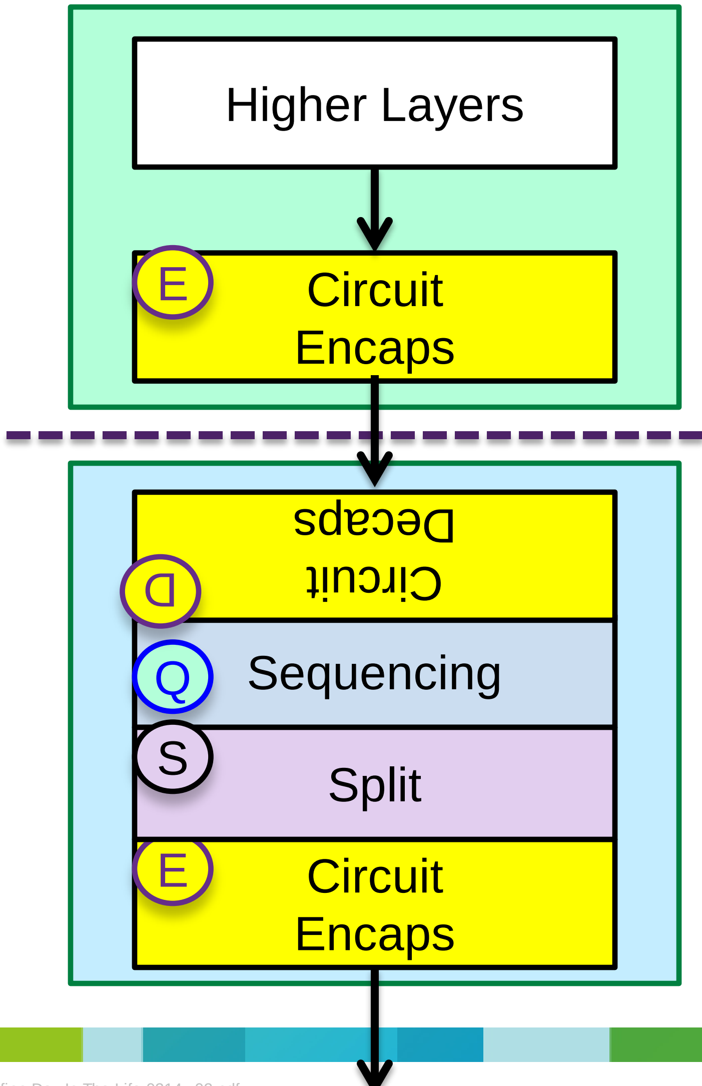
- Why is this TSN Decaps function upside down?
- Because it peers with Talker T's TSN Encaps function.

Peering relationships



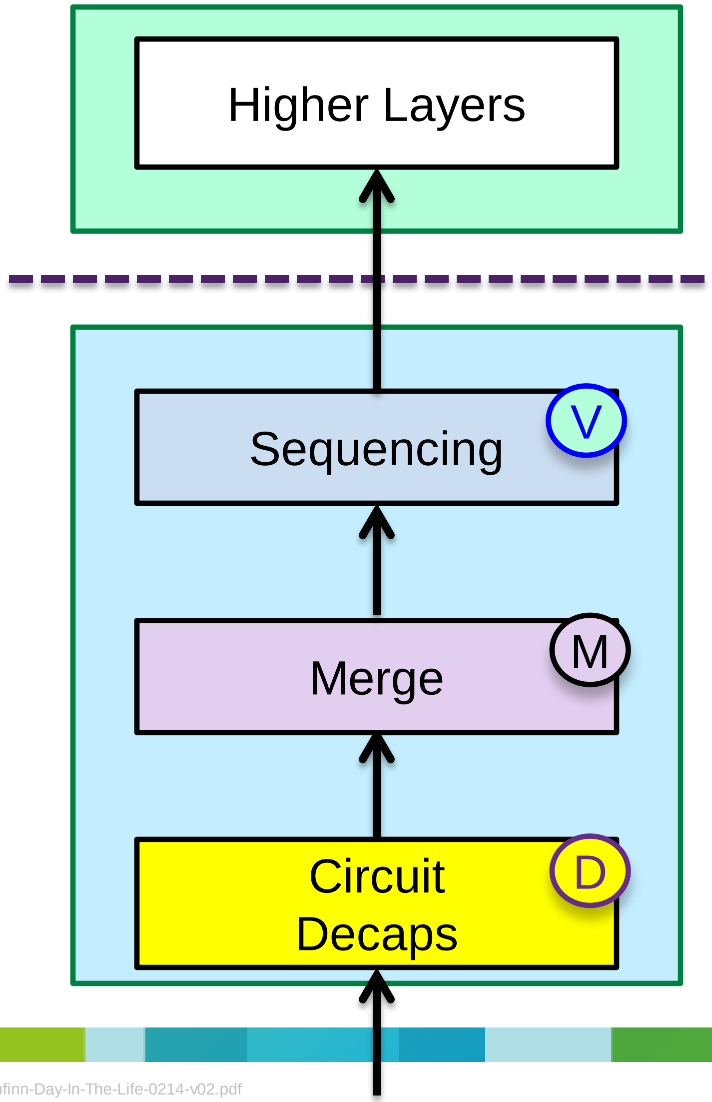
- This is a more accurate picture, but you can only get so much information on one slide.

Peering relationships: Talker side



- In this example, the Circuit Encaps is in the **Talker** system (above the link).
- And the Sequencing is in Bridge 1 (below the link).

Peering relationships: Listener side

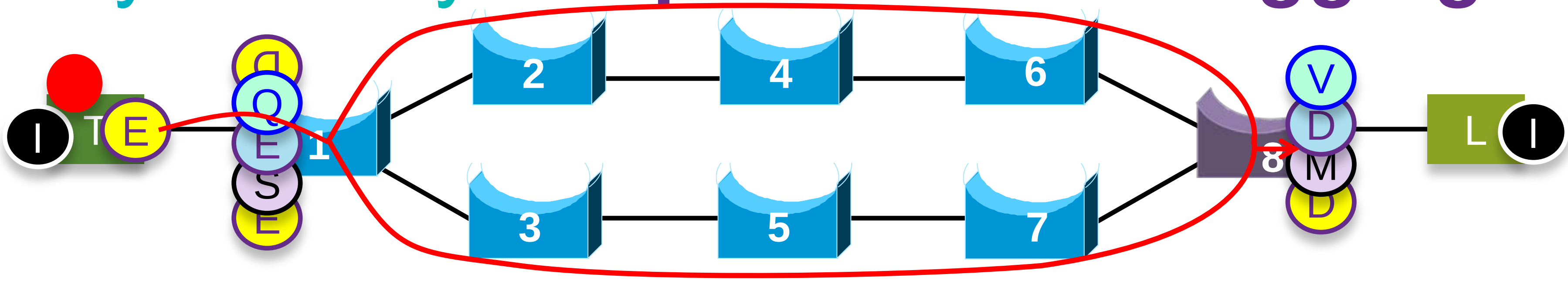


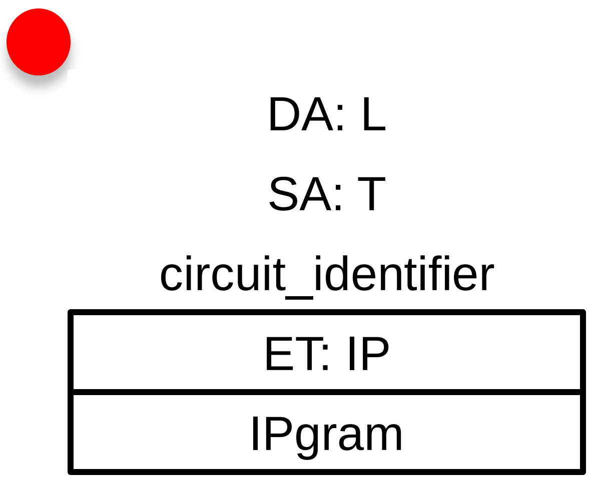
- In this example, the **Listener** system is TSN-unaware.
- And the Sequencing, Merge, and TSN Decaps are all in Bridge 8 (below the link).

Sequenced TSN tagging

- Day-in-the-life view

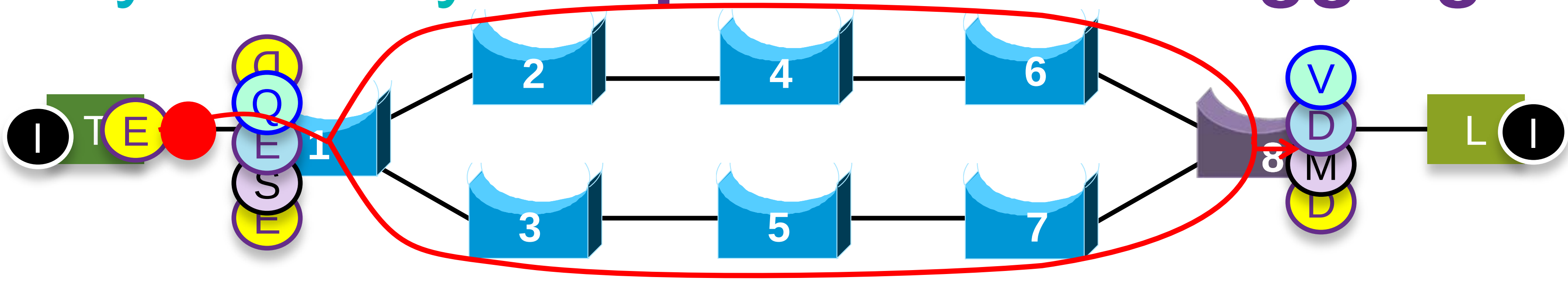
Layer 2 only: **Sequenced TSN tagging**





- Talker's stack is not VLAN-aware. This is what the frame is when it hits the TSN Encaps layer.
- Note that Bridge 1 would normally add a **VLAN 80 tag** to this frame.

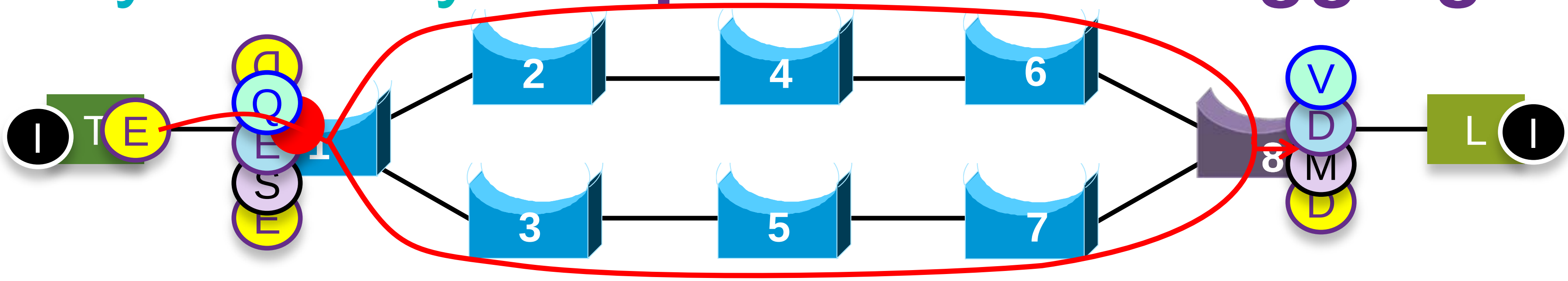
Layer 2 only: **Sequenced TSN tagging**



DA: TSN 734
SA: T
VLAN tag 99
ET: IP
IPgram

- Talker is TSN-aware, so the TSN Encaps layer (E) adds a VLAN tag, even though Talker's stack is not VLAN-aware.
- Talker could add sequence number, but doesn't.

Layer 2 only: **Sequenced TSN tagging**



DA: L

SA: T

circuit_identifier 734[99]

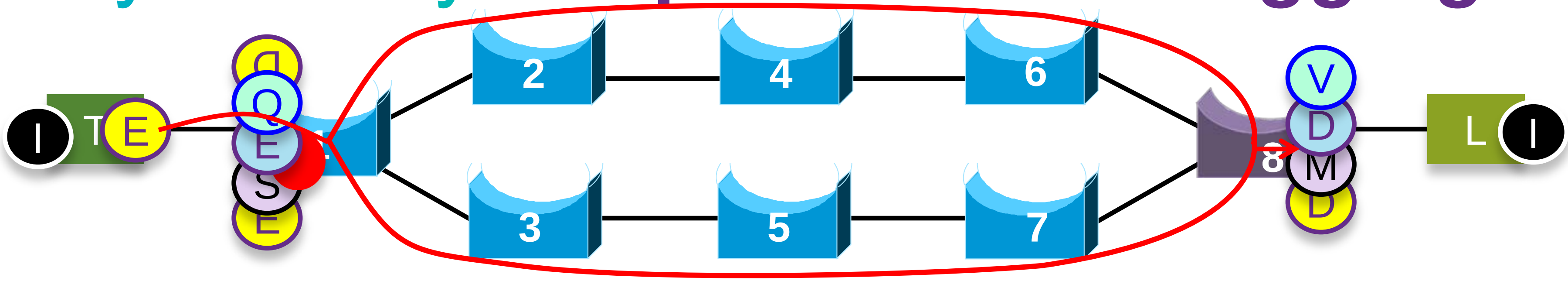
vlan_identifier 80

sequence_number

ET: whatever
data

- The Sequencing function Q adds a new TSN sequence number.
- (Notionally, the DA/VLAN have been restored. In practice, one would not bother.)

Layer 2 only: **Sequenced TSN tagging**



DA: L

SA: T

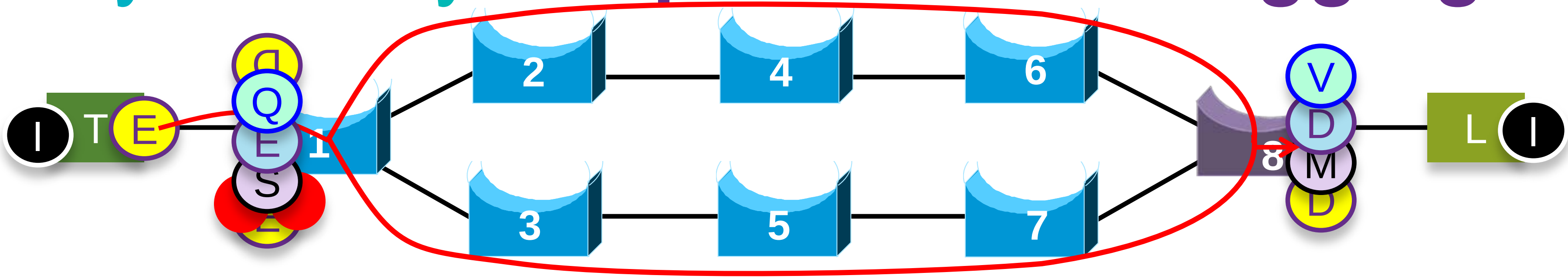
circuit_identifier 734[99]

vlan_identifier 80

ET: TSN Seq
Sequence #
ET: whatever
data

- The Sequencing encaps function (E) replaces the sequence_number parameter with a new TSN sequence number tag, to be defined by IEEE 802.1.

Layer 2 only: **Sequenced TSN tagging**



DA: L

SA: T

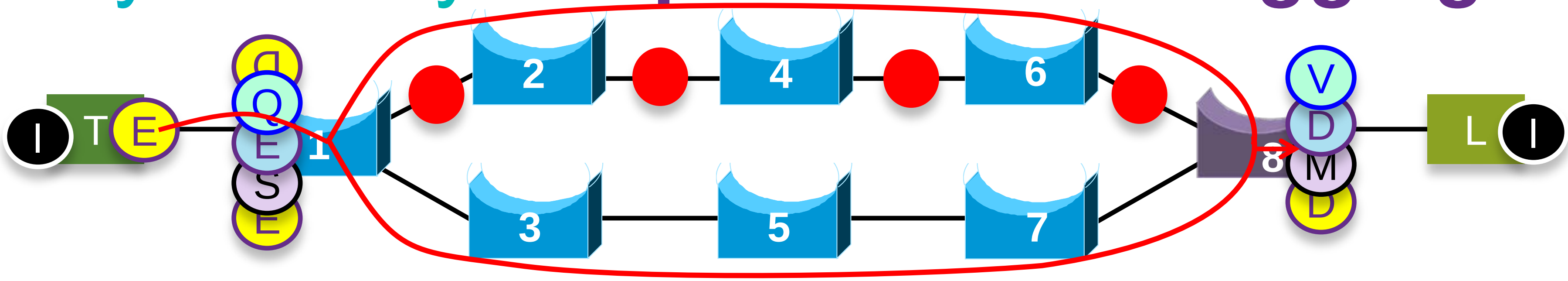
circuit_ID 7840[23] or 12[50]

vlan_identifier 80

ET: TSN Seq
Sequence #
ET: whatever
data

- The Split (S) function creates two packets, with different circuit_identifiers.

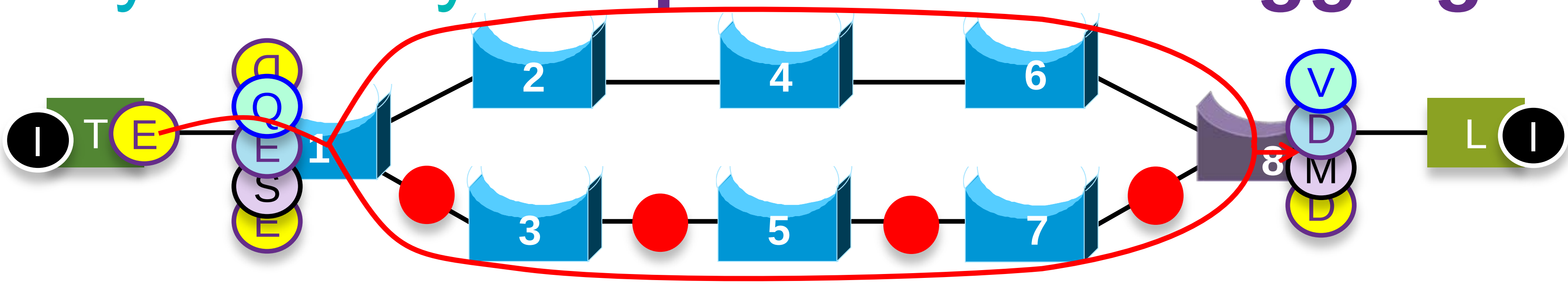
Layer 2 only: **Sequenced TSN tagging**



DA: TSN 7840
SA: T
VLAN tag 23
ET: TSN Seq
Sequence #
ET: whatever
data

- After being encoded, again, (E) this is the Ethernet frame on the upper path.

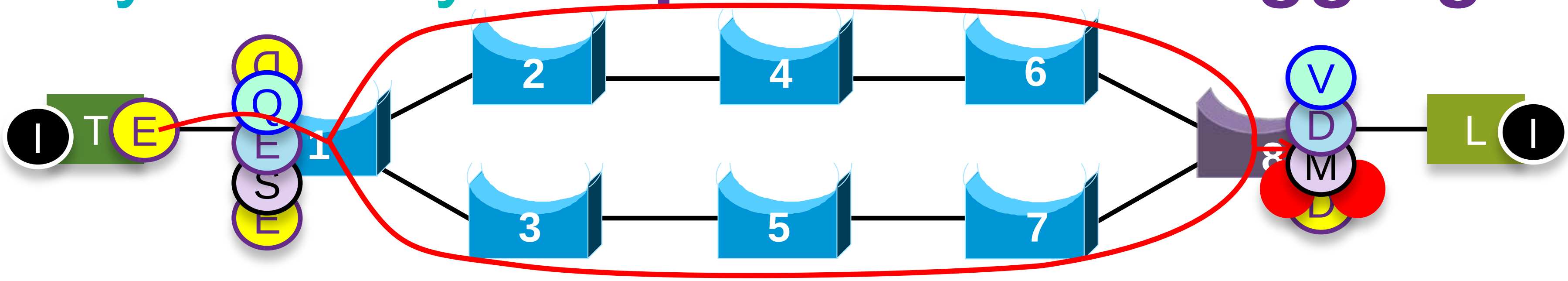
Layer 2 only: **Sequenced TSN tagging**



DA: TSN 12
SA: T
VLAN tag 50
ET: TSN Seq
Sequence #
ET: whatever
data

- Note that we have a different circuit ID on the second path.
- Another presentation is required to discuss whether the DA, the VLAN, both, or neither, should be different.

Layer 2 only: **Sequenced TSN tagging**



DA: L

SA: T

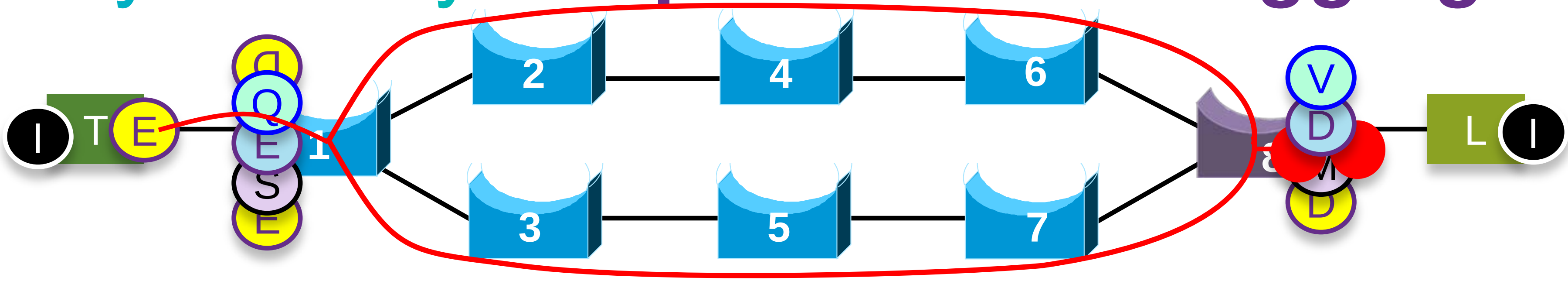
circuit_ID 7840[23] or 12[50]

vlan_identifier 80

ET: TSN Seq
Sequence #
ET: whatever
data

- The TSN Decaps function (D) restores the proper DA and VLAN, and extracts the circuit_identifier.

Layer 2 only: **Sequenced TSN tagging**



DA: L

SA: T

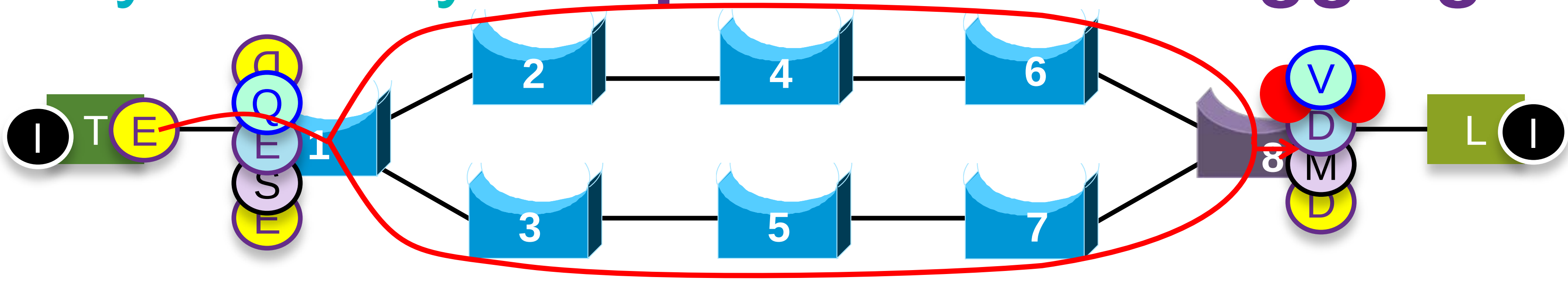
circuit_identifier 734[99]

vlan_identifier 80

ET: TSN Seq
Sequence #
ET: whatever
data

- The Merge function Ⓜ takes all packets and gives them the same circuit_identifier.
- (It is the same as on the Talker T to Bridge 1 link.)
- (There are still 2 packets.)

Layer 2 only: **Sequenced TSN tagging**



DA: L

SA: T

circuit_identifier 734[99]

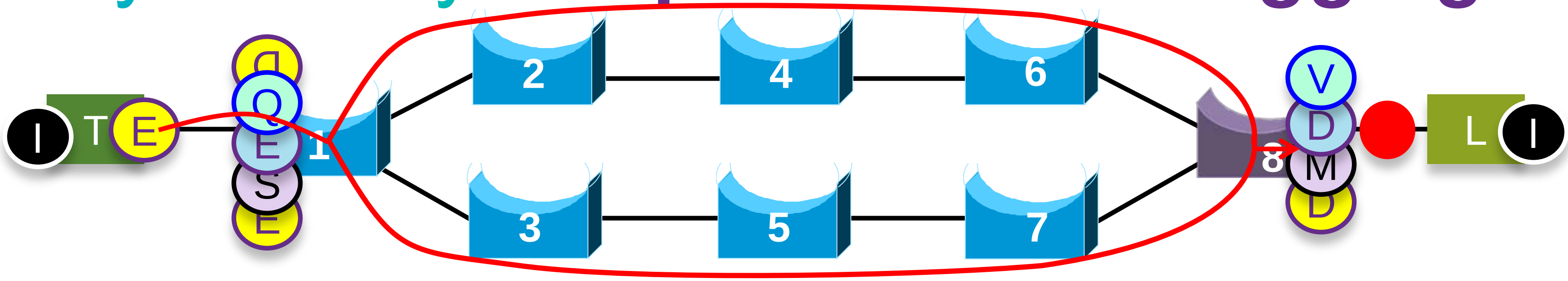
vlan_identifier 80

sequence_number

ET: whatever
data

- The Sequencing Decapsulation function (D) exposes the sequence_ number so that the Sequence Discard function (V) can discard the duplicates.
- (There are still 2 packets.)

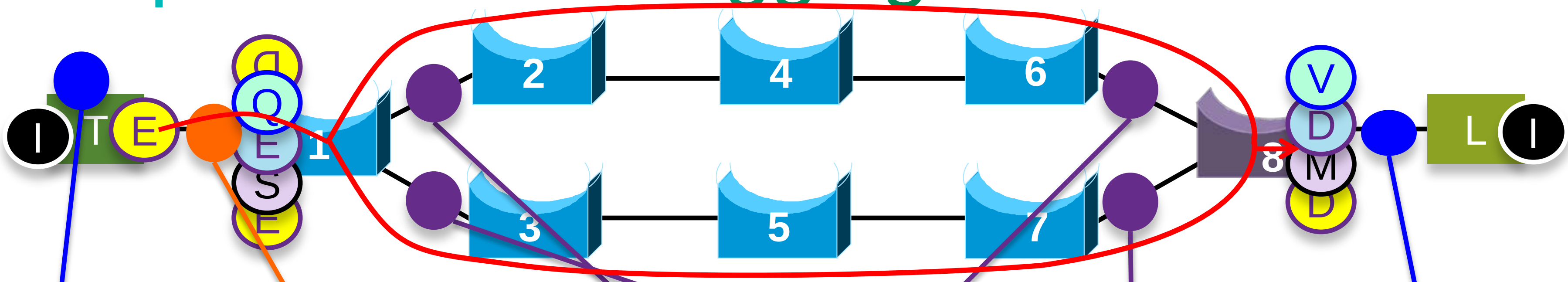
Layer 2 only: **Sequenced TSN tagging**



DA: L
SA: T
VLAN tag 80
ET: whatever
data

- A single frame is output from Sequencing function (V) .
- It is what would have come from the Talker, modulo the VLAN tag changes the bridges would make.

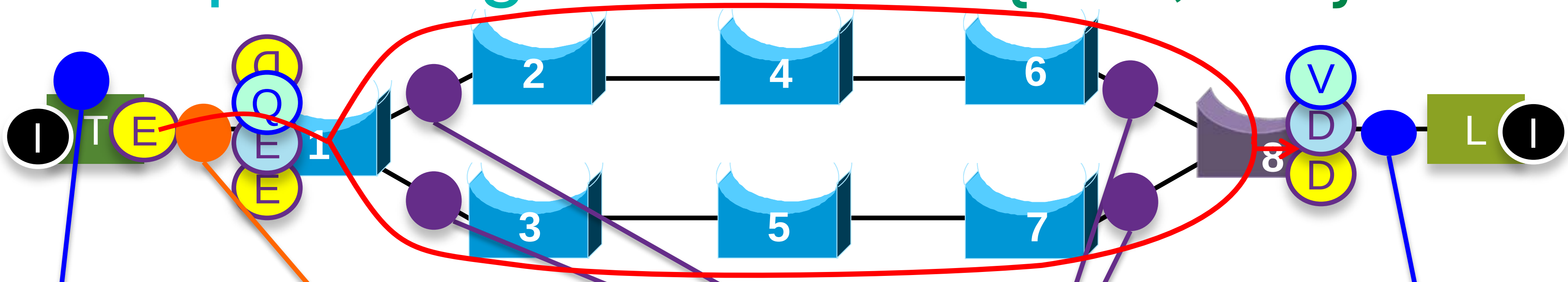
Summary:
Sequenced TSN tagging:



- This uses the full Split/Merge functionality with different cicuit_identifiers on the paths.

		DA: TSN 7840	DA: TSN 12	
		SA: T	SA: T	
DA: Listener L	DA: TSN 734	VLAN tag 23	VLAN tag 50	DA: L
SA: Talker T	SA: T	ET: TSN Seq	ET: TSN Seq	SA: T
circuit_ID	VLAN tag 99	Sequence #	Sequence #	VLAN 80
ET: whatever	ET: whatever	ET: whatever	ET: whatever	ET: whatever
data	data	data	data	data

Variant 1:
No Split/Merge – all same {VID, DA}



- This uses the full Split/Merge functionality with different cicuit_identifiers on the paths.

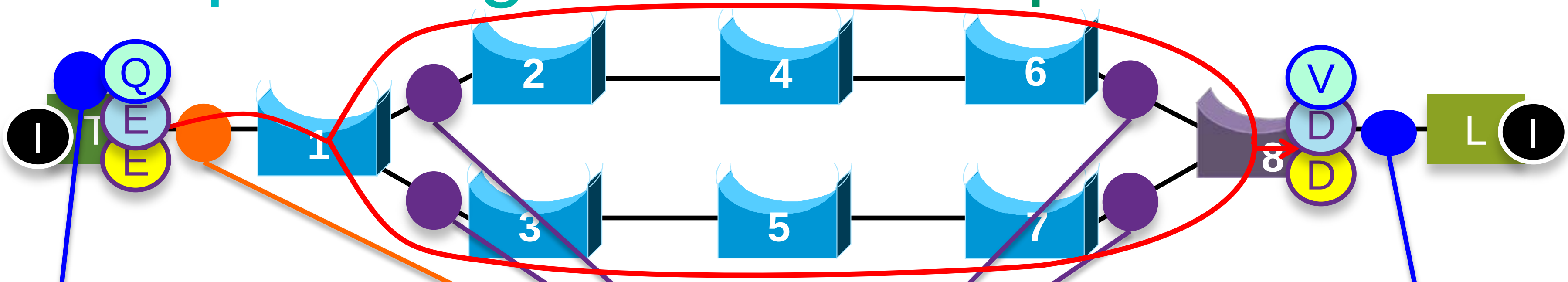
DA: Listener L
SA: Talker T
circuit_ID
ET: whatever
data

DA: TSN 734
SA: T
VLAN tag 99
ET: whatever
data

DA: TSN 734
SA: T
VLAN tag 99
ET: TSN Seq
Sequence #
ET: whatever
data

DA: L
SA: T
VLAN 80
ET: whatever
data

Variant 2:
No Split/Merge – Talker sequences



- If Talker T does the sequencing and encaps, and all paths use the same encaps, **it gets really simple!**

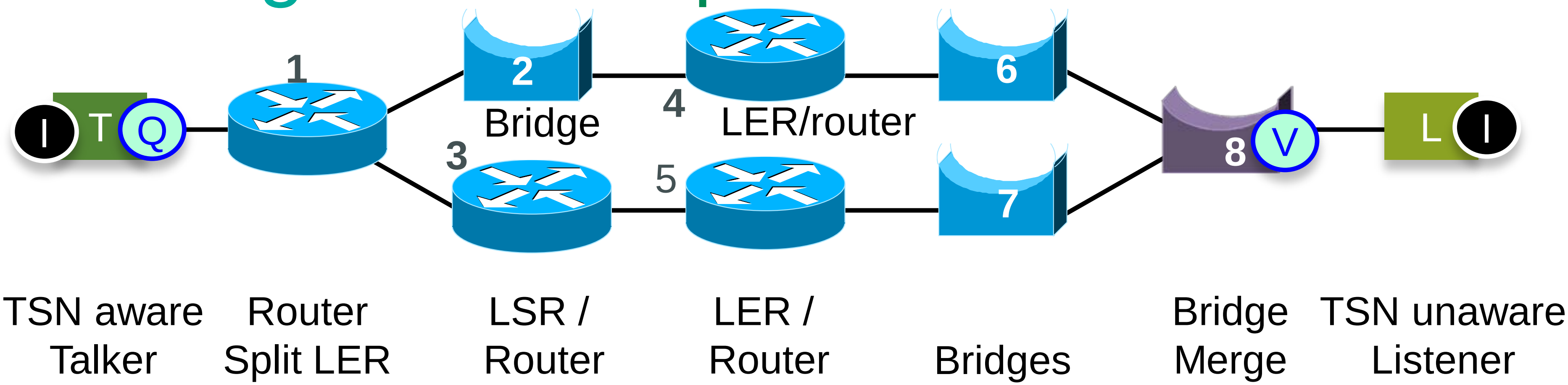
DA: Listener L
SA: Talker T
circuit_ID
ET: whatever
data

DA: TSN 734
SA: T
VLAN tag 99
ET: TSN Seq
Sequence #
ET: whatever
data

DA: L
SA: T
VLAN 80
ET: whatever
data

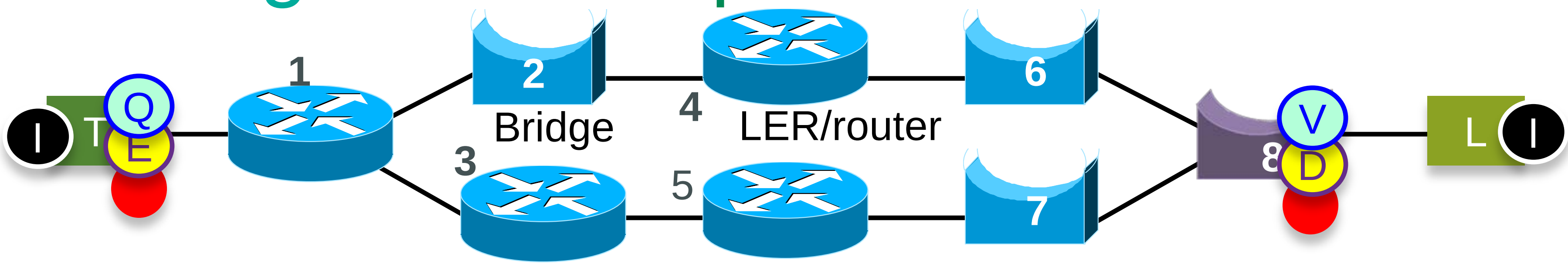
Case 2:
Mixed L2/L3 using IPgram pseudowires and Sequenced TSN

Peering relationships



- Single-port TSN- VLAN-aware Talker T and a single-port TSN- VLAN-unaware Listener L.
- The Talker sequences Q, and peers to the Discard V in Bridge 8.
- Talker attached to a router; Listener to a bridge.
- A network with a variety of routers and bridges.

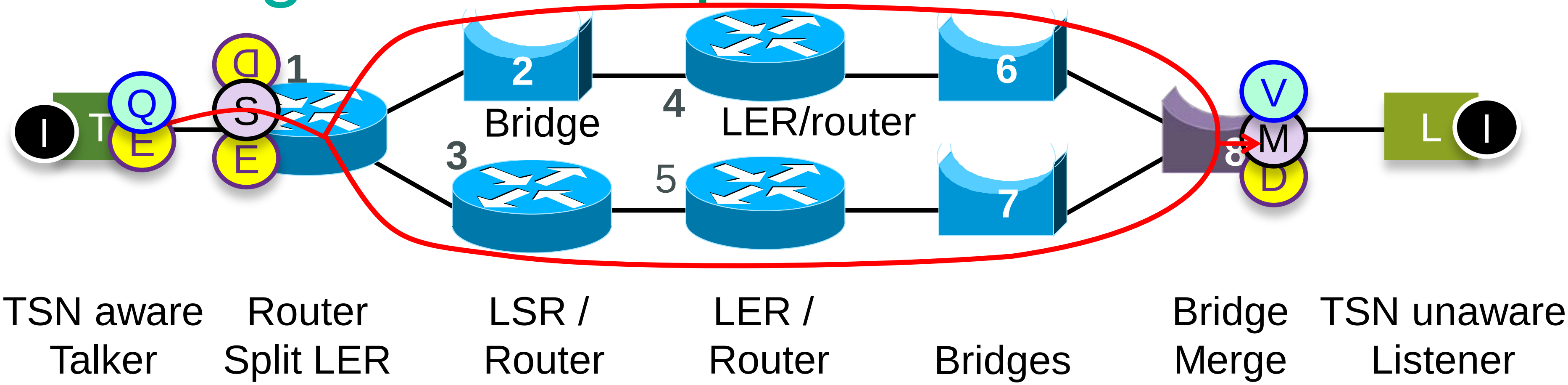
Peering relationships



pseudowire label 28
control (sequence)
IPgram

- Talker T and Bridge 8 have chosen to use an **IPgram pseudowire** (D) (E) for the circuit.

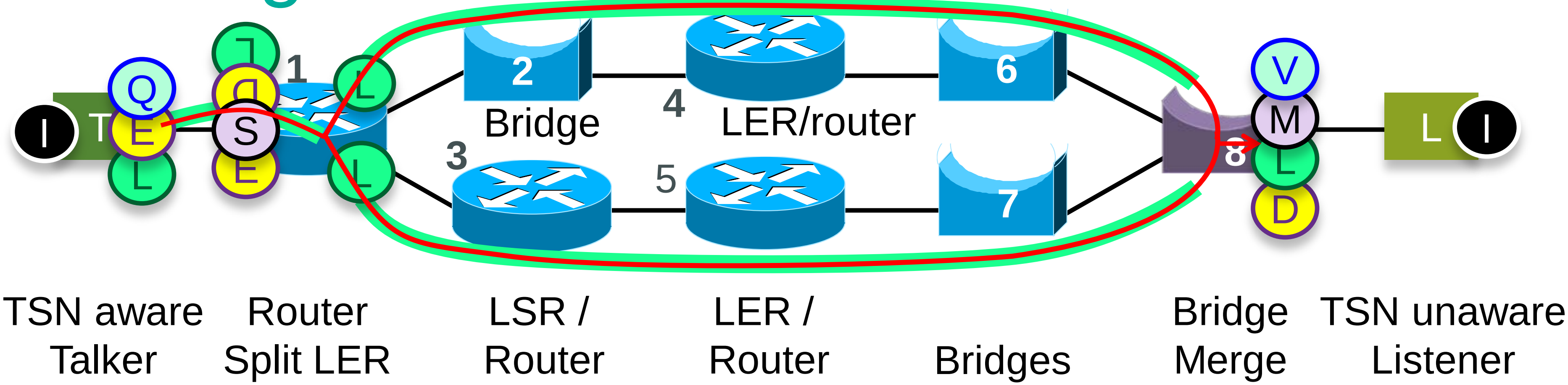
Peering relationships



S M Router 1 and Bridge 8 are the **split/merge (seamless redundancy)** peers, because they split and merge the circuits.

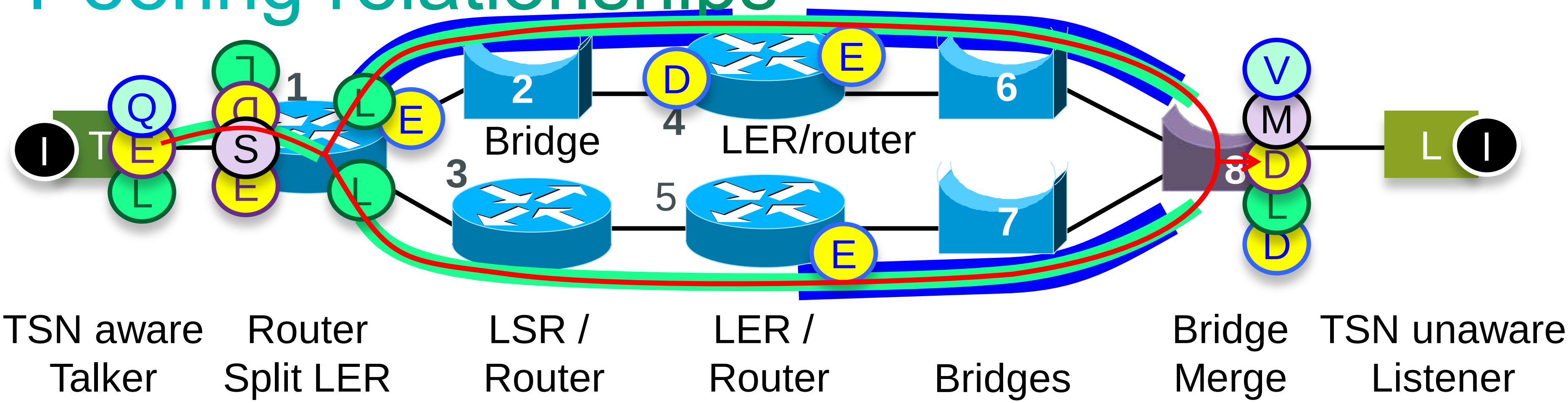
- (Inserting the Split function in Router 1 requires an extra Encode/Decode E D pair.)

Peering relationships



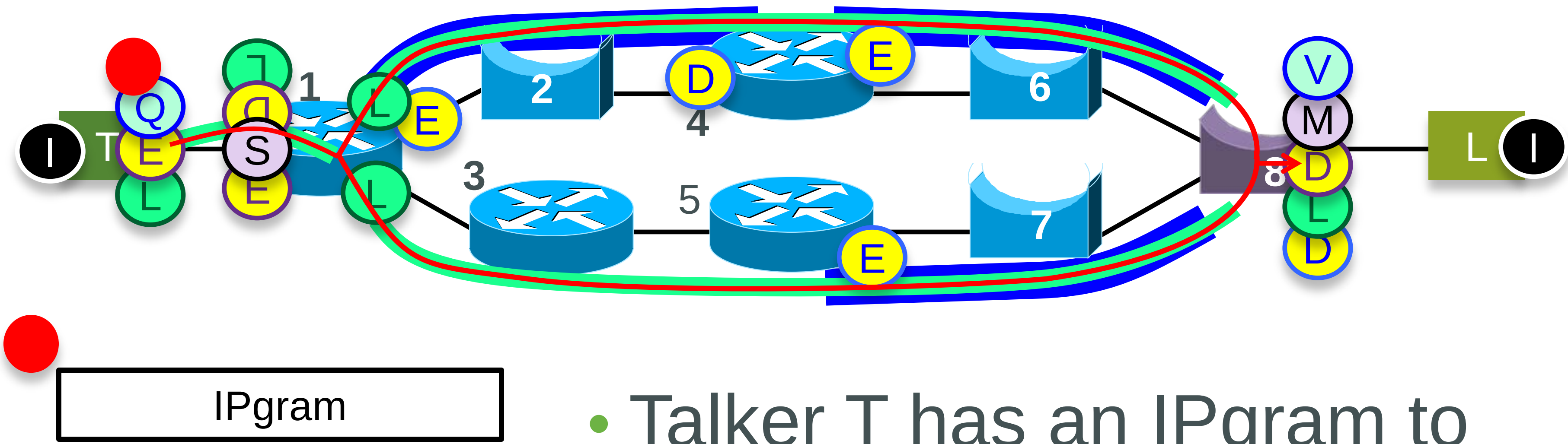
Assuming that the encode/decode used by the Split/Merge (S) (M) are pseudowires, we require a network of **Label Switched Paths (LSPs)** to connect T to (S) to (M). Each endpoint is a Label Edge Router (LER) function.

Peering relationships



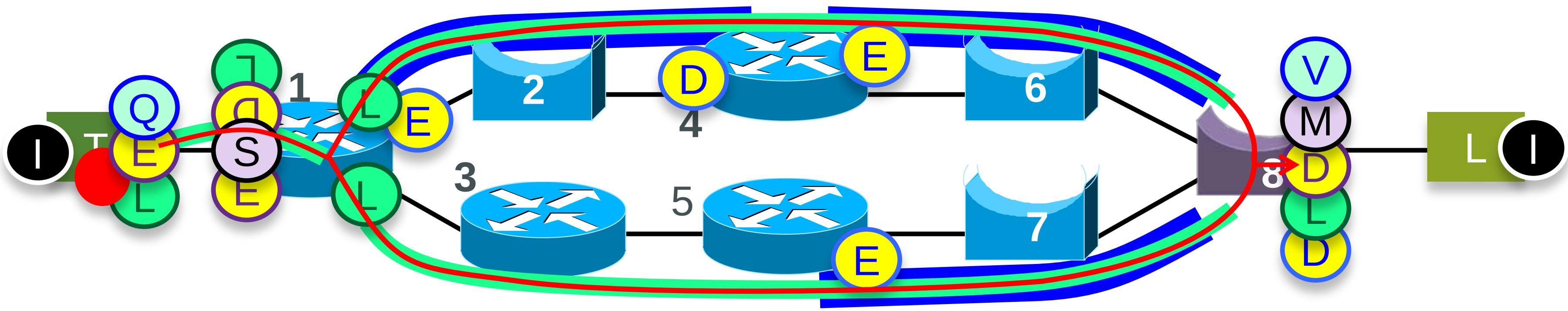
- E / D TSN Circuit Encaps/Decaps pairs are needed to convey the circuit over the various Bridged LANs. (Note the **blue**, instead of **purple**, letter and outline.)

Mixed L2/L3: **IPgram pseudowire**



- Talker T has an IPgram to send to Listener L.

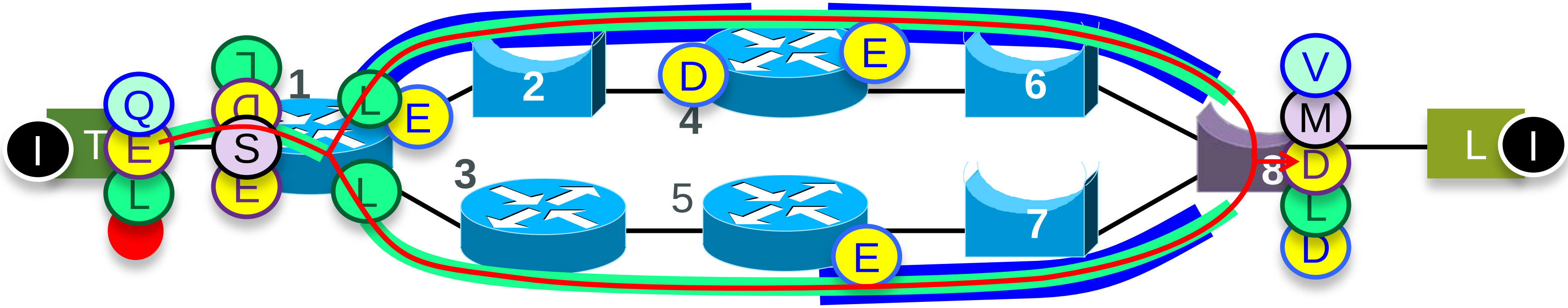
Mixed L2/L3: **IPgram pseudowire**



pseudowire label 28
control (sequence)
IPgram

- Talker T's combined TSN Encaps (E) and Sequencing (Q) functions use an **IPgram pseudowire** for the circuit.
- Bridge 8's functions (D) (V) are at the other end of the network.

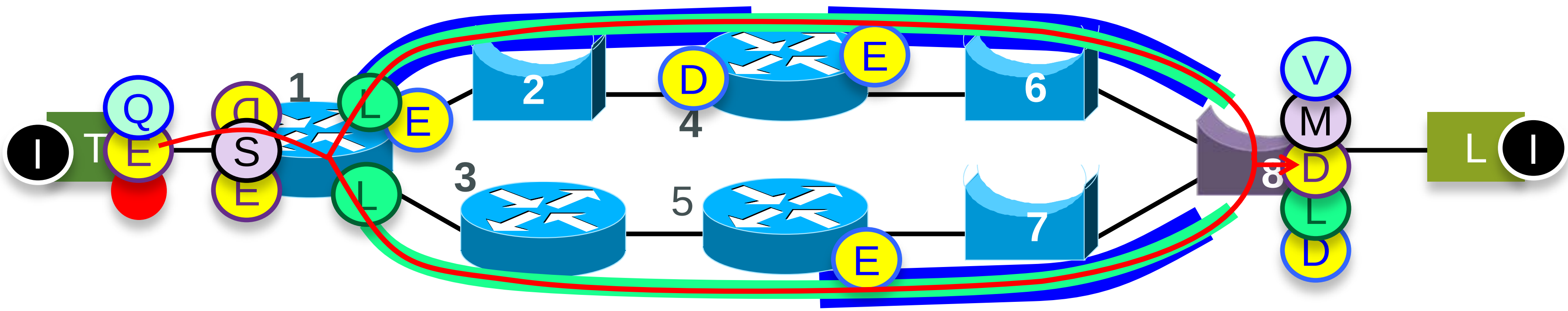
Mixed L2/L3: **IPgram pseudowire**



label 60
pseudowire label 28
control (sequence)
IPgram

- In the general case, the LER function (L) would encapsulate the pseudowire would be carried in an LSP.

Mixed L2/L3: **IPgram pseudowire**



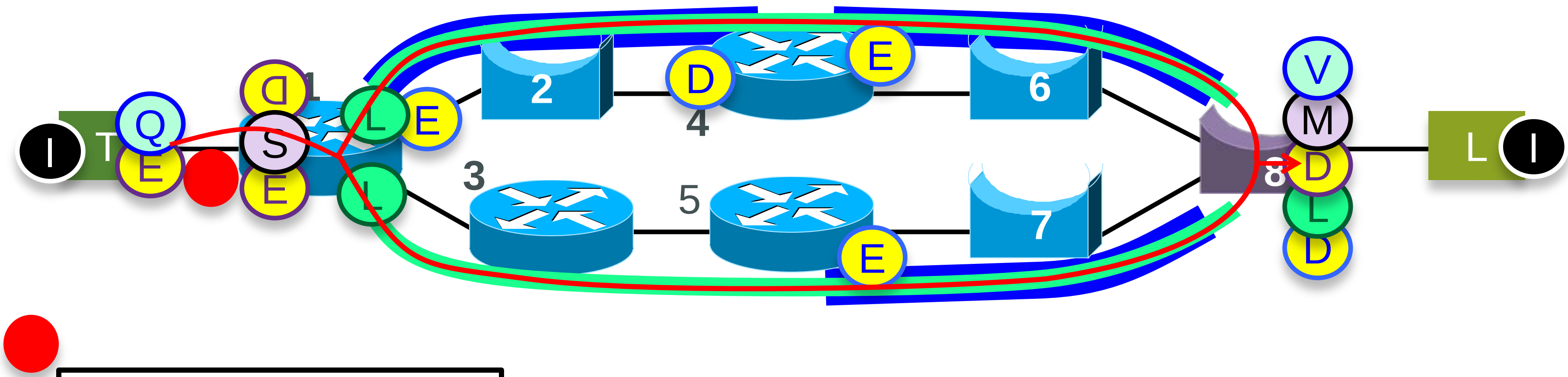
~~label 60~~
pseudowire label 28
control (sequence)
IPgram

- In this particular case, we will assume that Router 1 is doing a “Penultimate Hop Pop” (PHP) function. That eliminates the need for the outside label encaps L L.

Warning

- The PHP step may be controversial.
- Perhaps there is another MPLS label, a path label, on the frame between the Talker and Router 1.

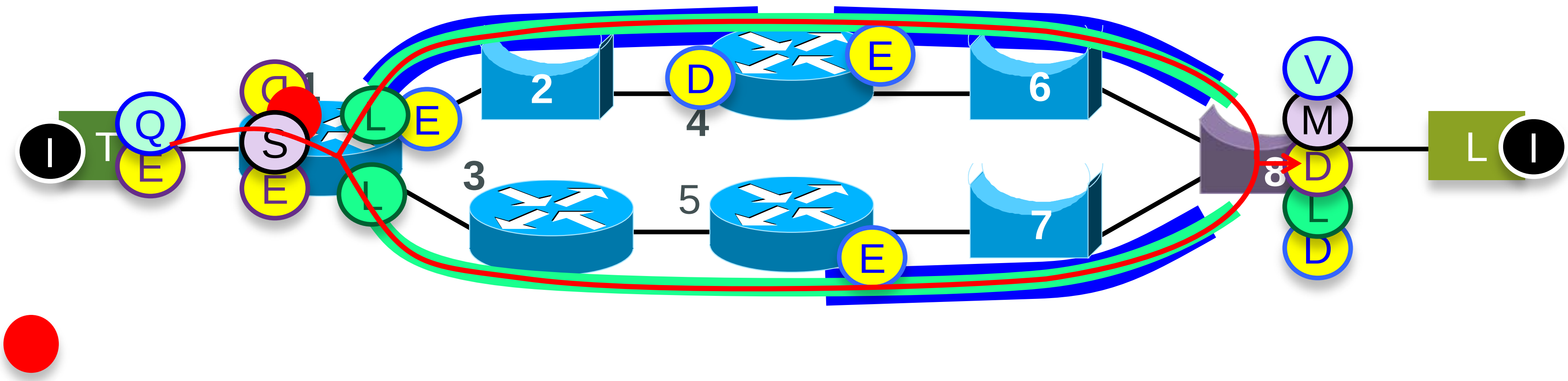
Mixed L2/L3: **IPgram pseudowire**



DA: Router 1
SA: T
ET: MPLS
pseudowire label 28
control (sequence)
IPgram

- So, the frame from Talker T to Router 1 looks like this on the Ethernet between Talker T and Router 1.

Mixed L2/L3: **IPgram pseudowire**



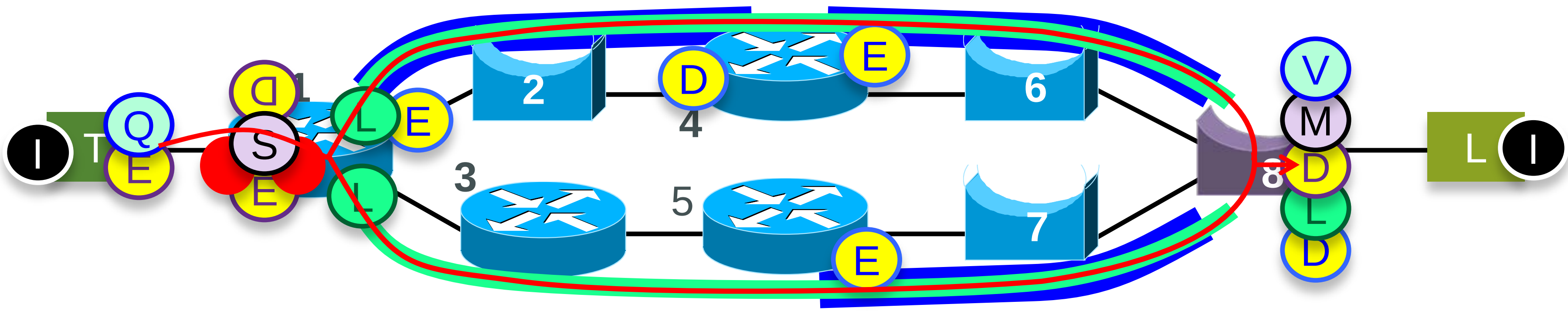
circuit_ID (psw 28)

sequence_# (control)

IPgram

- The Splitter function (S) in Router 1 is given the IPgram.

Mixed L2/L3: **IPgram pseudowire**



circuit_ID (psw 419)

sequence_# (control)

IPgram

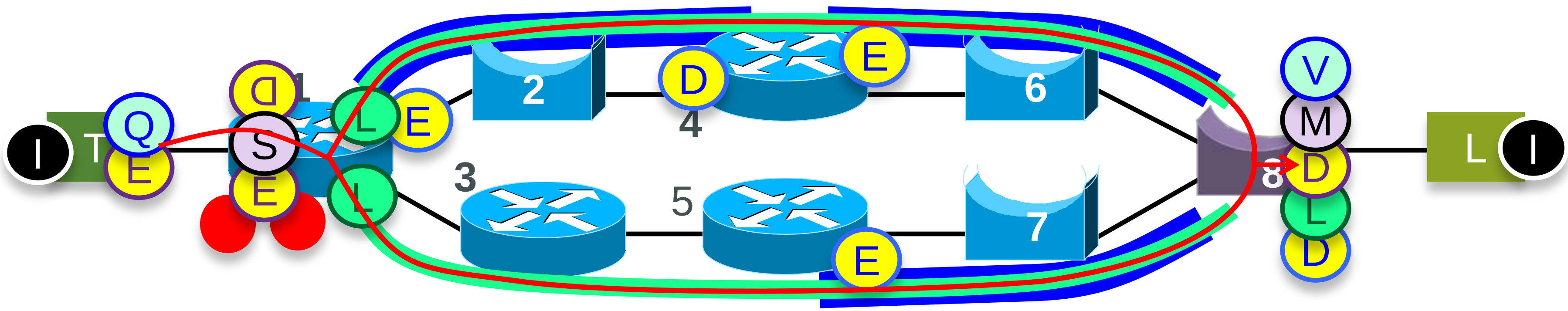
circuit_ID (psw 31)

sequence_# (control)

IPgram

- The Splitter function (S) has split the one pseudowire 28 into two pseudowires 419 and 31, copying the one control word to both of them.

Mixed L2/L3: **IPgram pseudowire**

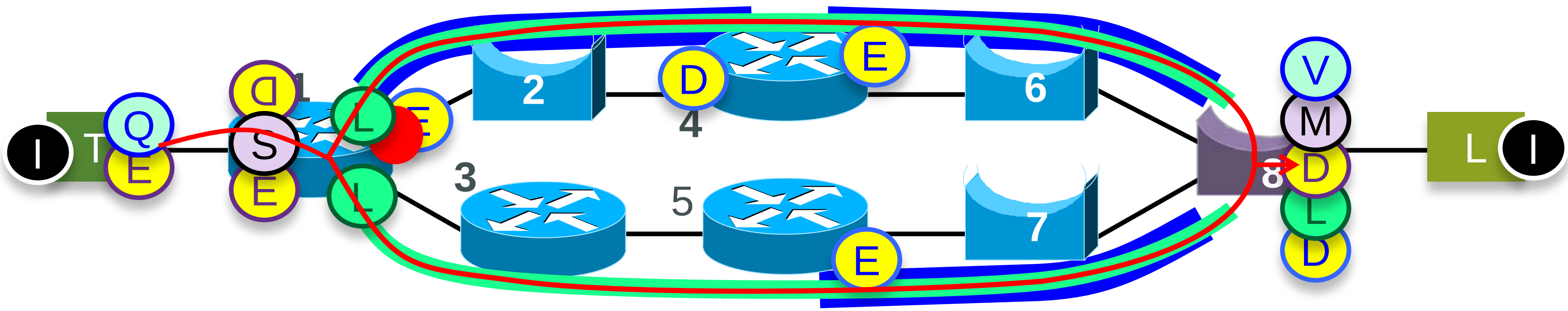


pseudowire label 419
control (sequence)
IPgram

pseudowire label 31
control (sequence)
IPgram

- The TSN Encapsulation function E (an IPgram pseudowire encaps) generates these two packets, ready to enter the two LSPs.

Mixed L2/L3: **IPgram pseudowire**



DA: Router 4

SA: Router 1

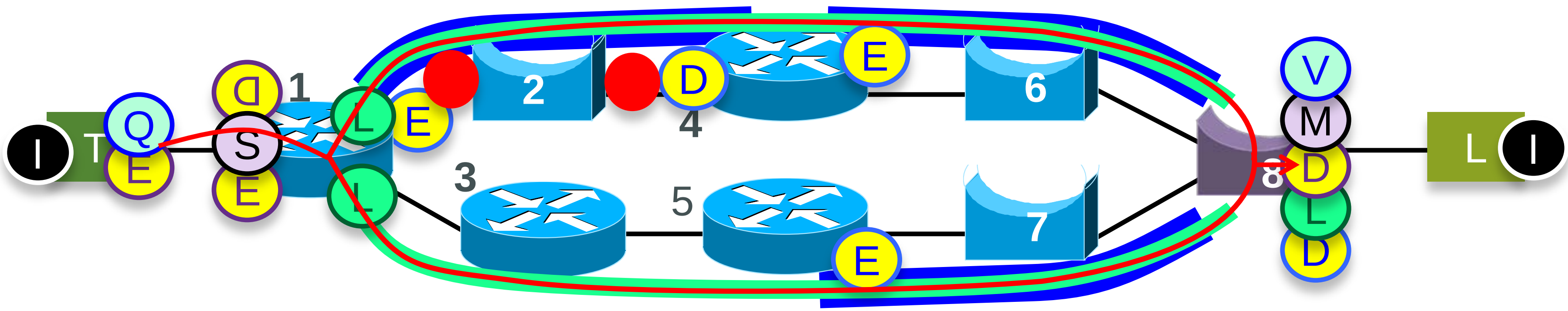
vlan_identifier 15

circuit_identifier

ET: MPLS
Tunnel label 51
pseudowire label 419
control (sequence)
IPgram

- The upper tunnel looks like this, when labeled with Tunnel 51, and before applying the TSN Encapsulation. This would be the usual Ethernet frame from Router 1 to Router 4

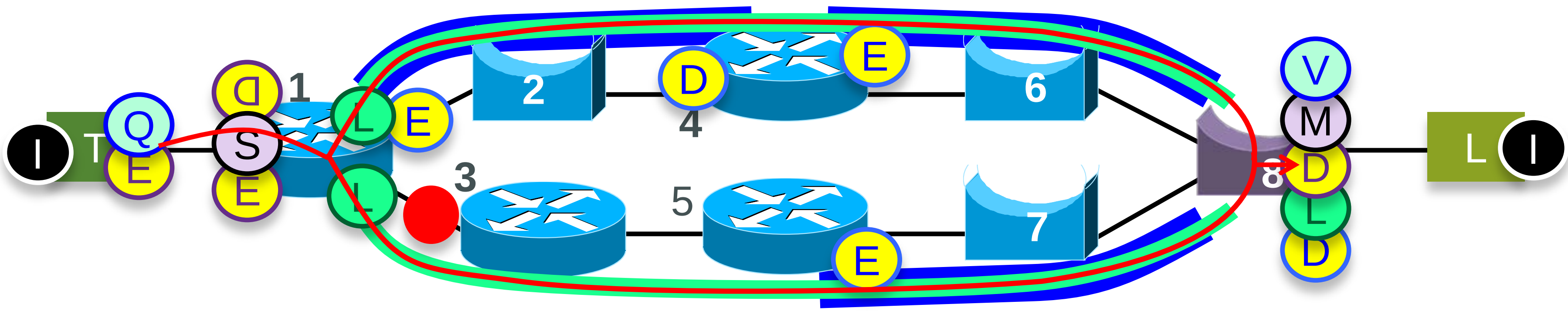
Mixed L2/L3: **IPgram pseudowire**



DA: TSN 140
SA: Router 1
VLAN tag 309
ET: MPLS
Tunnel label 51
pseudowire label 419
control (sequence)
IPgram

- But, Router 1 and Router 4 are separated by a TSN bridged network, so require a TSN encapsulation D E .
- This gets the packet to Router 4.

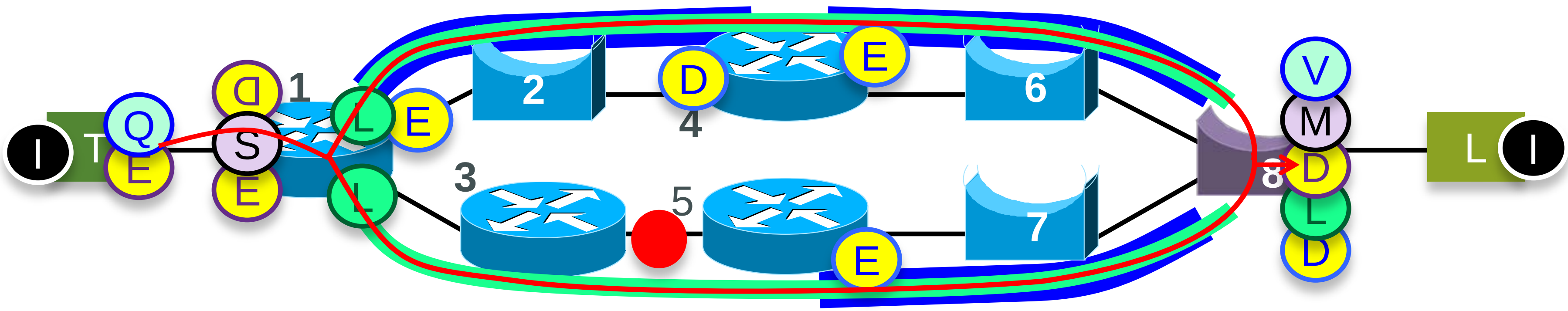
Mixed L2/L3: **IPgram pseudowire**



DA: Router 3
SA: Router 1
ET: MPLS
Tunnel label 557
pseudowire label 31
control (sequence)
IPgram

- Meanwhile, Router/LER 1, Router/LSR 3 and Router/LSR 5 are moving the second LSP packet along.
- No TSN encaps is needed in the absence of bridges.

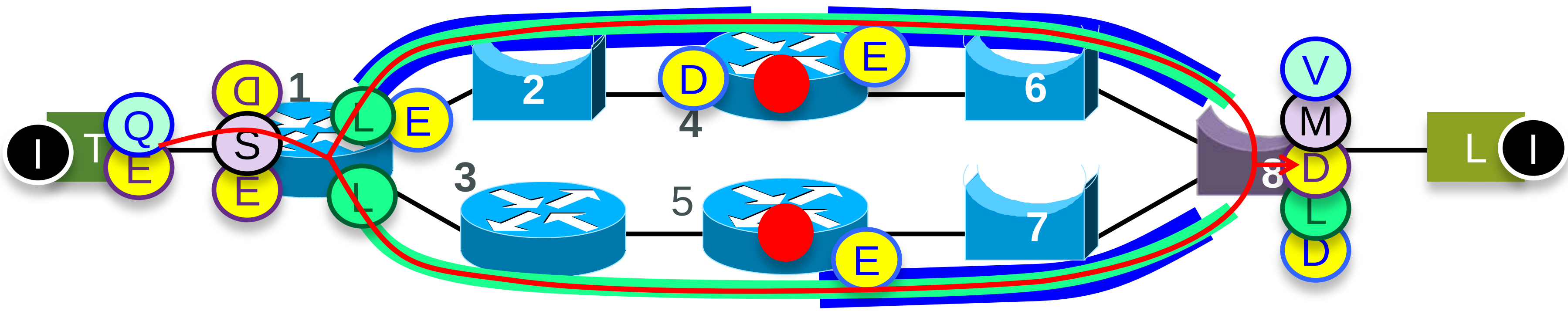
Mixed L2/L3: **IPgram pseudowire**



DA: Router 5
SA: Router 3
ET: MPLS
Tunnel label 346
pseudowire label 31
control (sequence)
IPgram

- Meanwhile, Router/LER 1, Router/LSR 3 and Router/LSR 5 are moving the LSP packet along.
- Router/LSR 3 changes the Tunnel label 557→346.

Mixed L2/L3: **IPgram pseudowire**



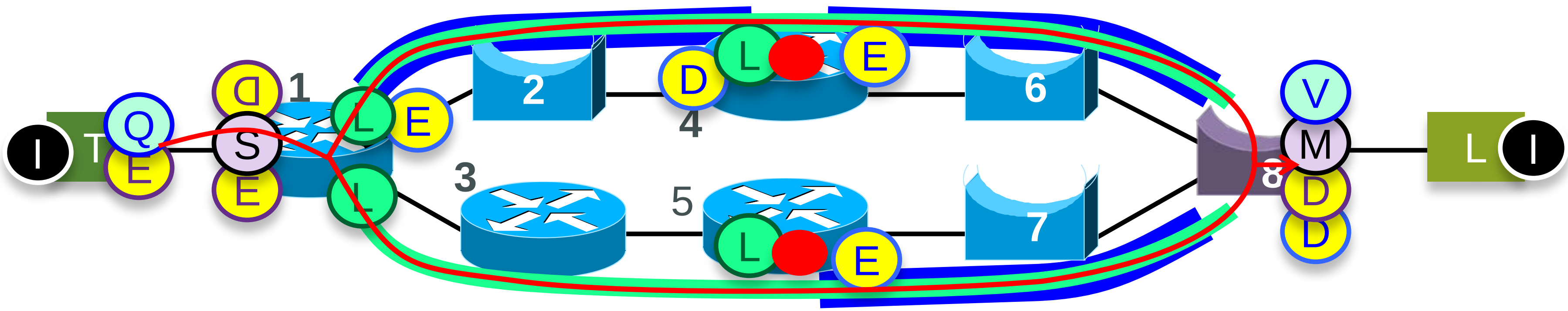
Tunnel label 51
pseudowire label 419
control (sequence)
IPgram

- After TSN decapsulation (D), Router 4 has this labeled packet.

Tunnel label 346
pseudowire label 31
control (sequence)
IPgram

- And Router 5 has this one.

Mixed L2/L3: **IPgram pseudowire**

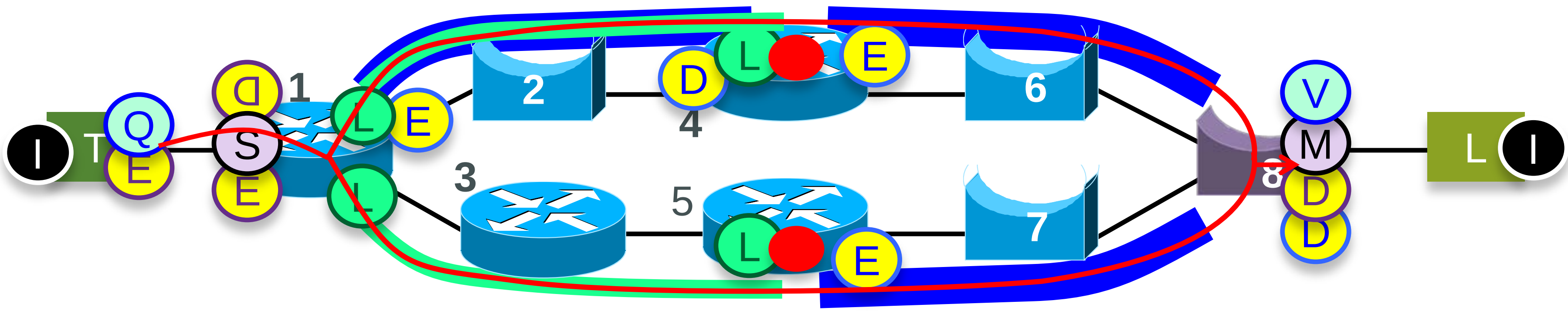


pseudowire label 419
control (sequence)
IPgram

pseudowire label 31
control (sequence)
IPgram

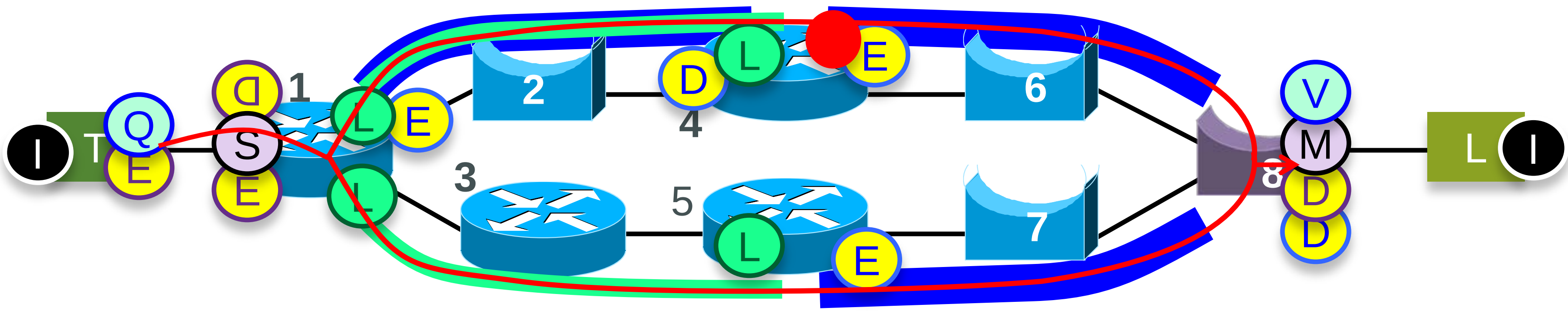
- For the sake of reduced frame size, Router/LSPs 4 and 5 perform PHP, which eliminates Tunnel labels 51 and 346 (and the LERs L in Bridge 8).

Mixed L2/L3: **IPgram pseudowire**



- One can argue the semantics of the green tunnels. In theory, each tunnel continues to its natural end at Bridge 8. The control plane may maintain this. But, in the data plane, the tunnel label disappears.
- So, we will shorten the tunnel in the diagram to match the data plane encapsulation

Mixed L2/L3: **IPgram pseudowire**



DA: Listener L

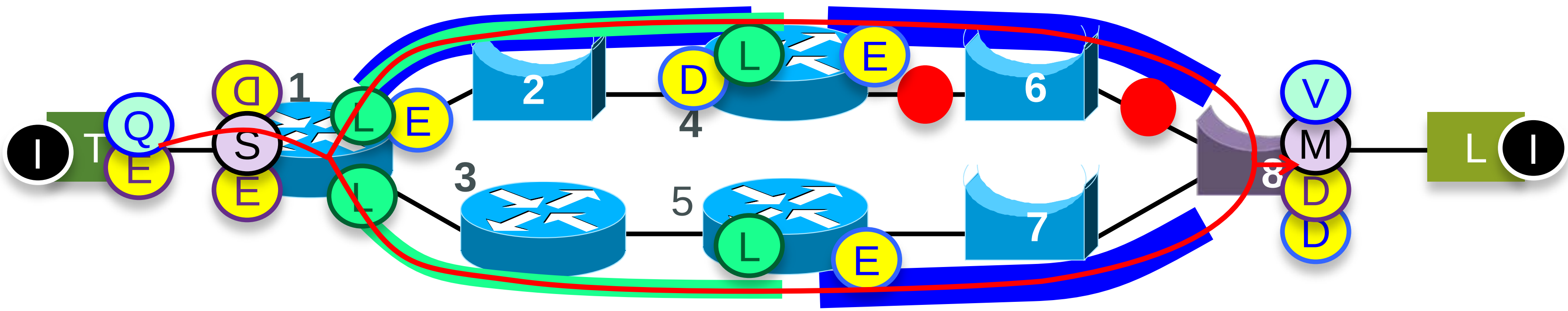
SA: Router 4

vlan_identifier 80

ET: MPLS
pseudowire label 419
control (sequence)
IPgram

- Router 4 prepares this Ethernet frame to transmit the pseudowire packet.

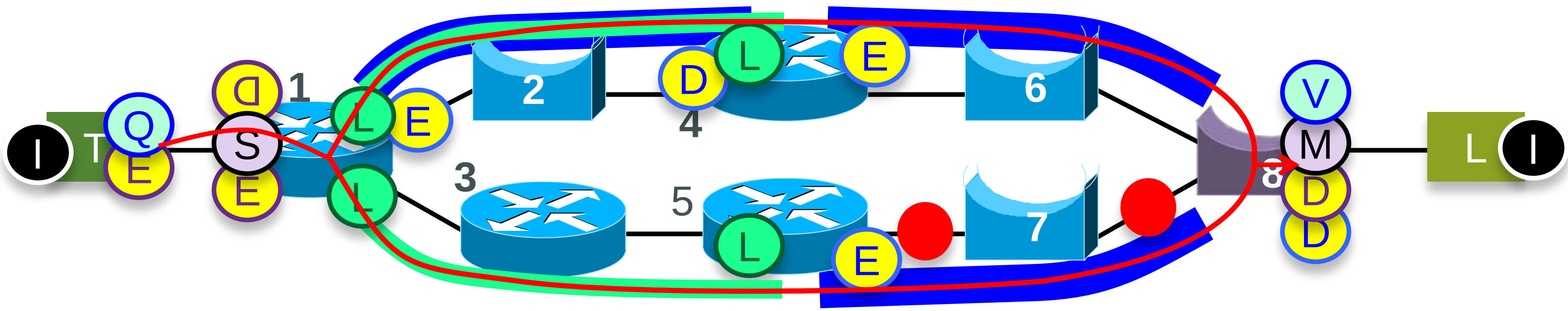
Mixed L2/L3: **IPgram pseudowire**



DA: TSN 994
SA: Router 4
VLAN tag 7
pseudowire label 419
control (sequence)
IPgram

- And Router 4's TSN Encaps function (E) produces this.

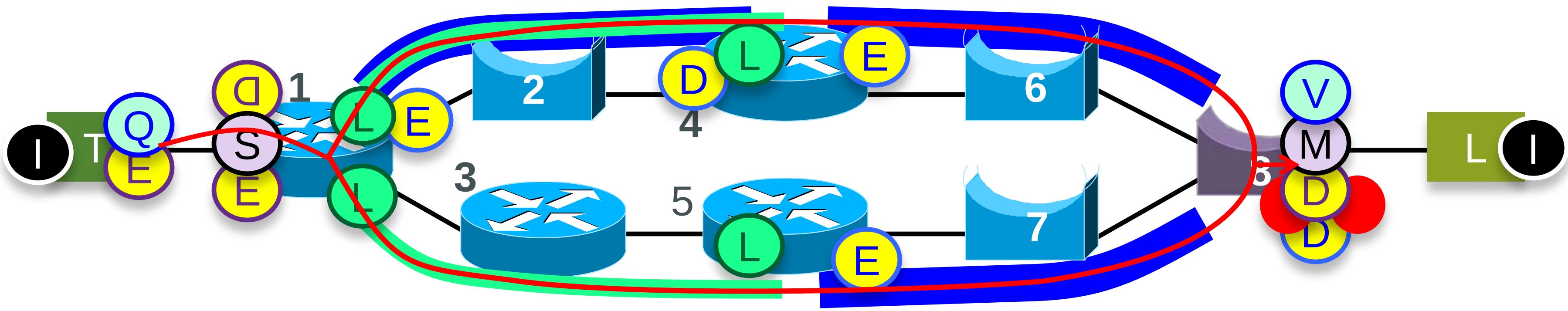
Mixed L2/L3: **IPgram pseudowire**



DA: TSN 2006
SA: Router 5
VLAN tag 18
pseudowire label 31
control (sequence)
IPgram

- And IPgram pseudowire label 31 is translated by Router 5’s TSN Encaps into this.

Mixed L2/L3: **IPgram pseudowire**



DA: Listener L

SA: Router 4 or 5

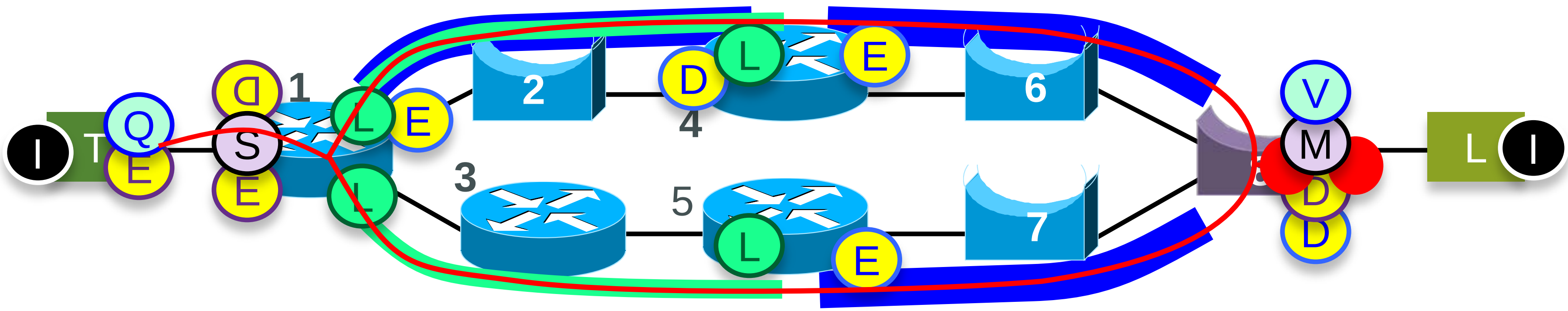
vlan_identifier 80

circ_ID 994[7] or 2006[18]

pseudowire label 419
control (sequence)
IPgram

- The TSN Decaps function (D) exposes the pseudowire and restores the Ethernet frame parameters.

Mixed L2/L3: **IPgram pseudowire**



DA: Listener L

SA: Router 4 or 5

vlan_identifier 80

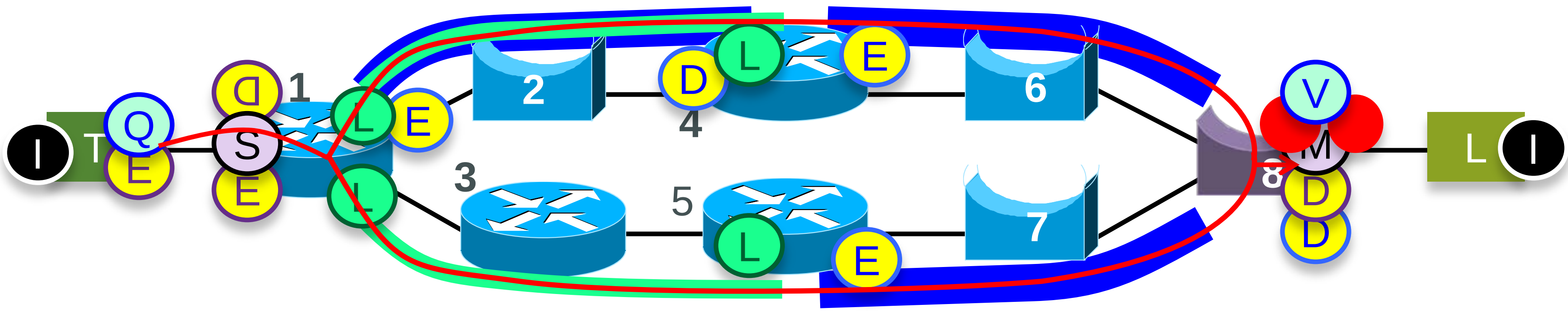
circuit_ID psw 419 or 31

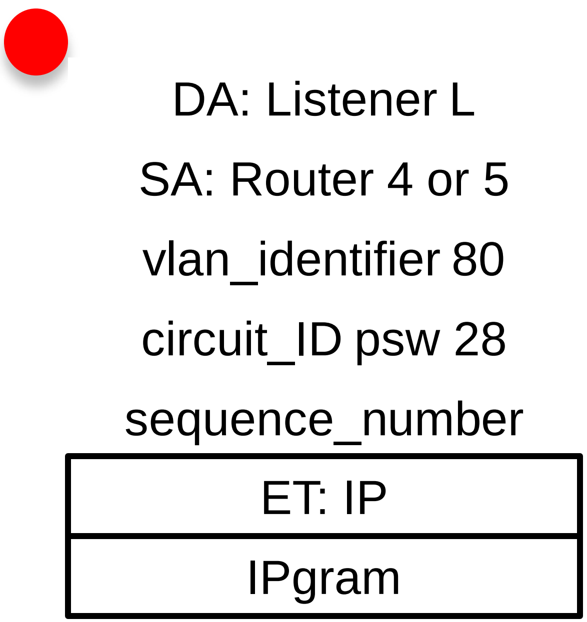
sequence_number

ET: IP
IPgram

- The Merge function (M) has to operate on the circuit and sequence number after the pseudowire Decaps function (D) makes these parameters available (and adds the IP EtherType).

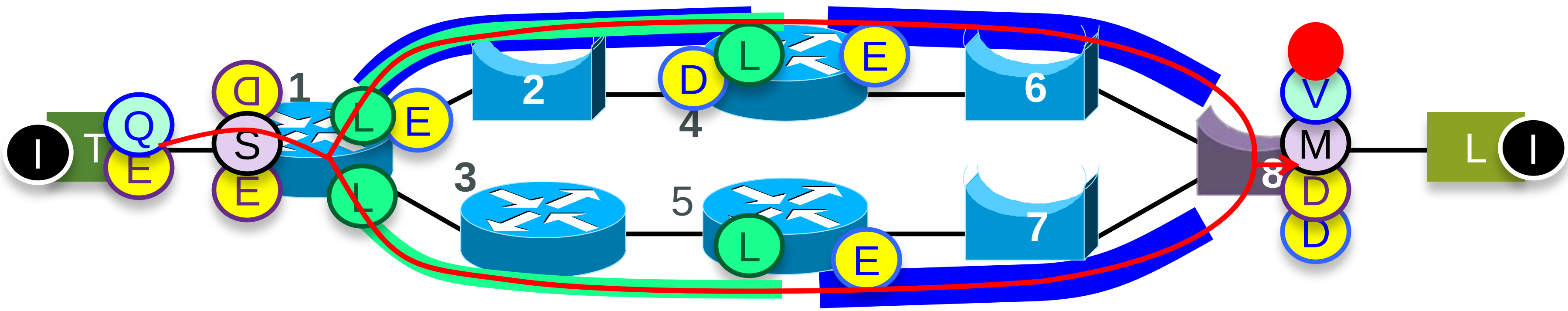
Mixed L2/L3: **IPgram pseudowire**





- Output from Merge function (M)
- Pseudowire labels 419 and 31 have been combined into the original pseudowire label 28.
- There are still two packets!

Mixed L2/L3: **IPgram pseudowire**



DA: Listener L

SA: Router 4

vlan_identifier 80

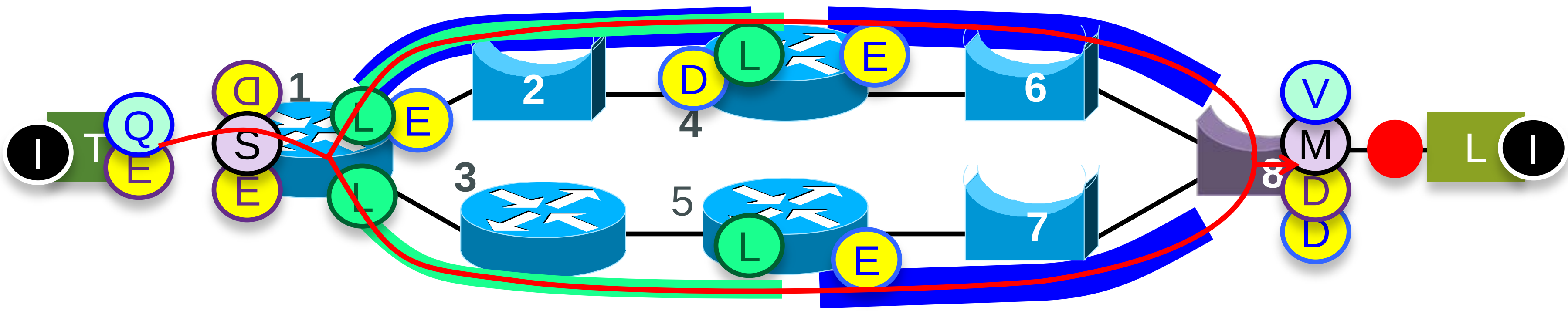
circuit_ID psw 28

sequence_number

ET: IP
IPgram

- The Sequencing Discard function (V) then deletes the redundant frames, passing whichever (from Router 4, in this case) happens to arrive, first.

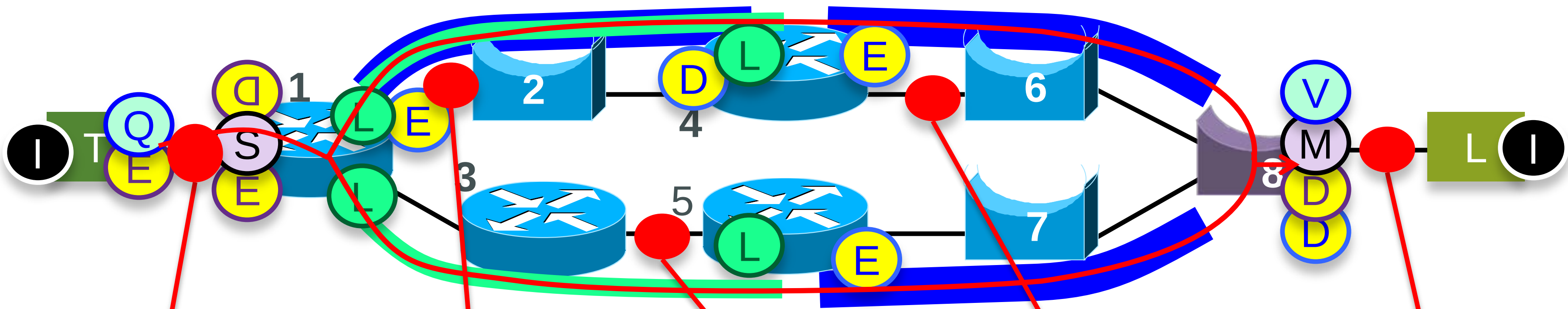
Mixed L2/L3: **IPgram pseudowire**



DA: L
SA: Router 4
VLAN tag 80
ET: IP
IPgram

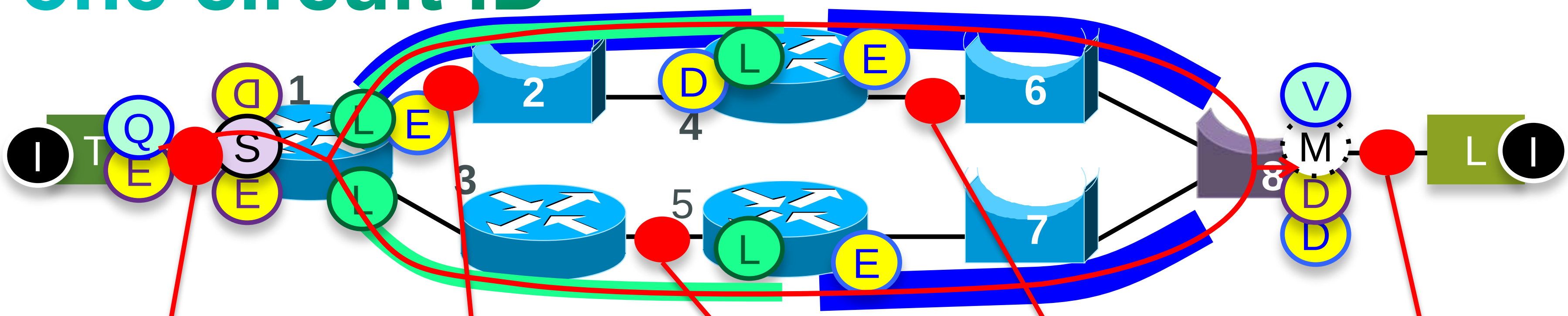
- When the frame is put on the wire to Listener L, the circuit_identifier and sequence_number are discarded, and the remaining parameters make the frame.

SUMMARY: IPgram pseudowire



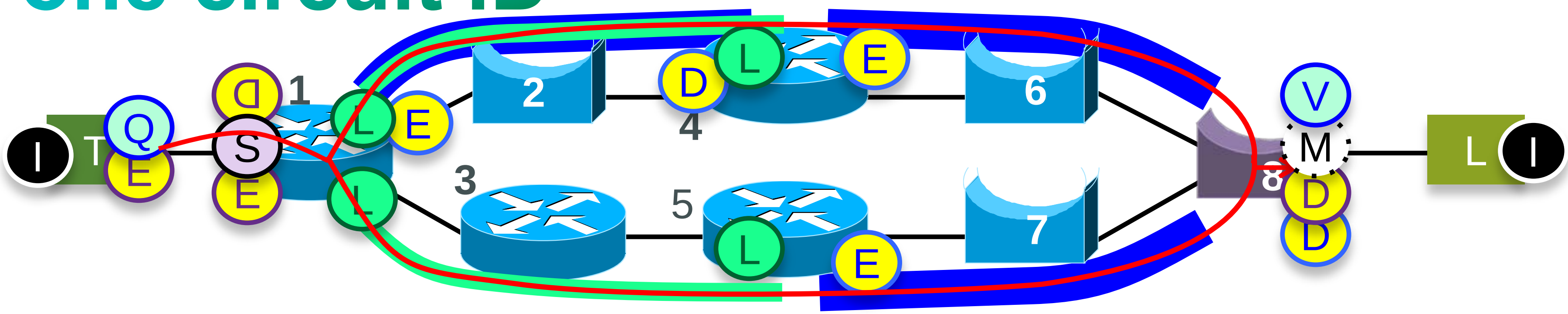
	DA: TSN 140			
	SA: Router 1	DA: Router 5		
DA: Router 1	VLAN tag 309	SA: Router 3	DA: TSN 994	
SA: T	ET: MPLS	ET: MPLS	SA: Router 4	DA: Listener L
ET: MPLS	Tunnel 51	Tunnel 346	VLAN tag 7	SA: Router 4
Pseudowire 28	Pseudowire 449	Pseudowire 31	Pseudowire 419	VLAN tag 80
control (seq)	control (seq)	control (seq)	control (seq)	ET: IP
IPgram	IPgram	IPgram	IPgram	IPgram

Variant 3: End-to-end pseudowire, one circuit ID



	DA: TSN 140			
	SA: Router 1	DA: Router 5		
DA: Router 1	VLAN tag 309	SA: Router 3	DA: TSN 994	
SA: T	ET: MPLS	ET: MPLS	SA: Router 4	DA: Listener L
ET: MPLS	Tunnel 51	Tunnel 346	VLAN tag 7	SA: Router 4
Pseudowire 28	Pseudowire 28	Pseudowire 28	Pseudowire 28	VLAN tag 80
control (seq)	control (seq)	control (seq)	control (seq)	ET: IP
IPgram	IPgram	IPgram	IPgram	IPgram

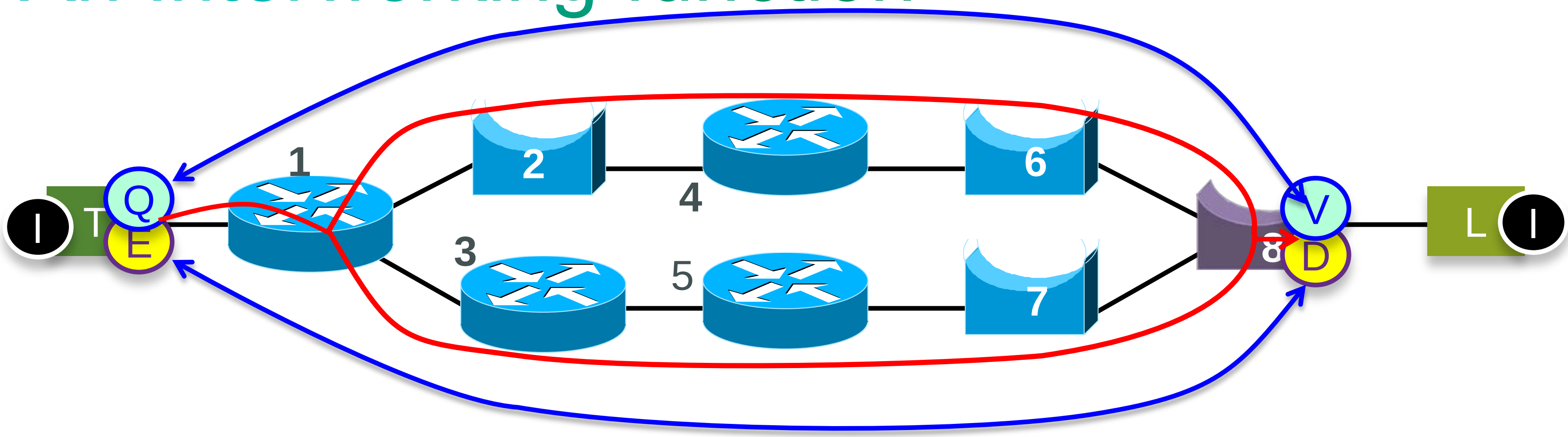
Variant 3: End-to-end pseudowire, one circuit ID



- Note that the Split function (S) is still present, in this case, because pseudowire duplication is not a function that is built into the data plane. It does not create new pseudowire labels, though.
- But, the Merge function (M) is now a no-op.

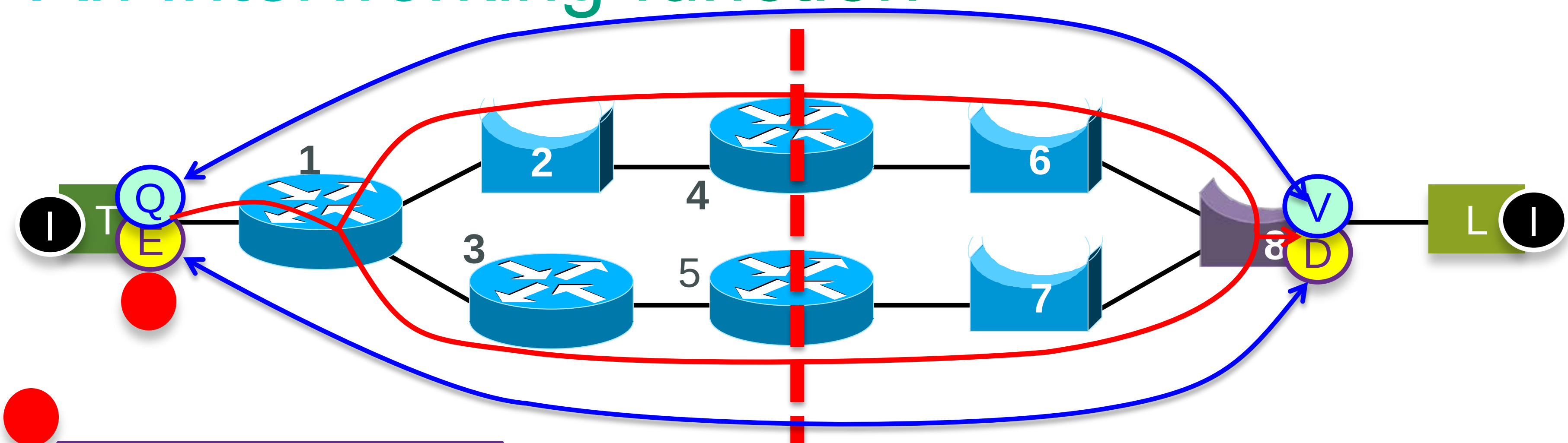
Case 3:
IPgram Pseudowire / Sequenced TSN stitching

An Interworking function



- Ignoring the lower layers, for a moment, we have the Q E pair in Talker T **peering** with the D V pair in Bridge 8.

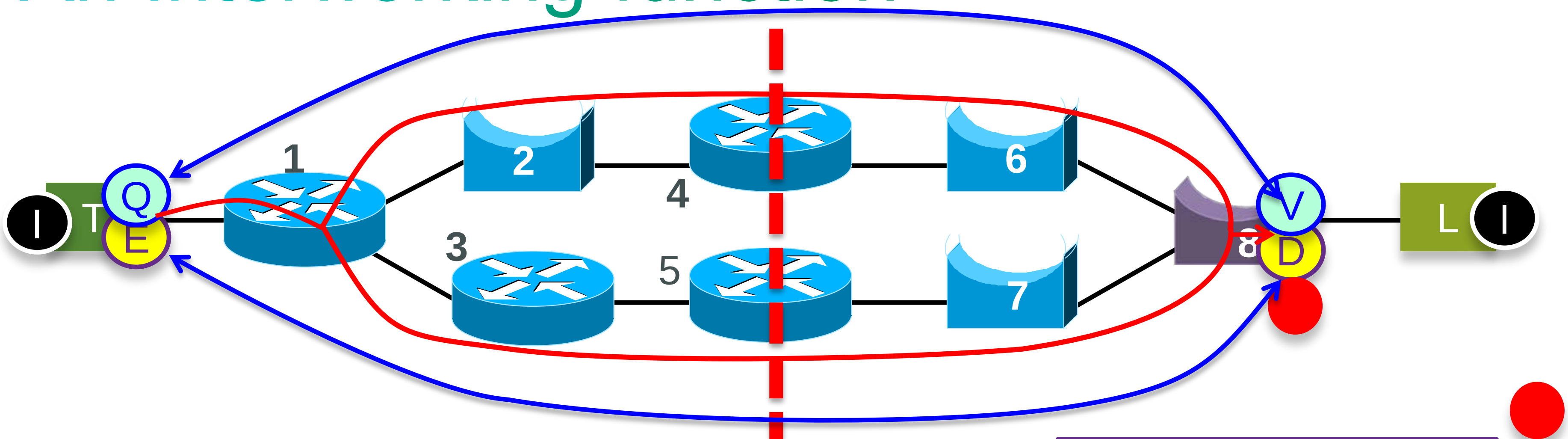
An Interworking function



pseudowire label 419
control (sequence)
IPgram

- In the left-hand world, we want the Circuit ID Encaps/Decaps to be an **IPgram pseudowire**, because it is the "natural" format for a router.

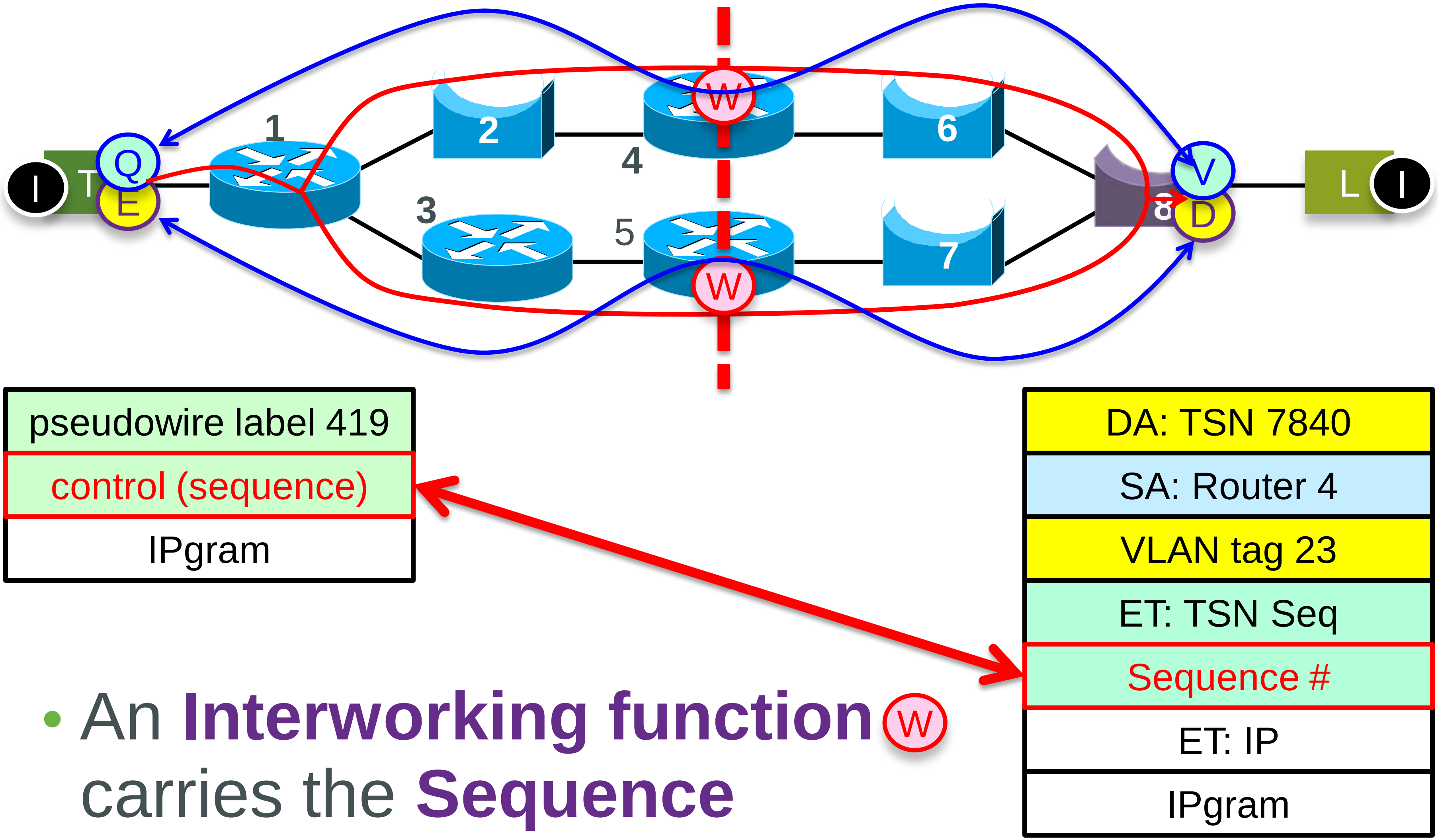
An Interworking function



- In the right-hand world, we want the Circuit ID Encaps/Decaps to be the **Serialized TSN encaps**, because it is the “natural” format for a Bridge.

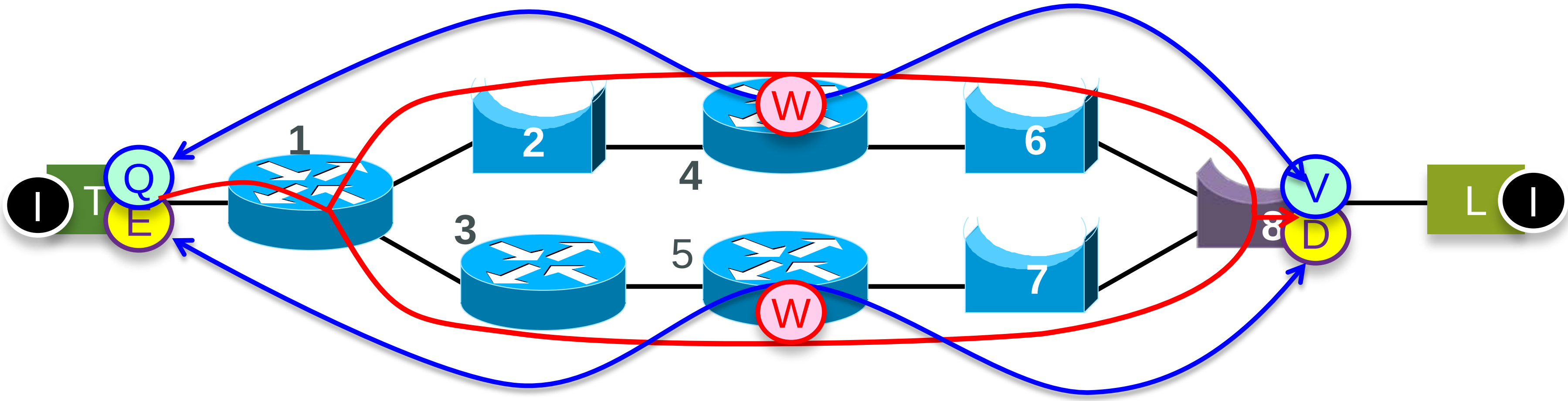
DA: TSN 7840
SA: Router 4
VLAN tag 23
ET: TSN Seq
Sequence #
ET: IP
IPgram

An Interworking function



- An **Interworking function** (W) carries the **Sequence number** across the gap.

An Interworking function



- The Interworking Functions (W) enable the **TSN Pseudowire Encaps** function (E) and the **Sequenced TSN Decaps** function (D) at the very ends of the network to be **peers**, just like the Sequenced TSN and IPgram pseudowire end-to-end cases.

An Interworking function

Talker side

Higher layers
Pseudowire Circuit Encaps/Decaps

Listener side

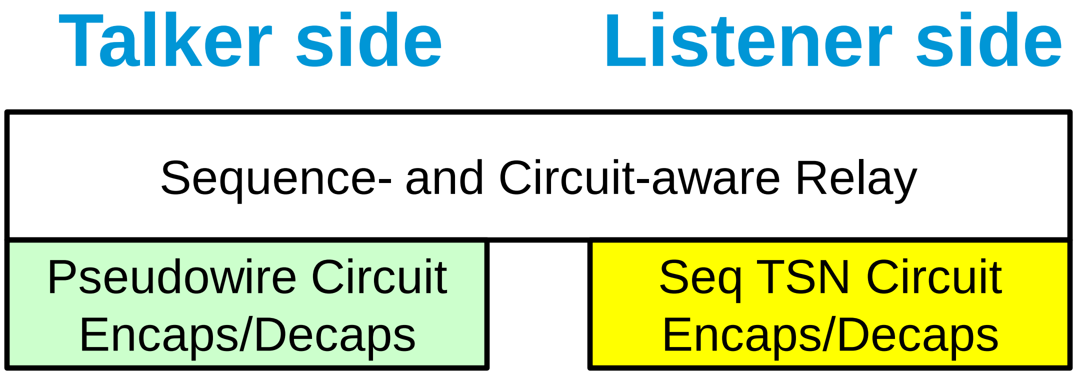
Higher layers
Seq TSN Circuit Encaps/Decaps

- We have two differet protocol stacks, **pseudowire** and **sequenced TSN**, that perform essentially the same function.
- We want them to peer with each other.

An Interworking function

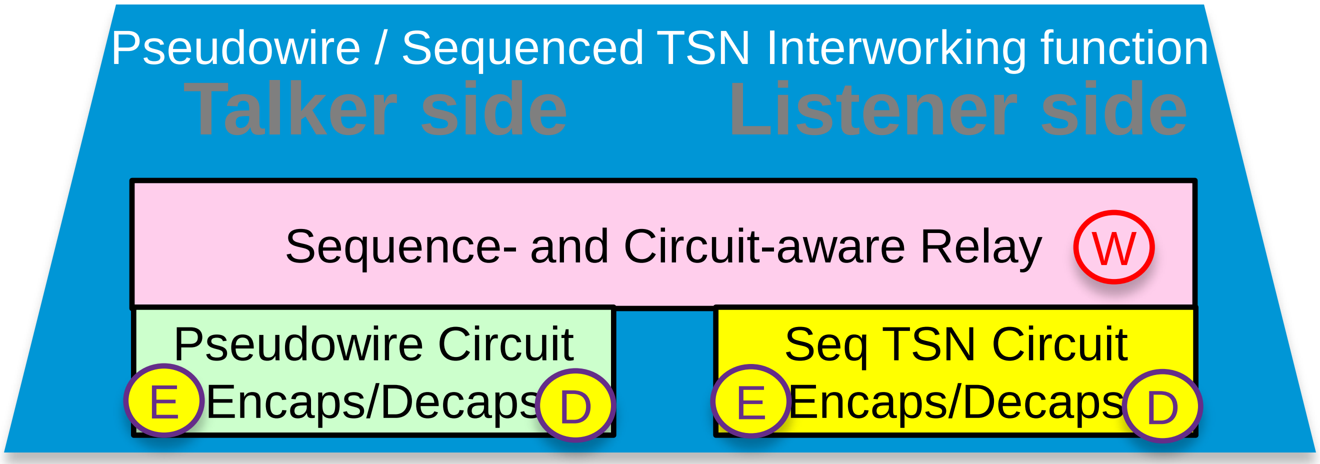
- Note that we are **not** including the sublayers that **act** on the sequence numbers.

An Interworking function



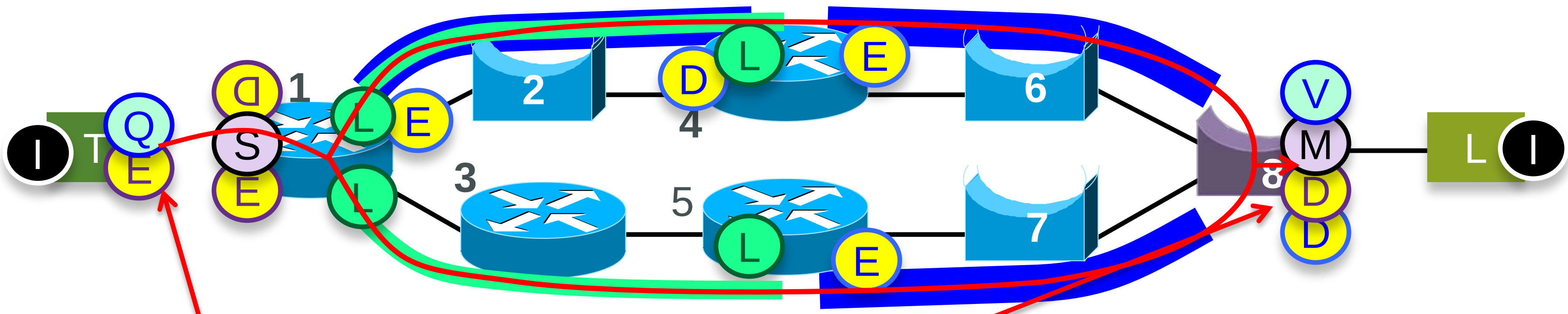
- If we connect these two stacks with a trivial two-port relay that carries the sequence_number and circuit_identifier parameters intact, . . .

An Interworking function



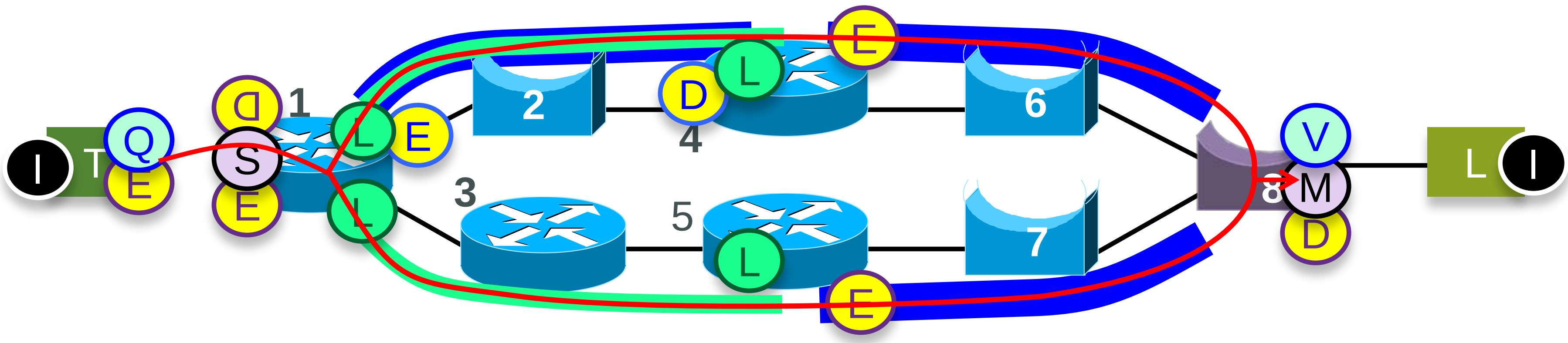
- . . . we have an “interworking function.”
- We’ll show the interworking relay as W, and the whole interworking function as DWE.

Mixed L2/L3 **IP Pseudo / TSN Stitching**



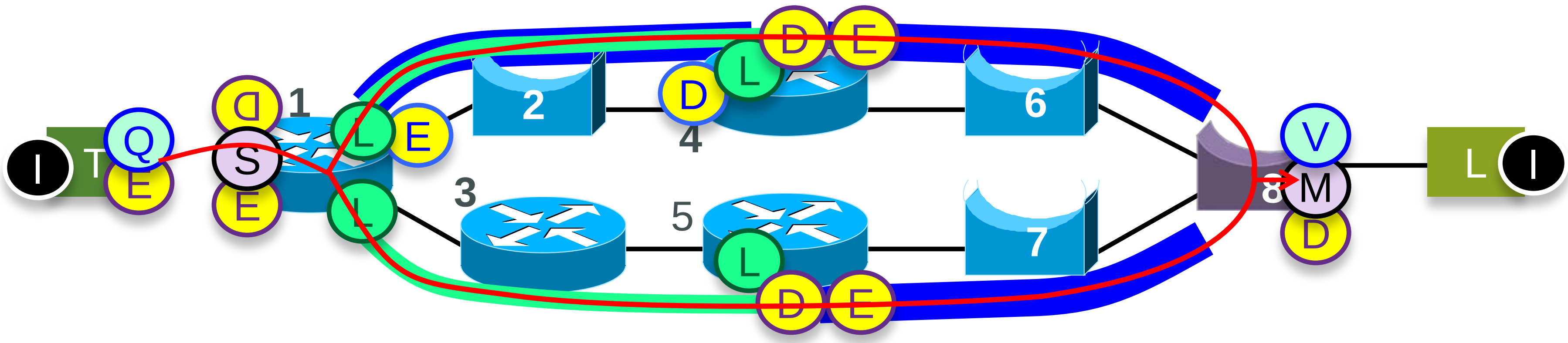
- Returning to our previous discussion, we were peering IPgram pseudowire encapsulations at both ends.

Mixed L2/L3 **IP Pseudo / TSN Stitching**



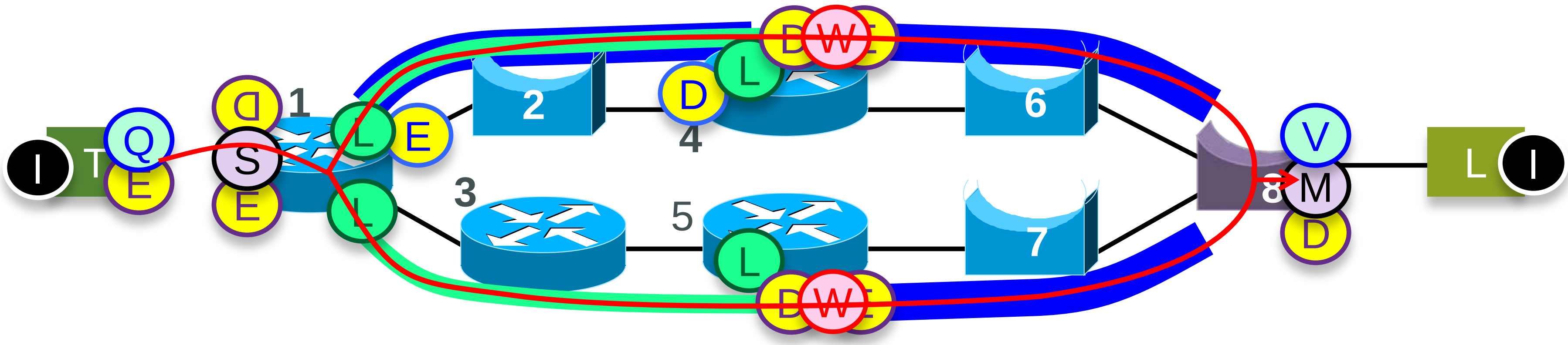
- On the right hand side, we eliminate the simple TSN encaps (E)(D) used to carry the pseudowire, and replace the IPgram pseudowire encaps with the Sequenced TSN enaps that we want.

Mixed L2/L3 **IP Pseudo / TSN Stitching**



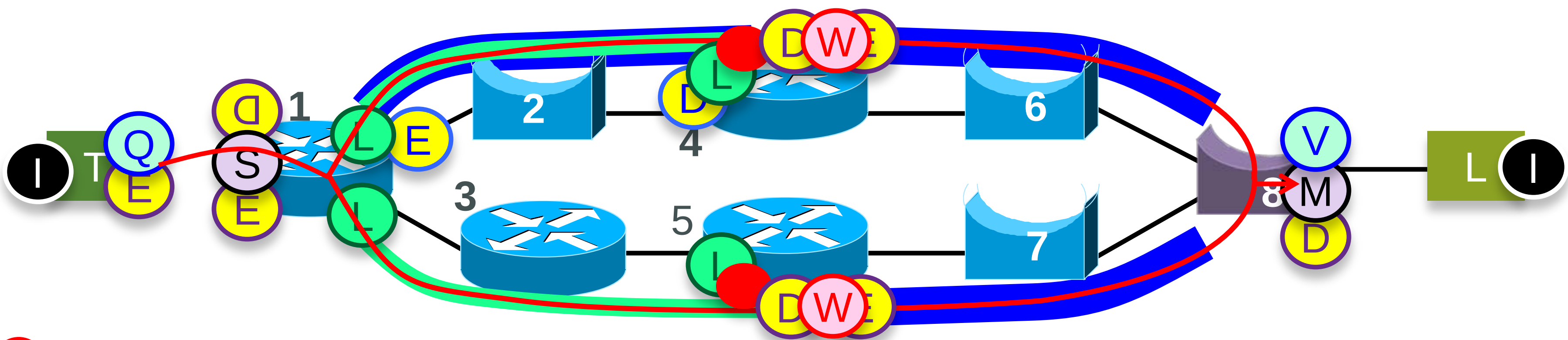
- On the left hand side, we supply termination (D) for the IPgram pseudowire encaps (E) used by Talker T.

Mixed L2/L3 **IP Pseudo / TSN Stitching**



- When we add the interworking relay W ,the IPgram pseudowire / Sequenced TSN Stitching Interworking Function DWE cements the gap.

Mixed L2/L3 **IP Pseudo / TSN Stitching**

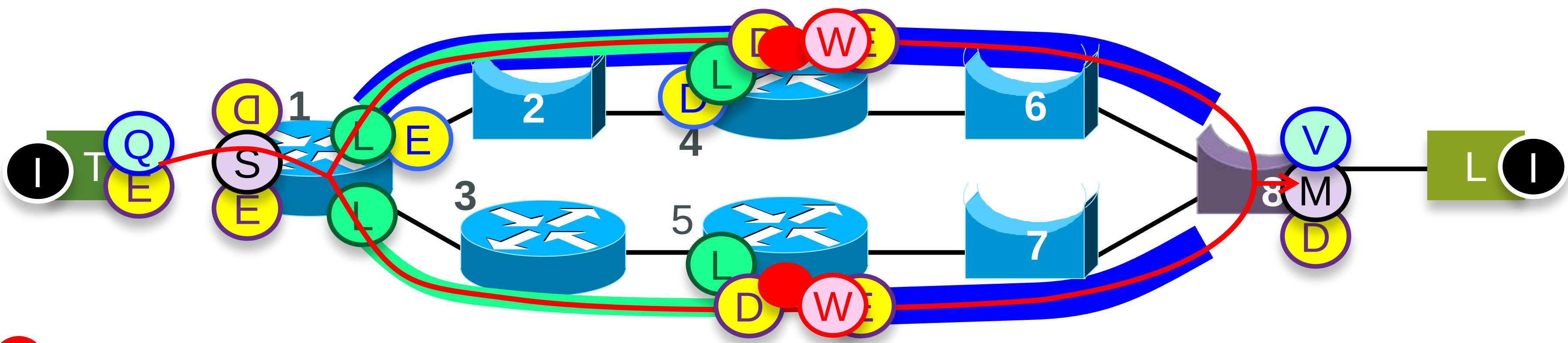


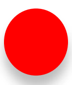
pseudowire label 419
control (sequence)
IPgram

pseudowire label 31
control (sequence)
IPgram

- At this point in the end-to-end IPgram pseudowire description, we had the “naked” pseudowire packets in Routers 4 and 5.

Mixed L2/L3 **IP Pseudo / TSN Stitching**

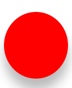




circuit_ID 419/7840[23]

sequence_number

IPgram



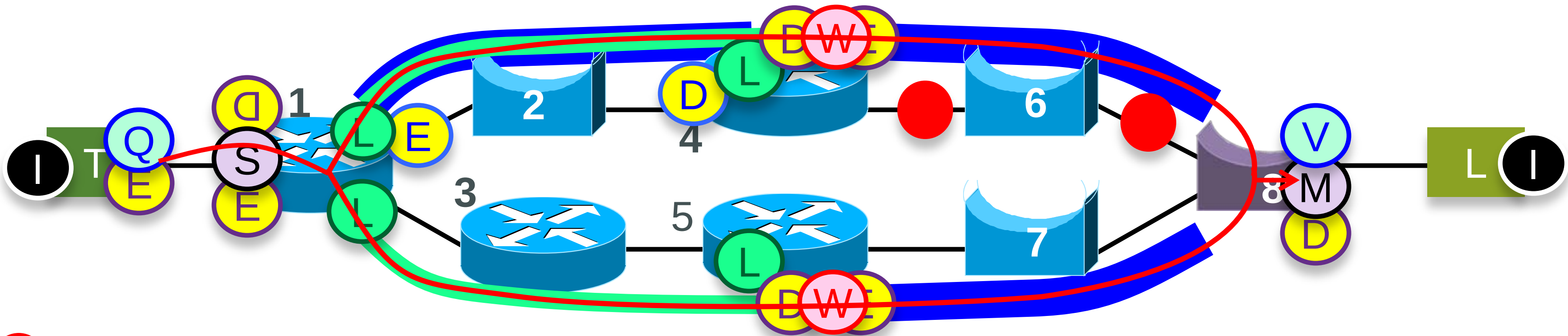
circuit_ID 31/12[50]

sequence_number

IPgram

- These packets are decapsulated.

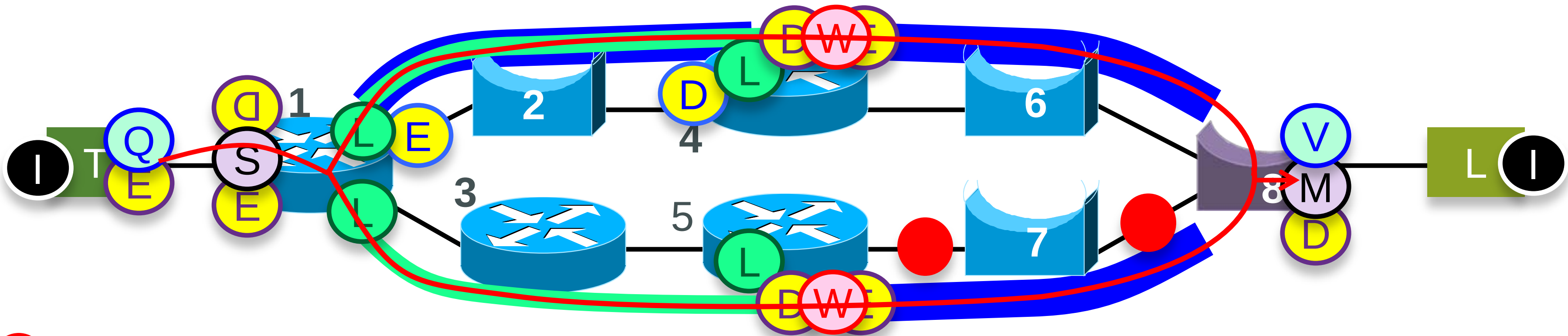
Mixed L2/L3 **IP Pseudo / TSN Stitching**



DA: TSN 7840
SA: Router 4
VLAN tag 23
ET: TSN Seq
Sequence #
ET: IP
IPgram

- And when re-encapsulated by the Sequenced TSN Encaps (E), the packet in Router 4 becomes 7840[23].

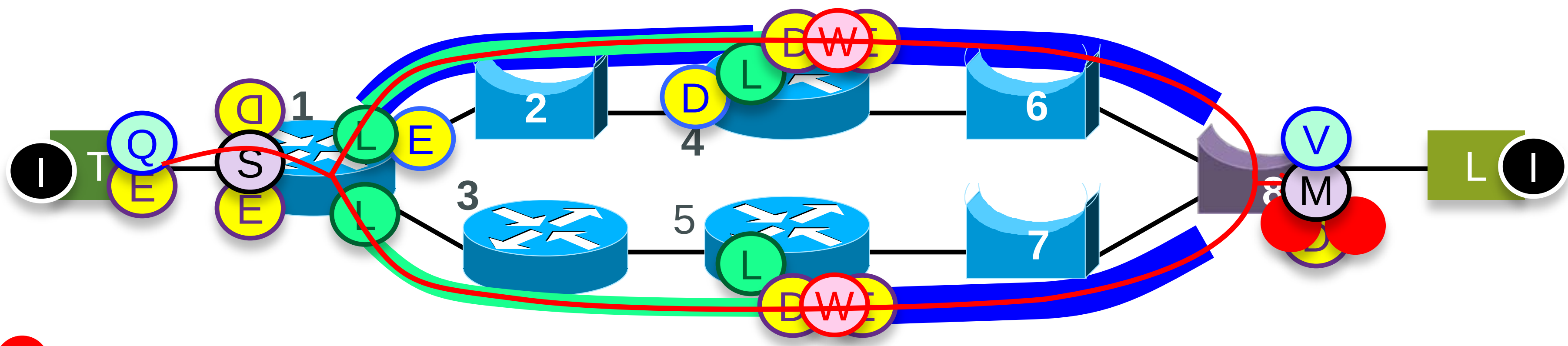
Mixed L2/L3 **IP Pseudo / TSN Stitching**



DA: TSN 12
SA: Router 5
VLAN tag 50
ET: TSN Seq
Sequence #
ET: IP
IPgram

- And IPgram pseudowire label 346 is translated by Router 5’s Interworking function D W E into TSN circuit 12[50].

Mixed L2/L3 **IP Pseudo / TSN Stitching**



DA: Listener L

SA: Router 4 or 5

vlan_identifier 80

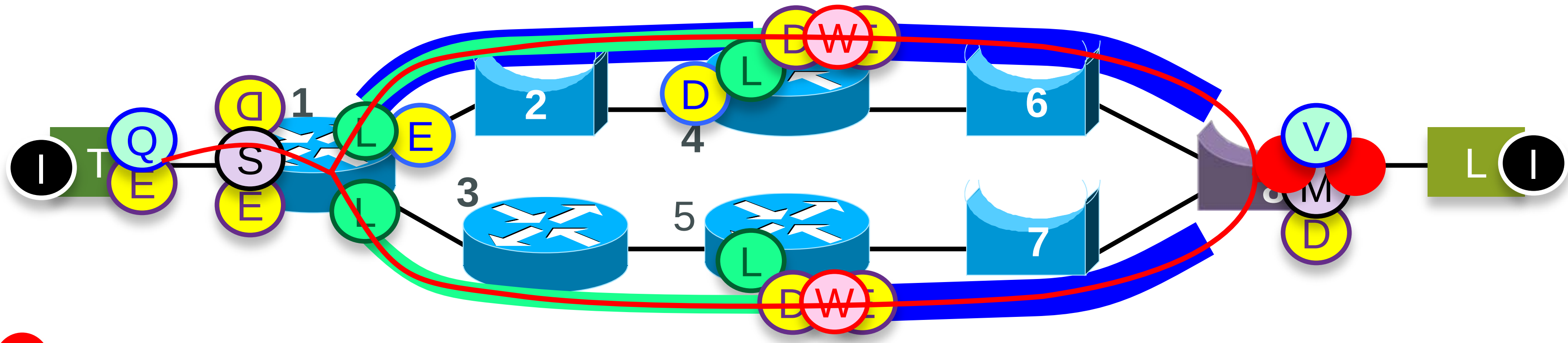
circuit_ID 7840[23] or 12[50]

sequence_number

ET: IP
IPgram

- The TSN Decaps function (D) unwraps the circuit_identifier and sequence_number, and restores the Ethernet frame parameters.

Mixed L2/L3 **IP Pseudo / TSN Stitching**



DA: Listener L

SA: Router 4 or 5

vlan_identifier 80

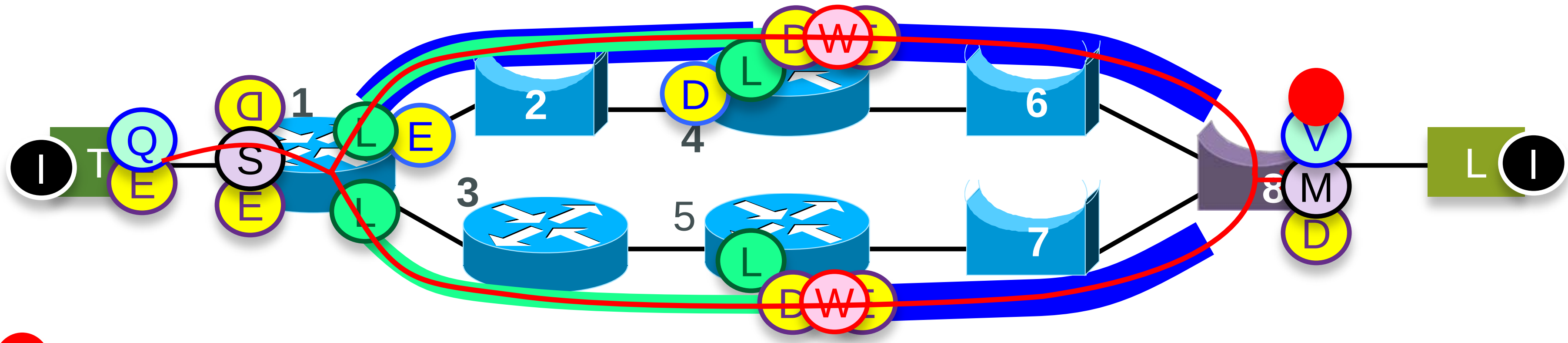
circuit_ID xyz

sequence_number

ET: IP
IPgram

- Output from Merge function (M)
- TSN circuit IDs 7840[23] and 12[50] have been combined, but there are still 2 packets.
- **To Bridge 8**, this is the end-to-end circuit from Talker T.

Mixed L2/L3 **IP Pseudo / TSN Stitching**



DA: Listener L

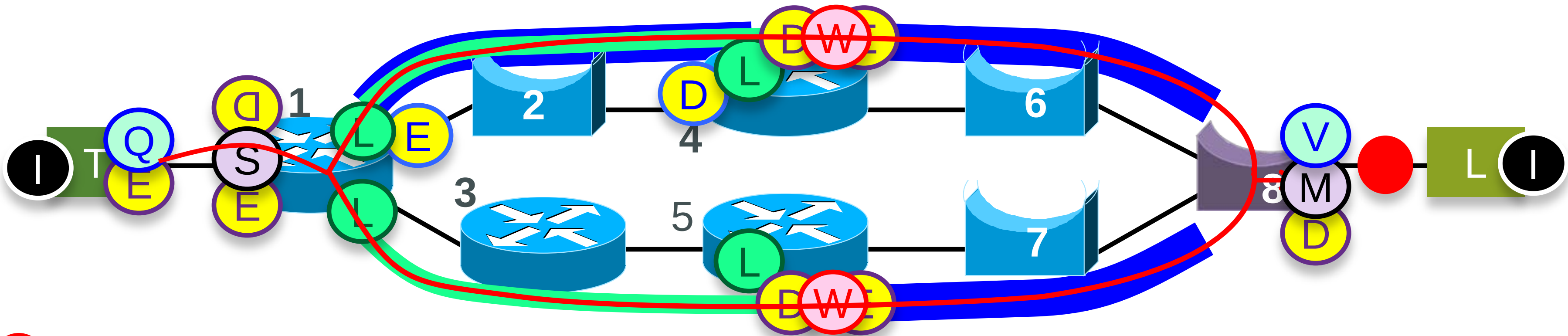
SA: Router 4

vlan_identifier 80

ET: IP
IPgram

- The Sequencing Discard function (V) passes only one of the frames.
- The circuit_identifier and sequence_number are no longer needed.

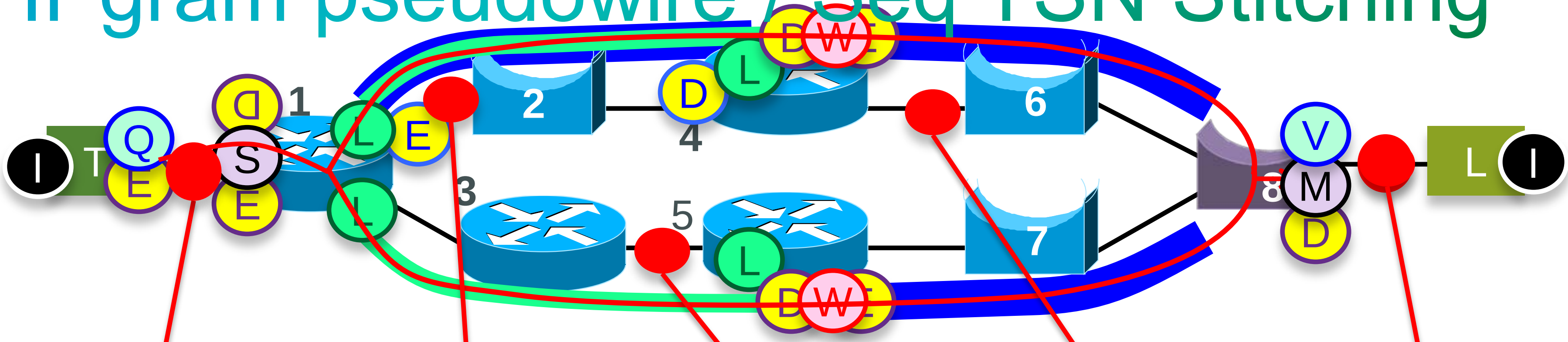
Mixed L2/L3 **IP Pseudo / TSN Stitching**



DA: Listener L
SA: Router 4
ET: IP
IPgram

- After discarding the unused parameters and converting the rest to the appropriate frame, this is what is output to Listener L.

SUMMARY:
IPgram pseudowire / Seq TSN Stitching



DA: Router 1
SA: T
ET: MPLS
Pseudowire 28
control (seq)
IPgram

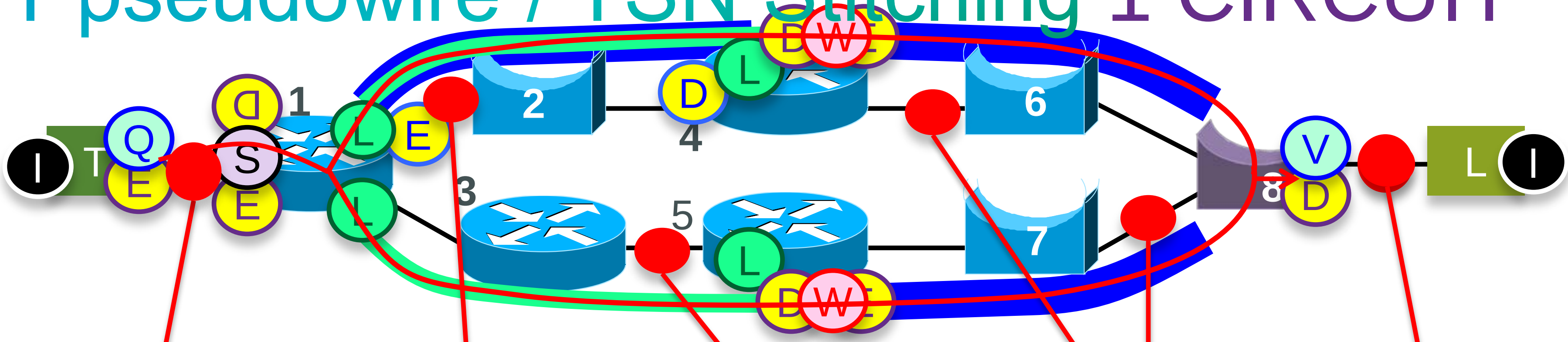
DA: TSN 140
SA: Router 1
VLAN tag 309
ET: MPLS
Tunnel 51
Pseudowire 449
control (seq)
IPgram

DA: Router 5
SA: Router 3
ET: MPLS
Tunnel 346
Pseudowire 31
control (seq)
IPgram

DA: TSN 12
SA: Router 5
VLAN tag 50
ET: TSN Seq
Sequence #
ET: IP
IPgram

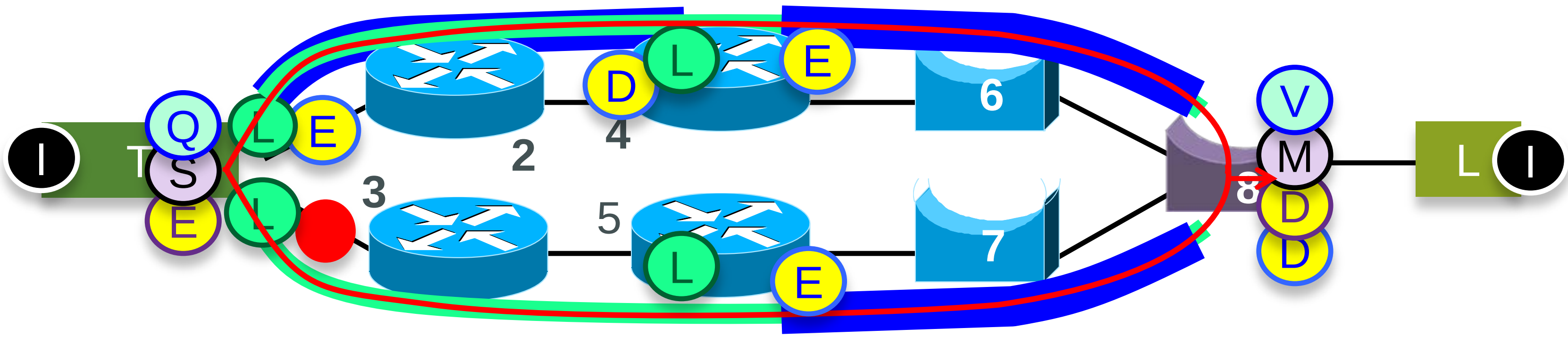
DA: Listener L
SA: Router 4
ET: IP
IPgram

Variant 4:
Ppseudowire / TSN Stitching 1 CIRCUIT



	DA: TSN 140			
	SA: Router 1	DA: Router 5	DA: TSN 12	
DA: Router 1	VLAN tag 309	SA: Router 3	SA: Router 5	
SA: T	ET: MPLS	ET: MPLS	VLAN tag 50	
ET: MPLS	Tunnel 51	Tunnel 346	ET: TSN Seq	DA: Listener L
Pseudowire 28	Pseudowire 28	Pseudowire 28	Sequence #	SA: Router 4
control (seq)	control (seq)	control (seq)	ET: IP	ET: IP
IPgram	IPgram	IPgram	IPgram	IPgram

Variant 5: Dual-homed Talker

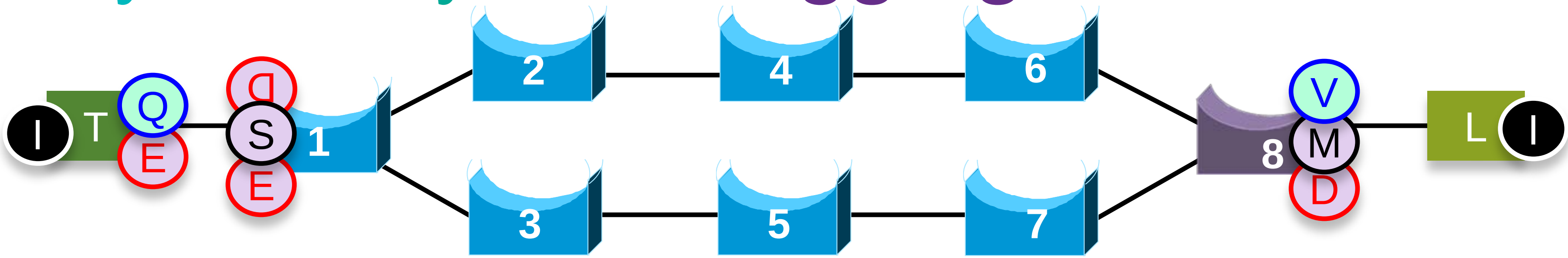


pseudowire label 28
control (sequence)
IPgram

- Talker T could be dual-homed.
- In this case, clearly T must supply the sequence numbers.
- The sequence numbers are usually part of the encapsulation.
- So, T terminates the pseudowire, not routers 2 and 3.

Case 4:
Layer 2 end-to-end HSR or PRP tagging

Layer 2 only: **HSR tagging**

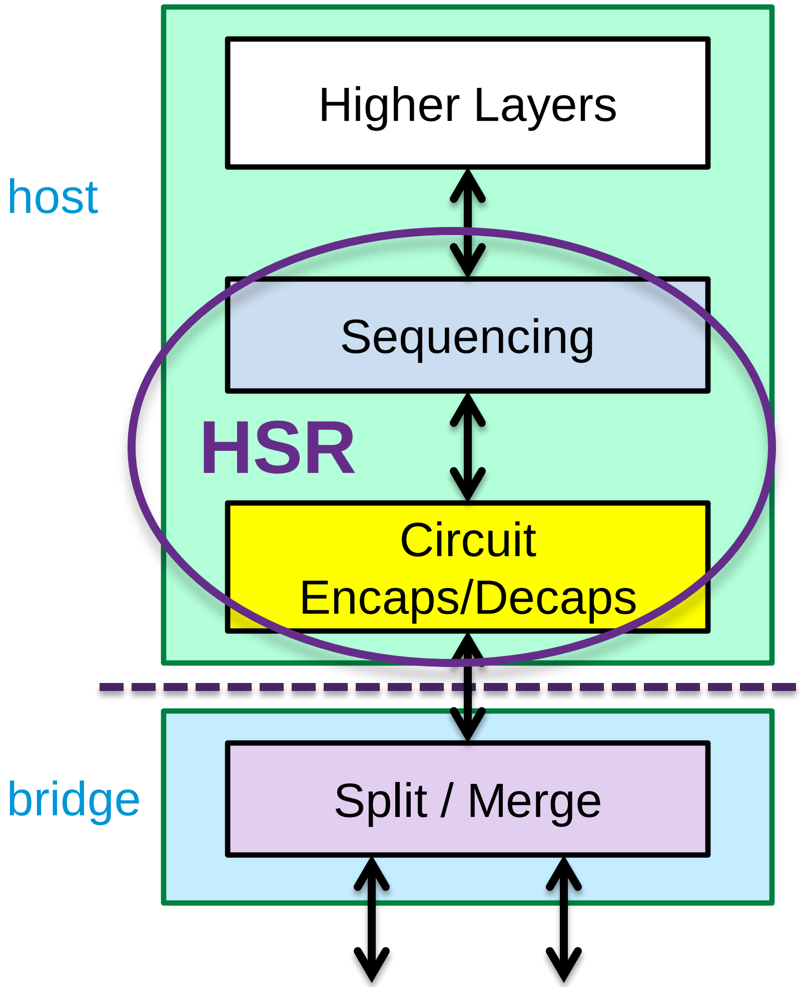


- Again, Talker is TSN-aware, Listener is not.
- This time, Talker is **not** VLAN-aware, Listener **is** VLAN-aware.
- In this case, HSR and TSN Encaps E and Decaps D are combined into a single layer.

HSR-like, not HSR

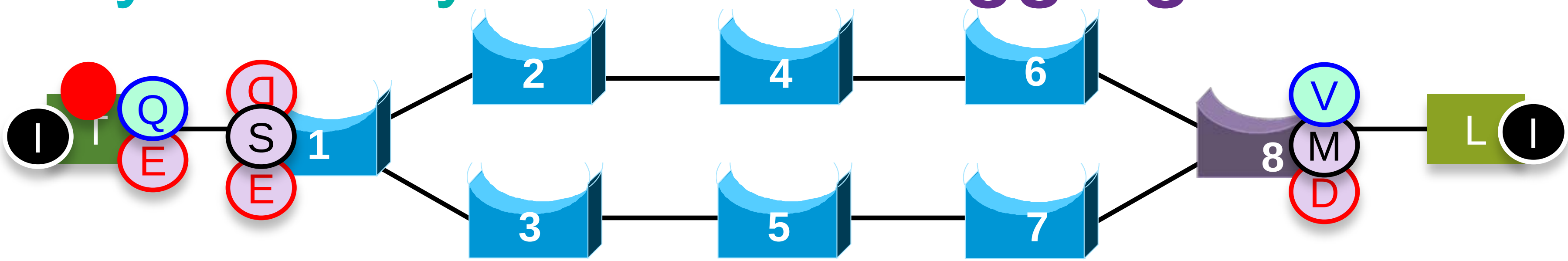
- This is not HSR. It is the HSR format used for a different purpose. This idea may or may not sit well with IEC TC65X.
- This “HSR-like” layer:
  - Connects to a single port, not two.
  - May use one sequence number variable per circuit, not one per host. (This is debatable.)
  - If the station is VLAN aware, has the VLAN tagging below (outside) the HSR sublayer.

HSR Layering



- Note that this is the layering – the top box is Talker T, and the bottom box is Bridge 1.
- HSR combines the Circuit Encaps/Decaps and Sequencing functions.
- It also encapsulates the destination MAC address which, as we will see, is not really very useful.

Layer 2 only: **HSR-like tagging**



DA: L

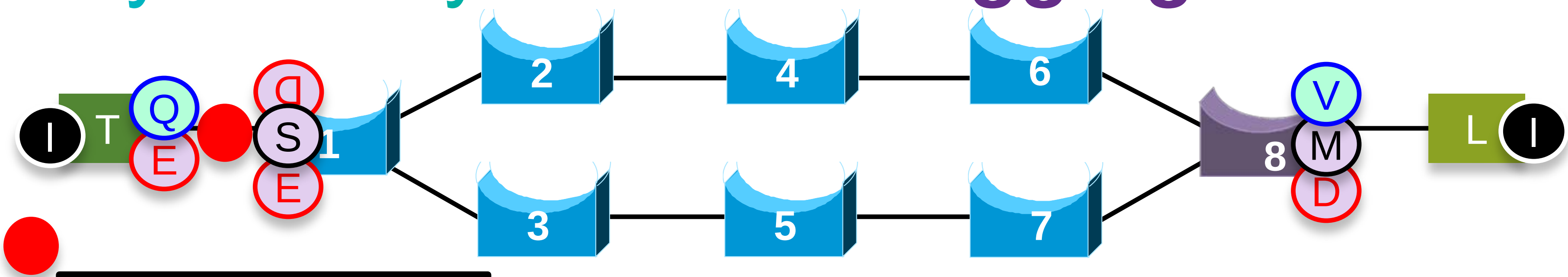
SA: T

circuit_identifier

ET: IP
IPgram

- Talker's stack is not VLAN-aware. This is what the frame is when it hits the TSN Encaps layer.
- Note that Bridge 1 would normally add a **VLAN 80 tag** to this frame.

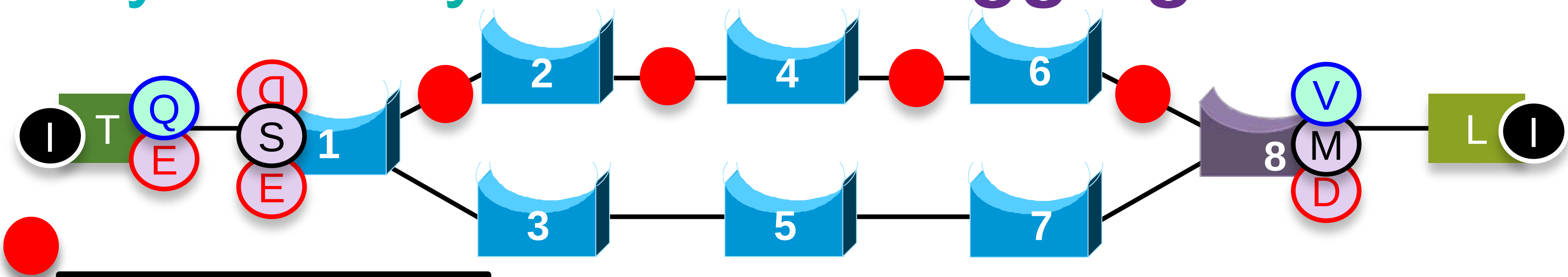
Layer 2 only: **HSR-like tagging**



DA: TSN 734
SA: T
VLAN tag 99
HSR EtherType
pid, length, sequence
DA: L
EtherType
Data

- The Sequencing (Q) and combined HSR/TSN Encaps layer (E) create a sequence number and add a TSN/HSR tag.

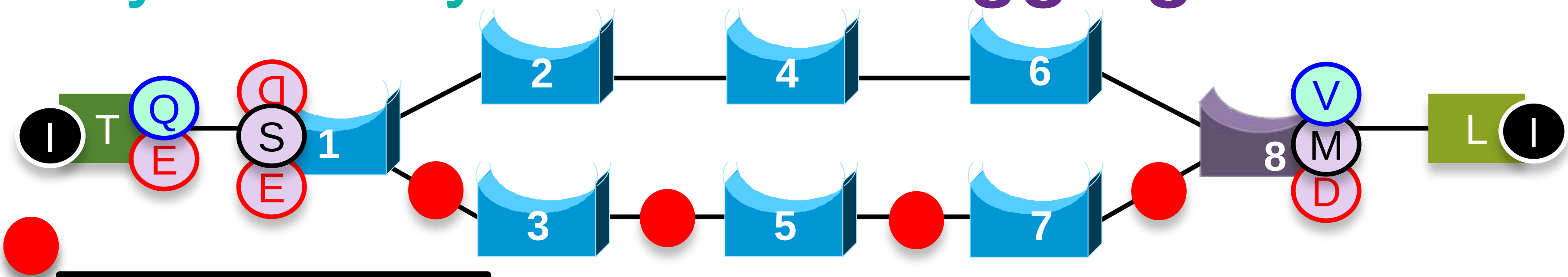
Layer 2 only: **HSR-like tagging**



DA: TSN 7840
SA: T
VLAN tag 23
HSR EtherType
pid, length, sequence
DA: L
EtherType
Data

- The Split function (S) operates on the TSN header, for the path ID, and the HSR header, for the sequence number.
  - (The “pid” field includes a “path A / path B” flag that intended to be different between the two paths. We may or may not follow that usage.)

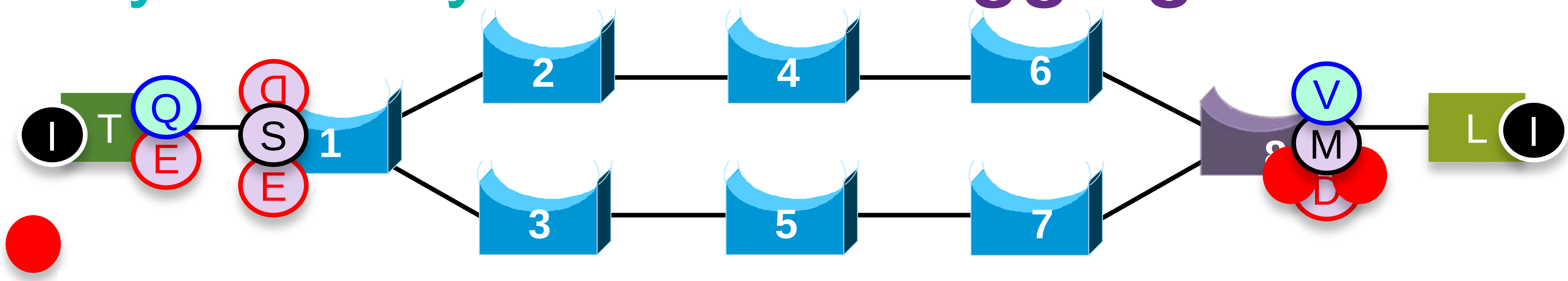
Layer 2 only: **HSR-like tagging**



DA: TSN 12
SA: T
VLAN tag 50
HSR EtherType
pid, length, sequence
DA: L
EtherType
Data

- The other path gets a different DA and VLAN tag.
- Note that the Split function split TSN 734[99] into TSN 7840[23] and 12[50].

Layer 2 only: **HSR-like tagging**



DA: Listener L

SA: Talker T

vlan_identifier 80

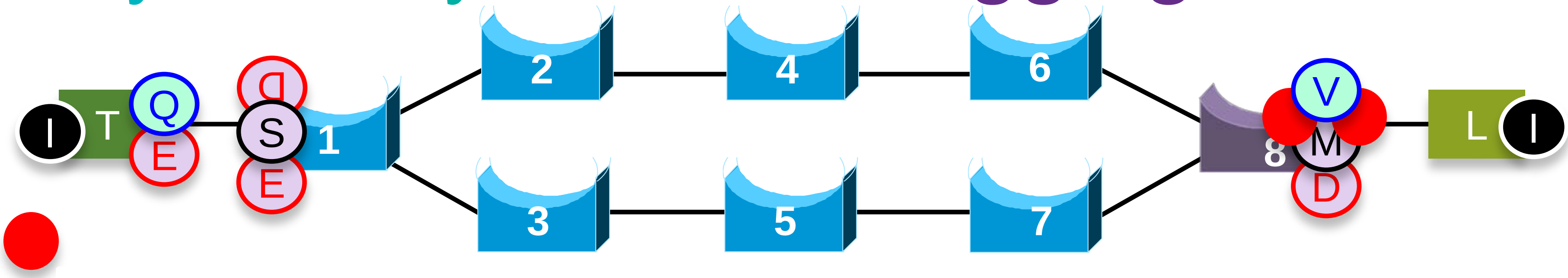
circuit_ID 7840[23] or 12[50]

sequence_number

EtherType
Data

- The Merge function (M) operates on the circuit_identifier exposed by the decapsulation function

Layer 2 only: **HSR-like tagging**



DA: Listener L

SA: Talker T

vlan_identifier 80

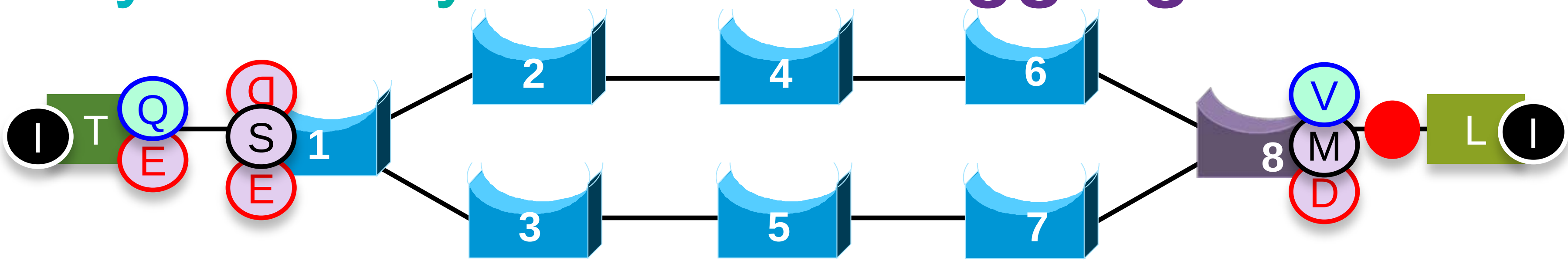
circuit_ID 734[99]

sequence_number

EtherType
Data

- Output from Merge function is the original 734[99] tunnel that originated from Bridge 1.
- Two packets are present until the Sequencing Discard function V discards one.

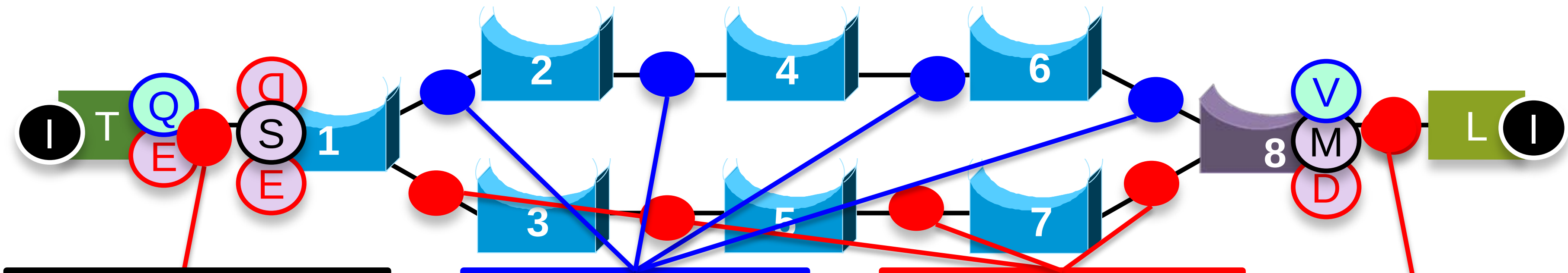
Layer 2 only: **HSR-like tagging**



DA: L
SA: T
VLAN tag 80
ET: IP
IPgram

- And this is delivered on the wire.

Summary: HSR-like tagging



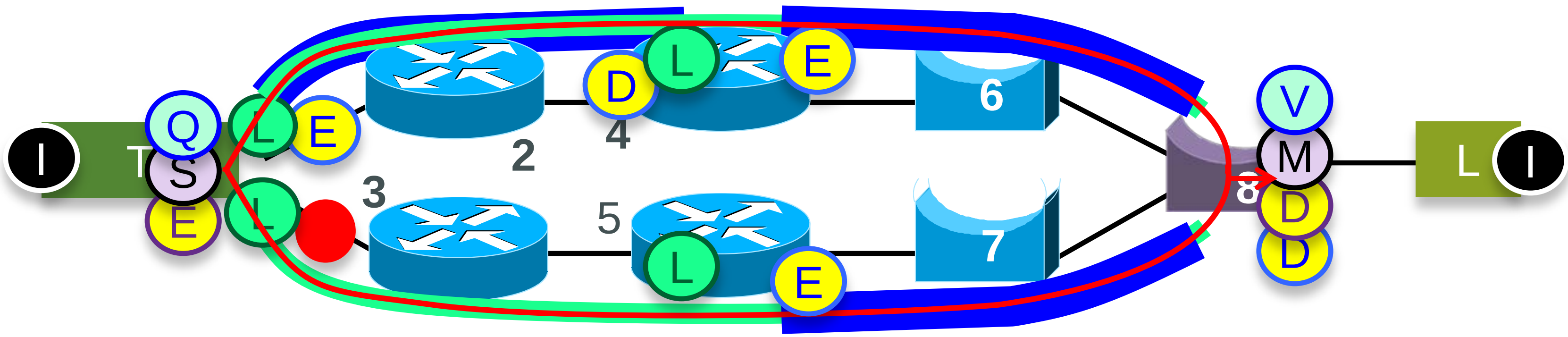
DA: TSN 734
SA: T
VLAN tag 99
HSR EtherType
pid, length, seq
DA: L
EtherType
Data

DA: TSN 7840
SA: T
VLAN tag 23
HSR EtherType
pid, length, seq
DA: L
EtherType
Data

DA: TSN 12
SA: T
VLAN tag 50
HSR EtherType
pid, length, seq
DA: L
EtherType
Data

DA: L
SA: T
VLAN tag 80
ET: IP
IPgram

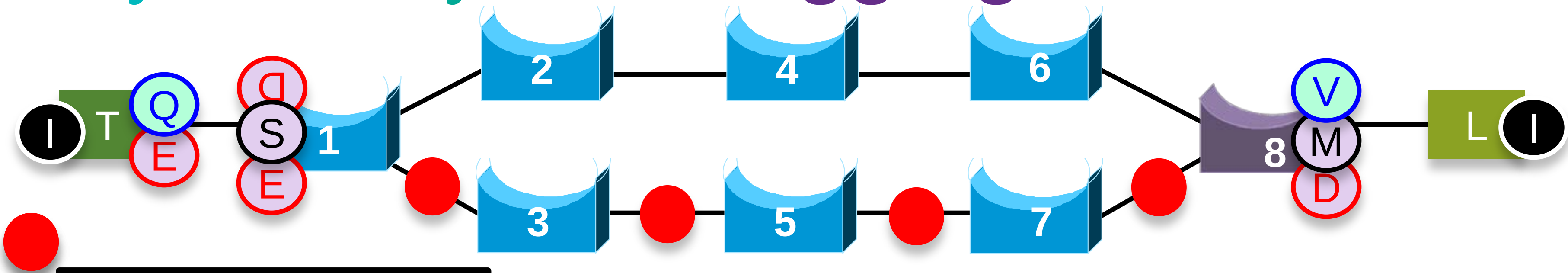
Variant 5: Dual-homed Talker



pseudowire label 28
control (sequence)
IPgram

- Talker T could be dual-homed.
- In this case, clearly T must supply the sequence numbers.
- The sequence numbers are usually part of the encapsulation.
- So, T terminates the pseudowire, not routers 2 and 3.

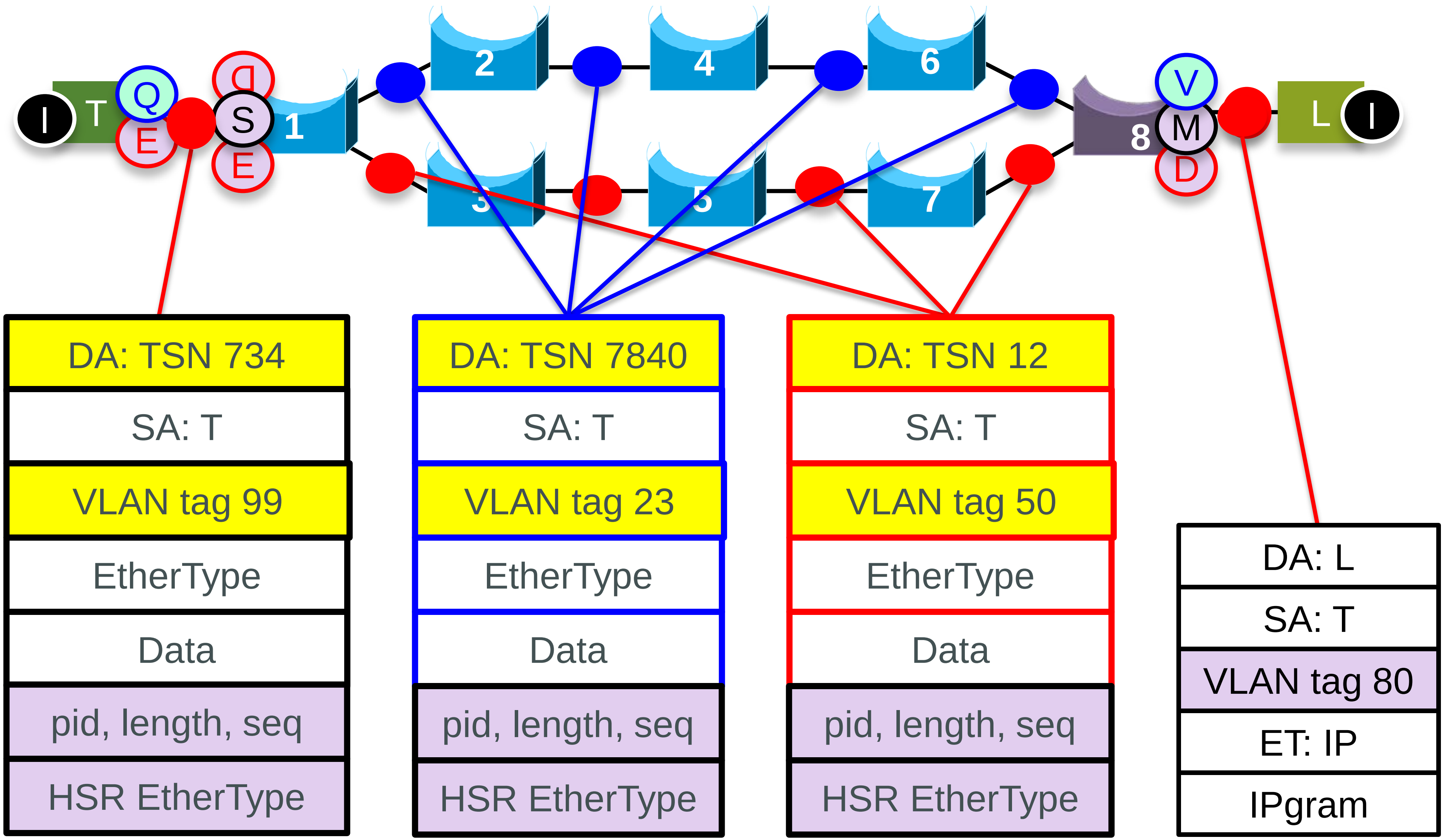
Layer 2 only: **PRP tagging**



DA: TSN 12
SA: T
VLAN tag 50
EtherType
Data
pid, length, sequence
HSR EtherType

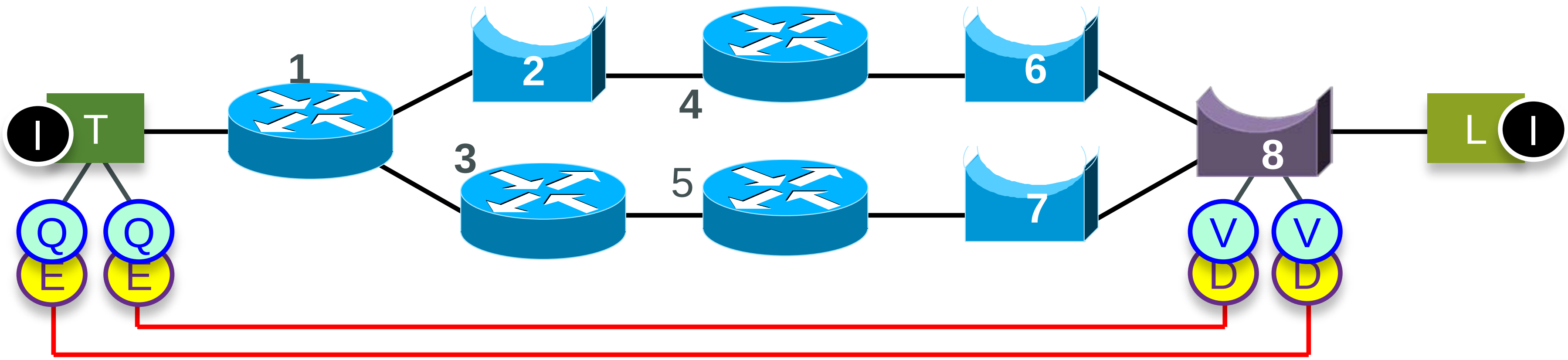
- PRP would work similarly.
- This could be useful to interoperate with existing deployments.
- **A big issue with the PRP trailer is that you can't tell what it's position is in the tag layering.**

Summary: PRP tagging



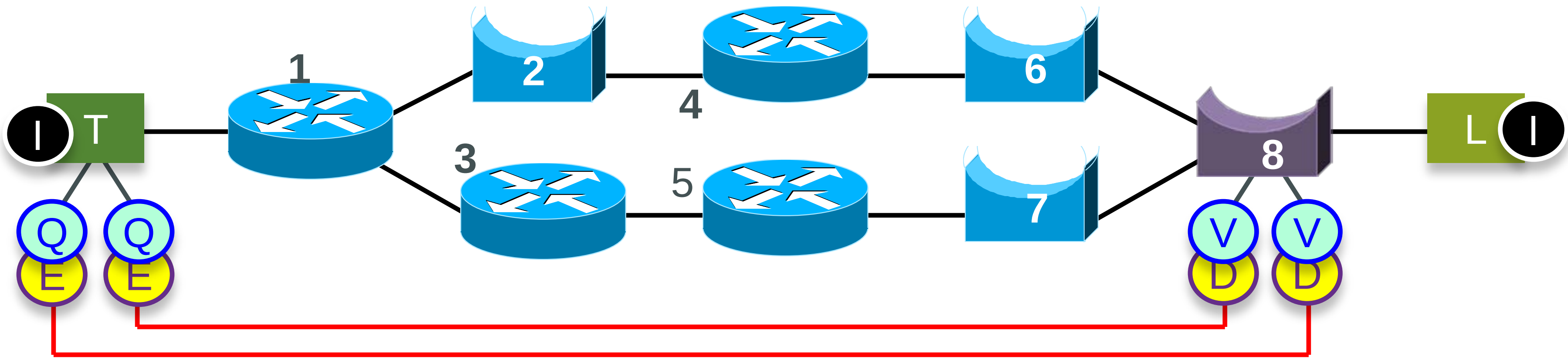
Case 5:
Layer 2 end-to-end Ethernet encapsulation

Ethernet tunneling



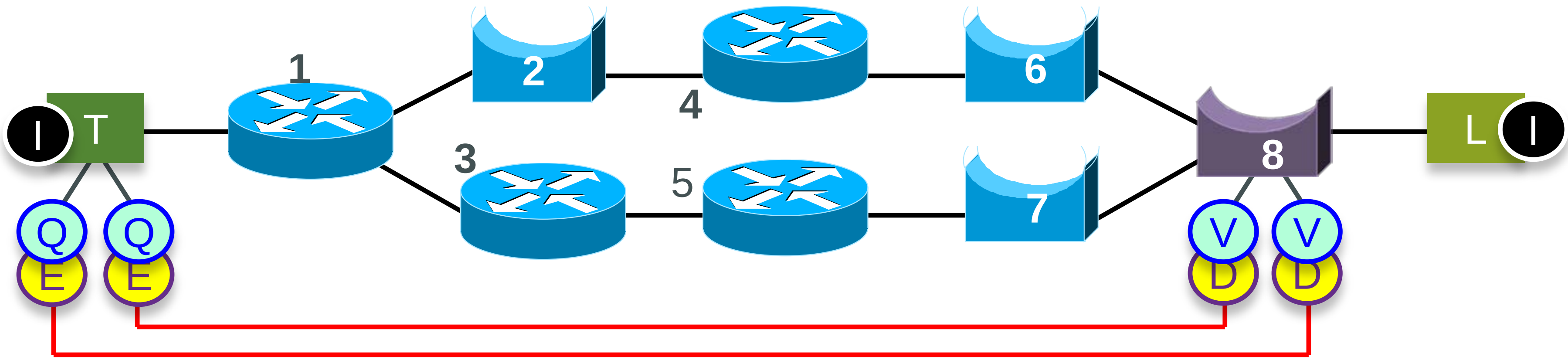
- You can always create end-to-end encapsulated Ethernet circuits using, for example, Ethernet pseudowire and/or PBB-TE MAC-in-MAC in the standard ways it's done, today.

Ethernet tunneling



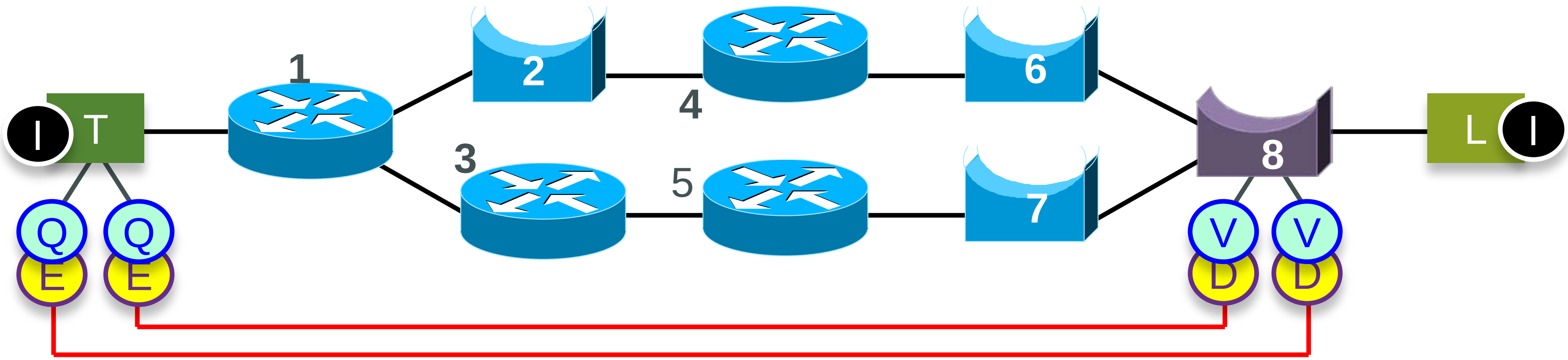
- The catch is that the Talker has a separate Ethernet port per TSN flow.
- This is not compatible with a simple IP stack; each port needs its own IP address.

Ethernet tunneling



- There are solutions to this classic “multi-homed IP host” problem.
- It would take some effort to make them compatible with our simple QoS purpose.

Ethernet tunneling



- And, of course, it still begs the question, “How are these packet encapsulated?”
- Again, there are many standard answers to that question.

Case 6:
IP Multicast

IP Multicast

- Assume for a moment that we do not need to transmit multiple copies on different paths, so we do not need the Sequencing functions. Q V
- Then, all we need is a per-flow circuit_identifier on every packet, at both L2 and L3.
- **IP Multicast can supply this**, even if it cannot supply a sequence number.

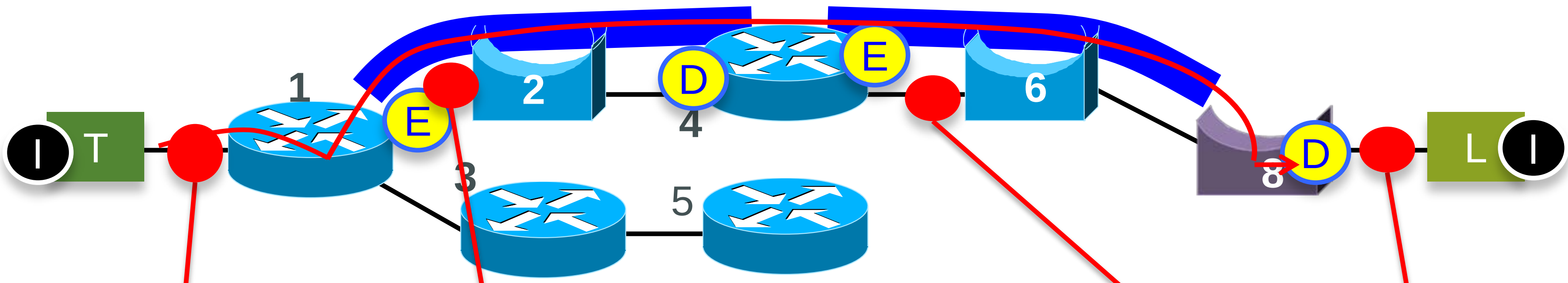
IP Multicast

- In general, an IP multicast flow is identified by the Multicast IP destination address and the (unicast) IP source address.
- There is a 32:1 mapping of IP multicast addresses to MAC Group DAs.

IP Multicast

- If the network administrator and protocols can ensure that the IP multicast addresses are unique over the flows, no TSN encapsulation is necessary.
- Otherwise, the usual TSN encapsulation will solve the Bridges' problems with multicast, and the Routers can easily identify the streams to apply TSN QoS.

IP Multicast



- The normal IP Group address and VLAN may or may not be sufficient for Bridged TSN networks, but the TSN encapsulation fixes this.

DA: Group Z
SA: T
ET: IP
MC IPgram

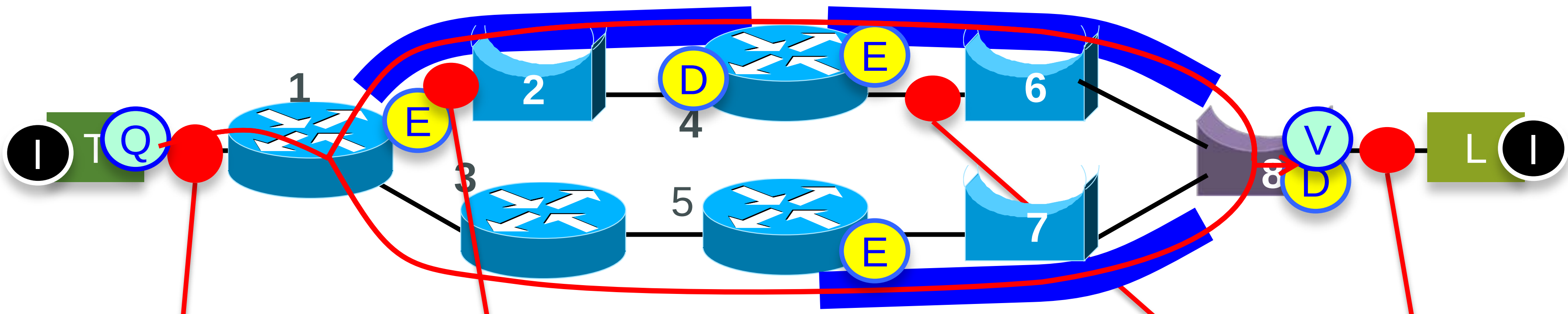
DA: Group Z?
SA: Router 1
VLAN tag 309?
ET: IP
MC IPgram

DA: Group Z?
SA: Router 4
VLAN tag 80
ET: IP
IPgram

IP Multicast

- And, **if** the protocol carried in the IP Multicast packet has a sequence number, then of course, the IP Multicast format is sufficient, in the data plane, for seamless redundancy, as well.

IP Multicast



- The IP multicast control protocols, of course, would need work.

DA: Group Z
SA: T
ET: IP
MC IPgram

DA: Group Z?
SA: Router 1
VLAN tag 309?
ET: IP
MC IPgram

DA: Group Z?
SA: Router 4
VLAN tag 80
ET: IP
IPgram

Summary

Summary

- The layering scheme in tsn-nfinn-L2-Data-Plane-0214-v04 works.
- There are existing protocols for carrying both all-L2 and mixed L2/L3 TSN circuits.
- There are other possibilities for creating TSN circuits: VxLAN, LISP, and dozens of as-yet proprietary schemes.
- A new IEEE 802.1 sequence number tag can handle Ethernet end-to-end seamless redundancy.
- Mixed L2/L3 seamless redundancy requires either:
  - Selecting a single end-to-end L2+ split/merge format (e.g. pseudowire); or
  - An interworking function between L3 and L2 split/merge technologies; or
  - Creating explicit end-to-end Ethernet tunnels.

Thank you.