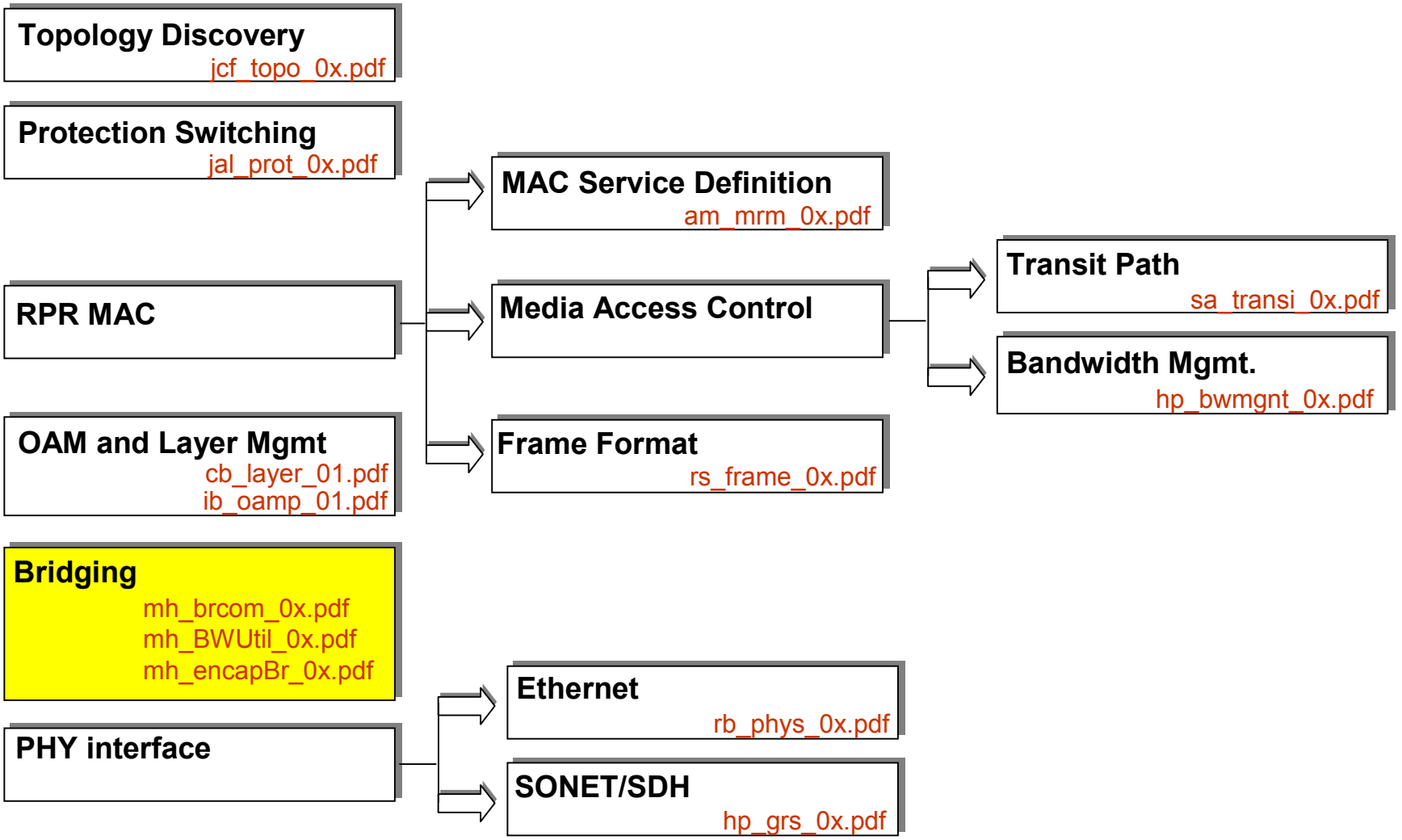
Encapsulated Bridging Support of 802.17

November 2001

Marc Holness, Nortel Networks
Anoop Ghanwani, Lantern Communications
Raj Sharma, Luminous
Robin Olsson, Vitesse
CP Fu, NEC

Components of a Complete RPR Proposal



Objective

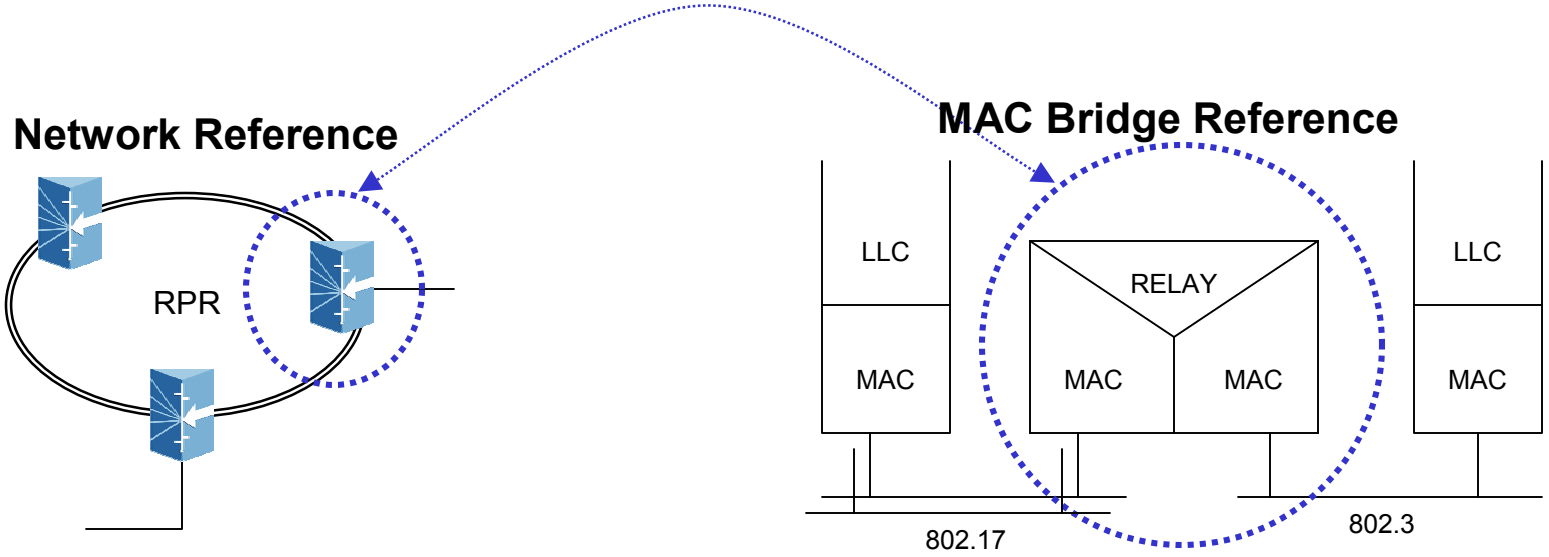
- Proposal of Encapsulated Bridging support for 802.17 MAC
- Solution alignment with 802 and 802.1 Architecture

Outline

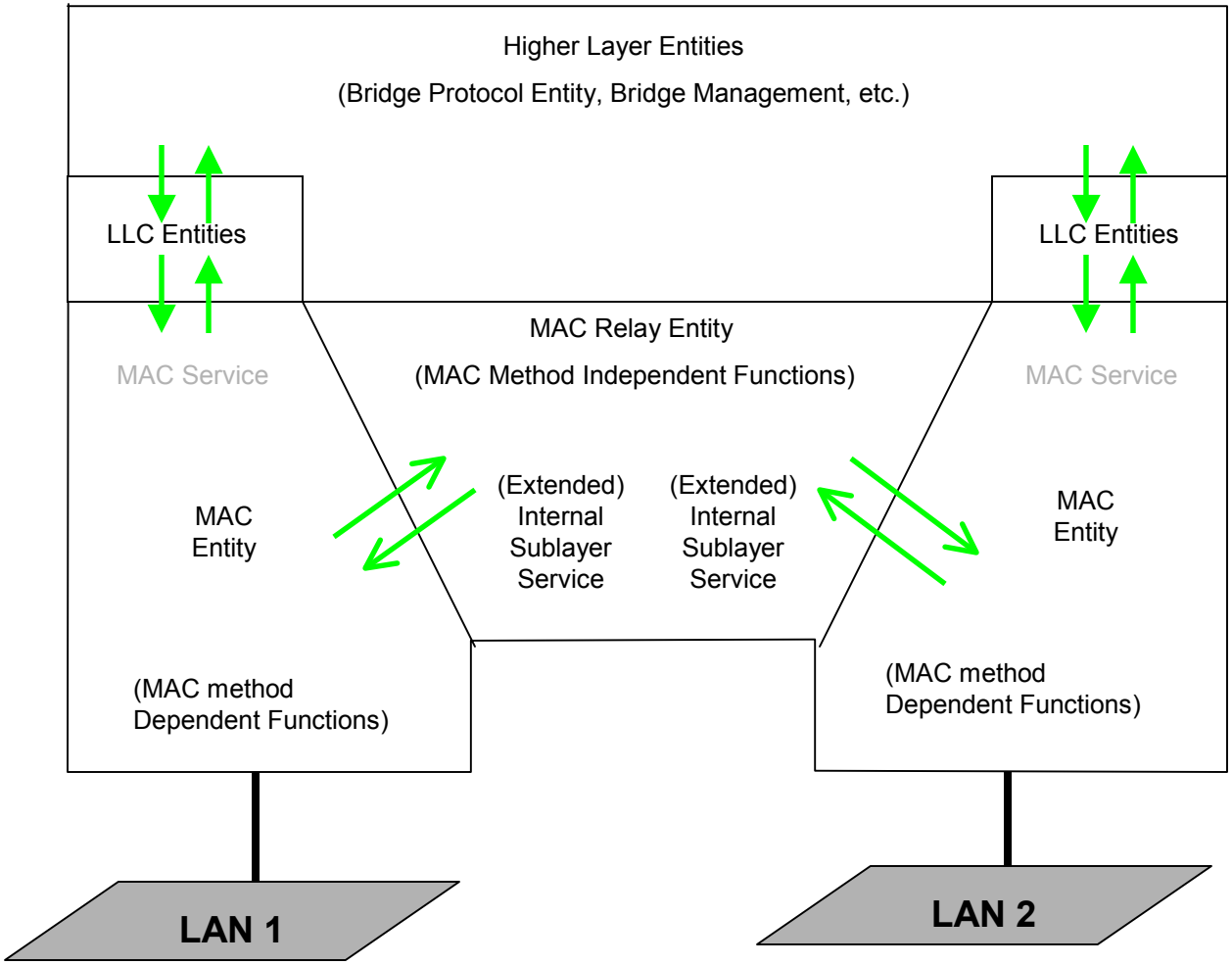
- 802.17 Bridge Reference Models
- Bridge Architecture
- Encapsulated Bridging Value Proposition
- Encapsulated Bridge Relay Entity Descriptions
- 802.17 MAC Extensions to support Encapsulated Bridging
- Conclusions

802.17 Bridge

- Station on Ring is Bridge
- Ring is the shared LAN media



802.1 Bridging Architecture



Encapsulated Bridging Scaling

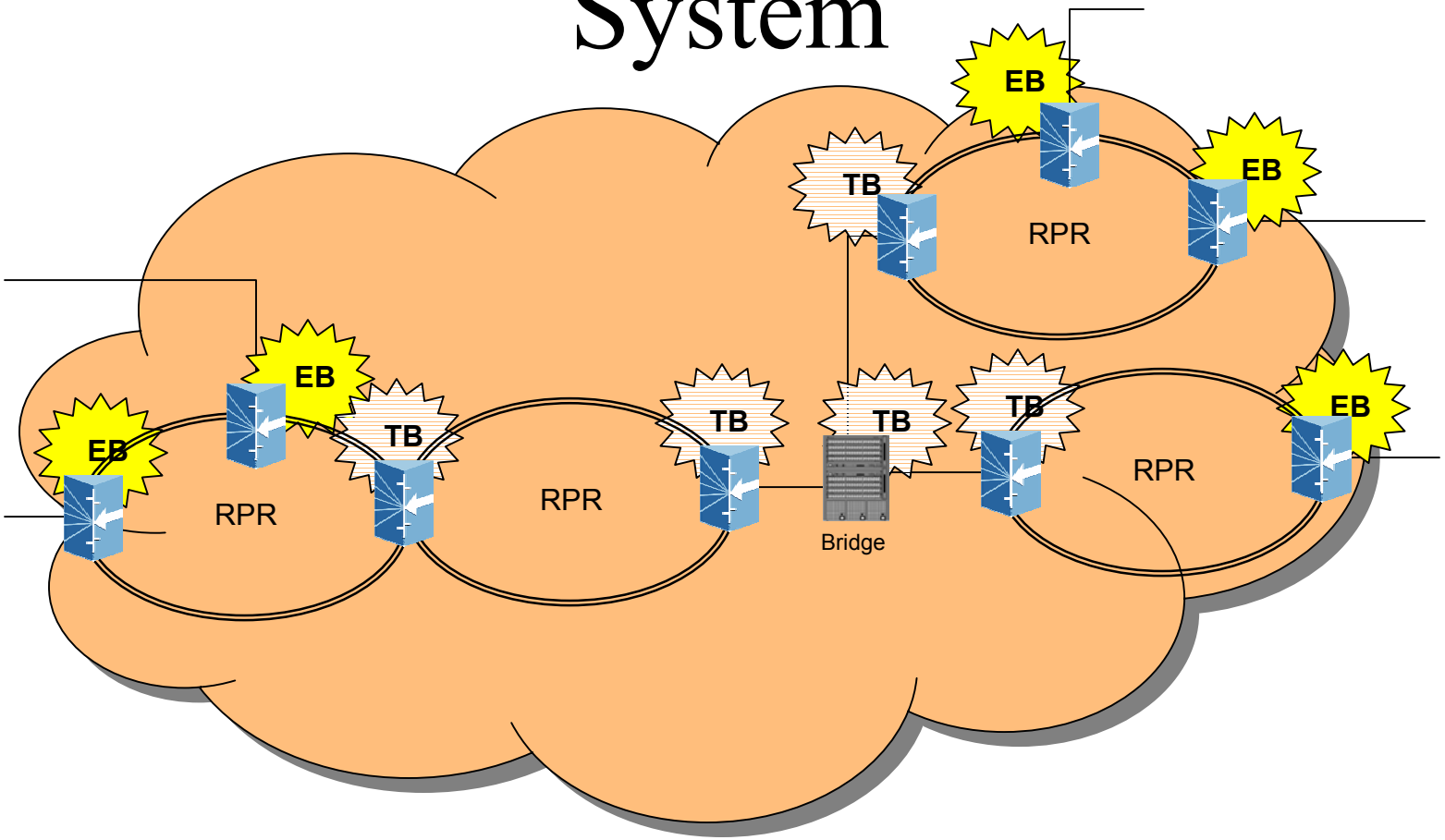
- From a (MAC learning) scaling perspective, Encapsulation Bridging benefits over Transparent Bridging is *only* realized in a Network setting (i.e., more than 1 switching device)
- Encapsulation Bridging (network) scaling benefits are not realized at the access/edge points of a Network, but at the core/interior of the Network

Encapsulated Bridging Value Proposition

	Edge/Access	Core
802.1D	• Bridge Relay FDB size scales with number of host MACs serviced by the Network. ***O(h_MAC)***	• Bridge Relay FDB size scales with number of host MACs serviced by the Network. ***O(h_MAC)***
Encap Bridge	• Bridge Relay FDB size scales with the number of RPR station MAC address within Service Provider Network. ***O(s_MAC)*** • Some sort of DB is needed to associated host MACs with Ring station MACs. Consequently, this DB scales with number of host MACs. ***O(h_MAC)***	• Run Transparent Bridge in the core. Consequently, the Bridge Relay FDB size scales with number of RPR station MACs. ***O(s_MAC)***

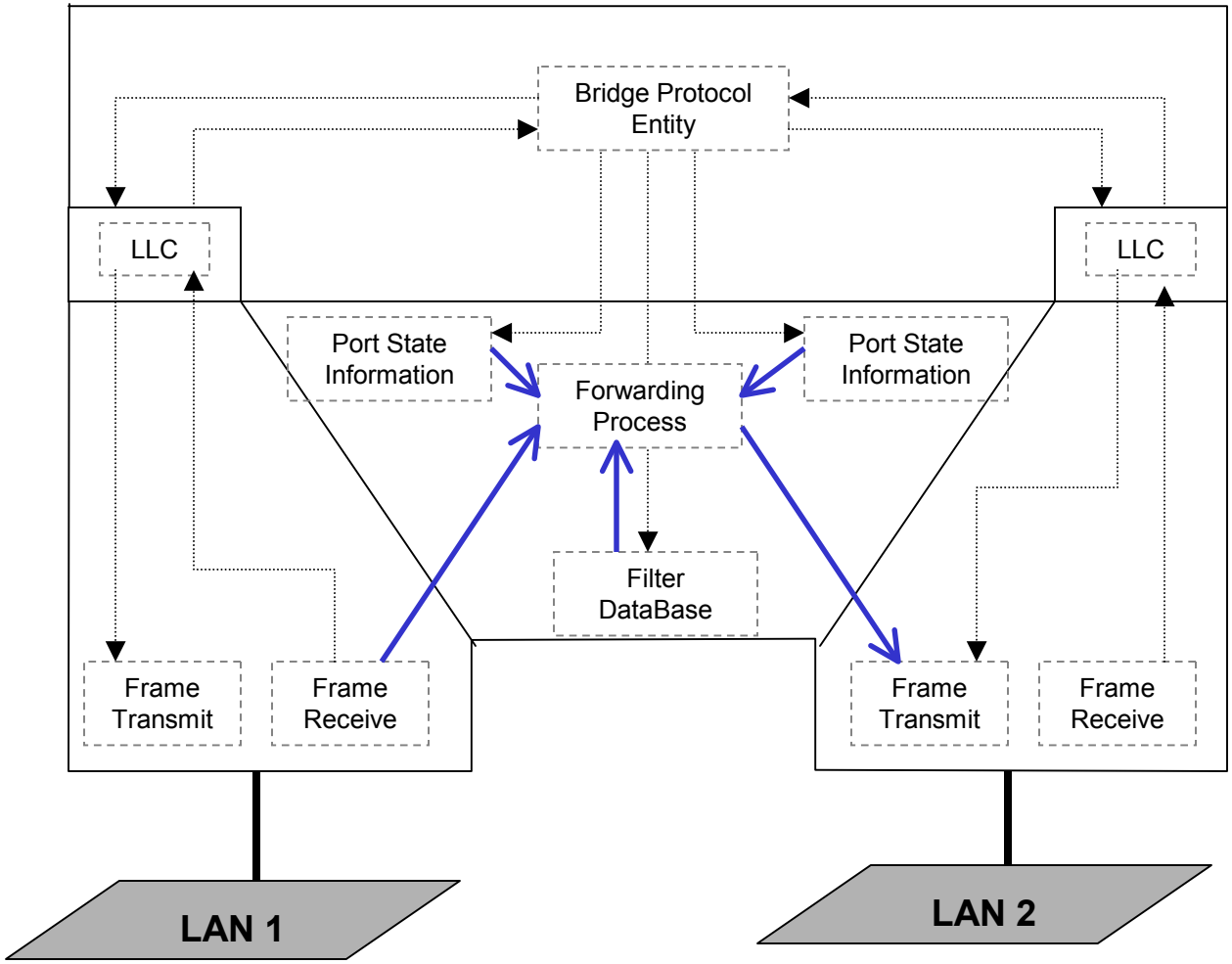
Assume: Range of h_MAC > s_MAC!!

802.17 Encapsulated Bridging System



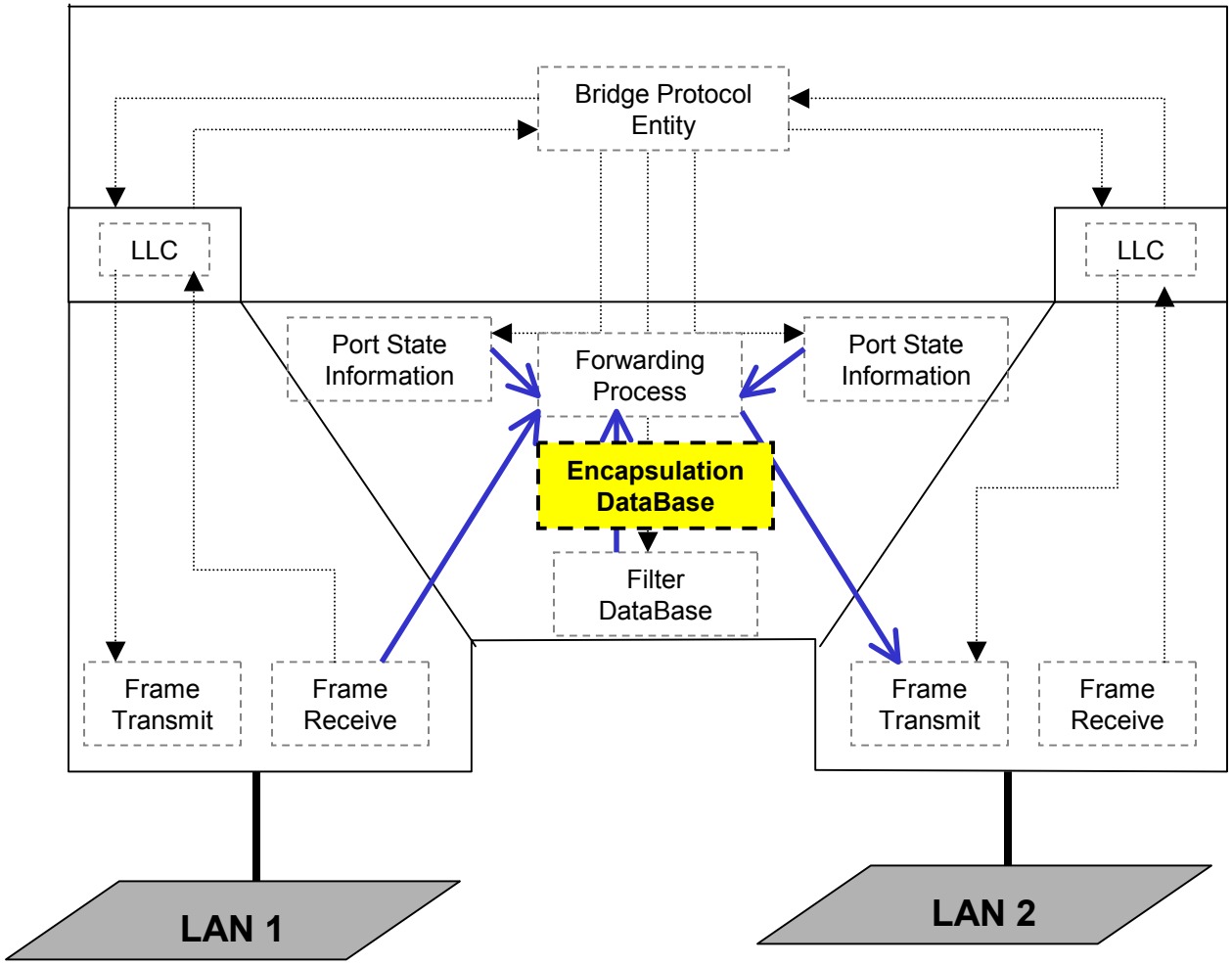
- **NOTE: Encapsulated Bridge (EB) have DB that scale with the host MAC address space. The Transparent Bridge (TB) have DB that scale with the Network MAC address space. Consequently, the core of the Network scales (and is 802.1D compliant).**
- **NOTE: Value proposition only realized in a (contained) Network when all edge nodes are EB conformant, and interior nodes are TB (802.1D/Q) conformant.**

Transparent Bridging (802.1D) Architecture



- Denotes Relaying MAC frames
- Denotes reception and transmission of BPDUs

Encapsulated Bridging Architecture



- Denotes Relaying MAC frames
- Denotes reception and transmission of BPDUs

Encapsulated Bridging Relay Entity

- Encapsulation Data Base is introduced. This DB will associate MAC address in one user space with MAC address in another user space
  - For example, associate MAC address in the Service Provider space (encapsulated address, typically those associated with the Stations on the Ring), and those of the Host MAC addresses
- Utilizes Filtering Data Base as defined by 802.1D. The addresses stored in this DB are those of the Customer space (e.g., Host address space), local to the station
- Relay Entity makes a decision to encapsulate packet with other address space (e.g., those associated with the MAC of the Ring stations), or de-encapsulate the relayed packet

NOTE: The model of operation is in no way intended to constrain real implementations of a MAC Encapsulated Bridge. These may adopt any internal model of operation compatible with the externally visible behavior that this proposal specifies. Conformance of equipment to this specification is purely in respect of observable protocol.

Encapsulated Bridging Internal Sublayer Service

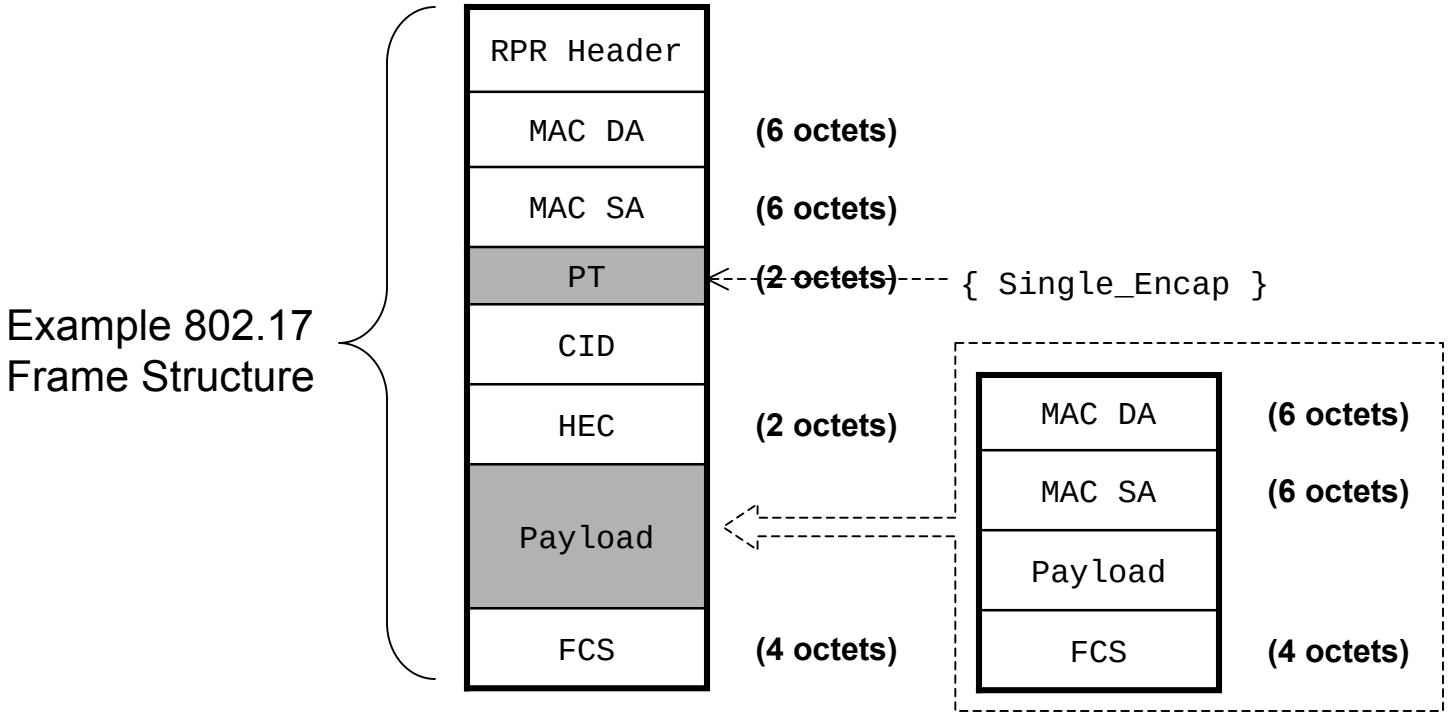
- A new Internal Sublayer Service (ISS) is defined for usage in the Encapsulated Bridge Relay Entities. This ISS will be referred to as the Encapsulated Bridge ISS (EB-ISS)
- The EB-ISS will be an extension of the ISS defined by 802.1D (and 802.1Q). An additional Service primitive parameter is introduced. The parameter will be referred to as `EncapID`. This is an optional parameter in both the Request and Indication Service primitives

  NOTE: Current non-conformant Encapsulated Bridging MACs (e.g., 802.3 MAC) will never include this parameter in an Indication Service primitive (sent to a MAC Relay Entity), nor will receive one in a Request Service primitive.

- The 802.17 MAC Entity will provide the EB-ISS to the Encapsulated Bridge Relay Entity
- Non-conformant Encapsulated Bridging MAC Entities will provide the applicable (802.1D/Q) ISS to the Bridge Relay Entity

802.17 MAC Frame Format

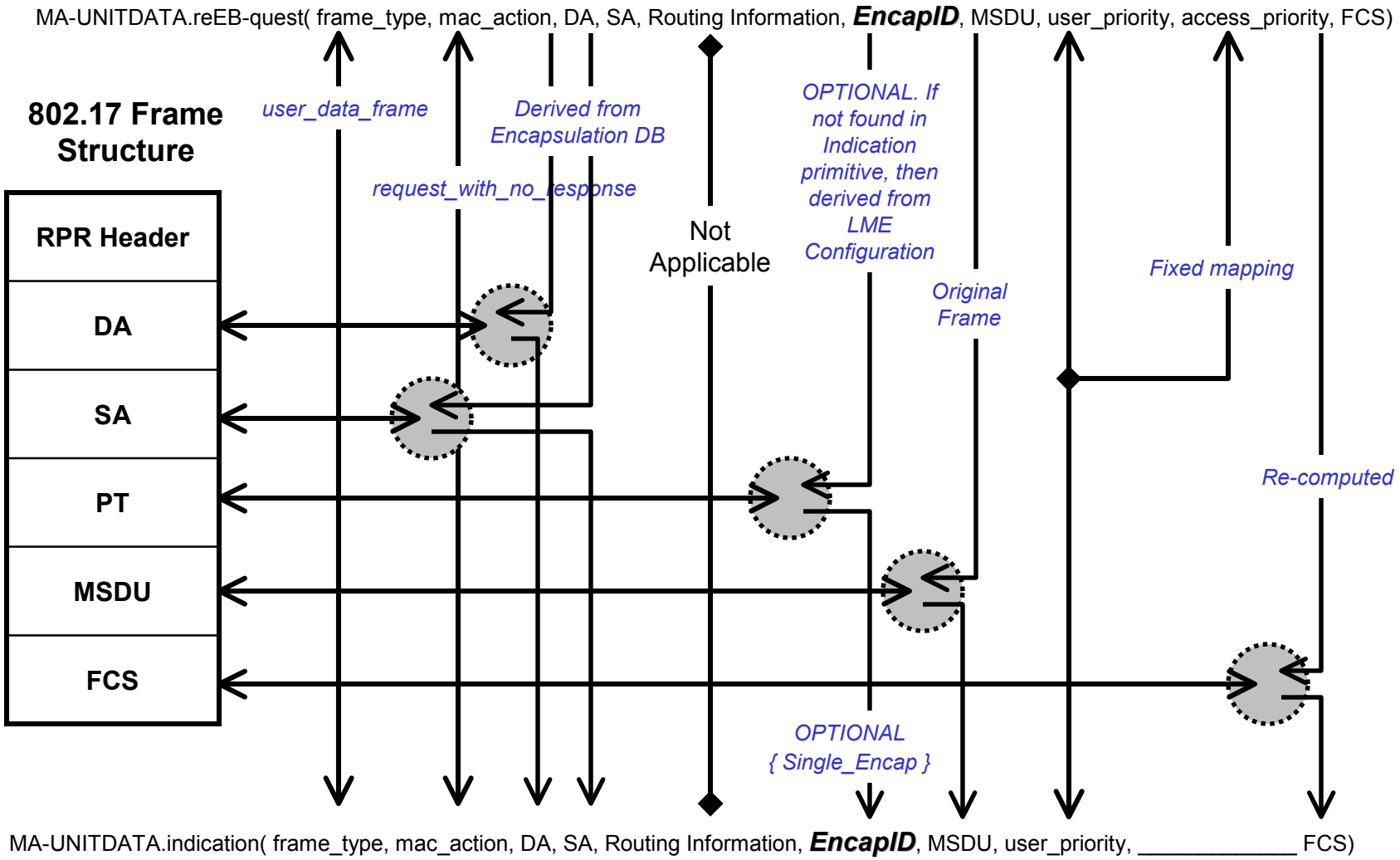
- Use the Payload Type (PT) field defined in the 802.17 frame structure to indicate that a packet is being encapsulated. The PT field is similar to the ET (Ethernet Type) field defined in 802.3
- An additional PT value will be defined. The new value that is being considered is `Single_Encap`



ISS Service Primitives & Parameters

Service Primitive	Transparent Bridge (802.1D)		Encapsulated Bridge	VLAN Bridge (802.1Q)
Indication	• Frame_Type • Mac_Action • DA • SA • RI • MSDU • User_Priority • FCS	+	• EncapID	• CFI • VLAN_Id • RIF_Info
Request	• Frame_Type • Mac_Action • DA • SA • RI • MSDU • User-Priority • Access_Priority • FCS	+	• EncapID	• CFI • VLAN_Class • RIF_Info • Include_Tag

EB-ISS Service Primitives Parameter Mapping



802.17 MAC Extension to Support Encap Bridging

- This proposal builds on top of the 802.17 MAC extensions to support 802.1D/Q compliant Bridges
- Upon frame reception, the 802.17 MAC will interpret the Payload Type (PT) in the 802.17 frame. If the PT denotes single Encapsulated Bridging (`Single_Encap`), then the `EncapID` parameter (found in the EB-ISS Service Indication Primitive) is updated. The EB-ISS defined Service Indication Primitive is sent to the Encapsulated Bridging Relay Entity
- Prior to frame transmission on the RPR, the 802.17 MAC will interpret the `EncapID` parameter found in the EB-ISS Service Request Primitive (if present) and set the PT field found in the 802.17 frame appropriately

Conclusions

- Encapsulated Bridging proposal supports 802.17 MAC.
- Encapsulated Bridging proposal is in-line with 802.1 Bridging Architecture.

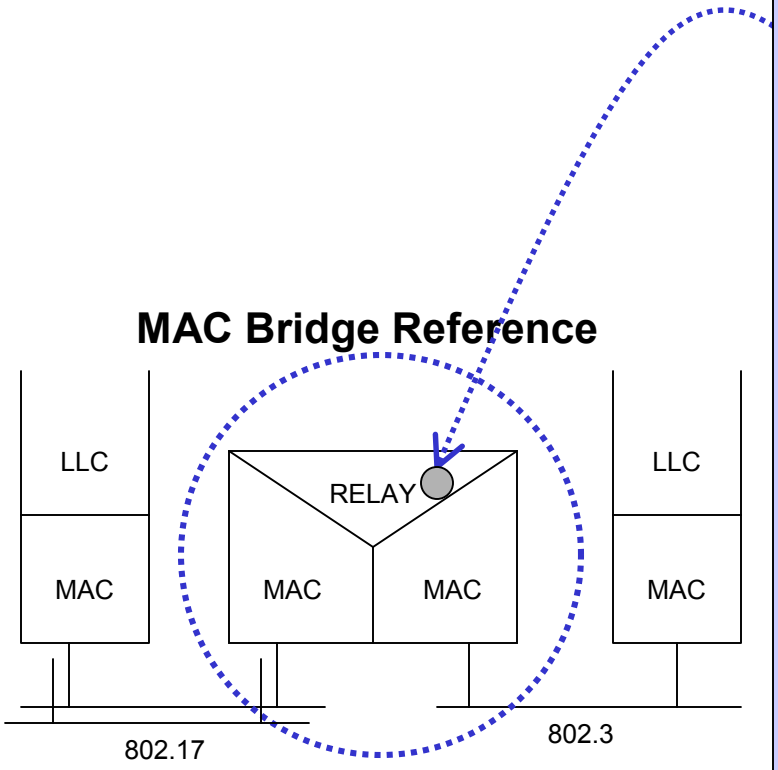
Proposal Cost-Benefit Analysis

- ☑ Re-uses the 802.1D defined Bridge Protocol Entity
- ☑ Supports STP (as defined in 802.1D), since can use 802.1D currently defined Bridge Protocol Entity
- ☑ 802.1D defined interface between 802.17 MAC Entity and LLC layer can be used
- ☑ Bridge aligns with 802.1 Architecture
- ☑ Bridge Relay Entity is a simple extension of 802.1D defined Bridge Relay Entity
- ☑ Solution can easily be extended to support other 802 MACs (e.g., 802.3)

- ✘ New Bridge Relay Entity needs to be introduced

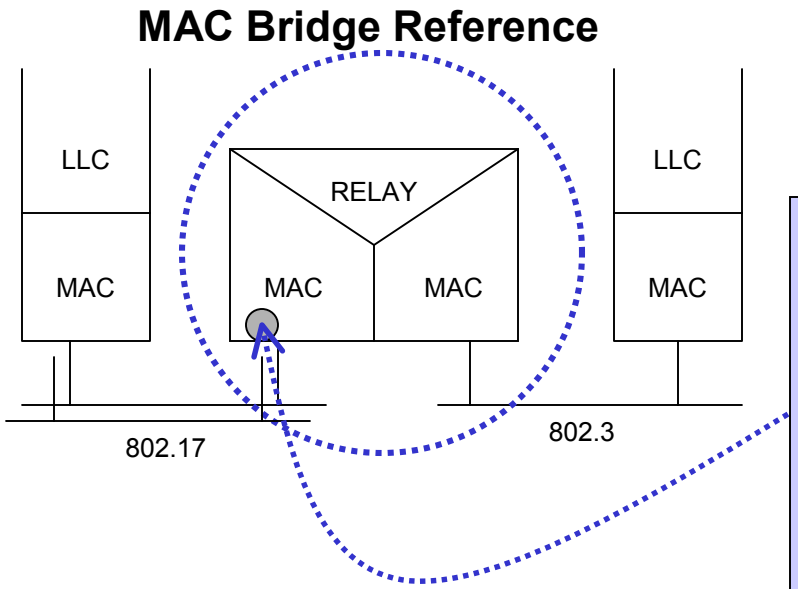
Back Up Charts

Encapsulated Bridge Relay Pseudo-Code




```
Case LME.Bridge in:
{ Encapsulate Bridge }:
   If Indication.EncapID
      /* Decapsulate Frame */
      { Single_Encap } -> type
      Indication.MSDU( type, SA ) -> Request.SA
      EncapDB.Update( Indication.SA, Request.SA, age )
      Indication.MSDU( type, DA ) -> Request.DA
      Indication.MSDU( type, Payload ) -> Request.MSDU
      Indication.MSDU ( type, FCS ) -> Request.FCS
   Else
      /* Encapsulate Frame */
      FDB.Update( Indication.SA, port, age )
      EncapDB.Update( Station.Addr, Indication.SA, age )
      { Single_Encap } -> Request.PT
      Request.MSDU( Indication.SA, Indication.DA, Indication.MSDU,
      Indication.FCS )
      EncapDB.Index( Indication.DA ) -> Addr
      If ( addr == NIL )
         #FFFFFF -> Request.DA
      Else
         Addr -> Request.DA
      End If
      Station.Addr -> Request.SA
      Request.ReComputeFCS()
   End If
{ Transparent Bridge }:
   /* Perform 802.1D/Q operations */
End Case
Request.DispatchToMAC()
```

802.17 MAC Enhancements Pseudo-Code

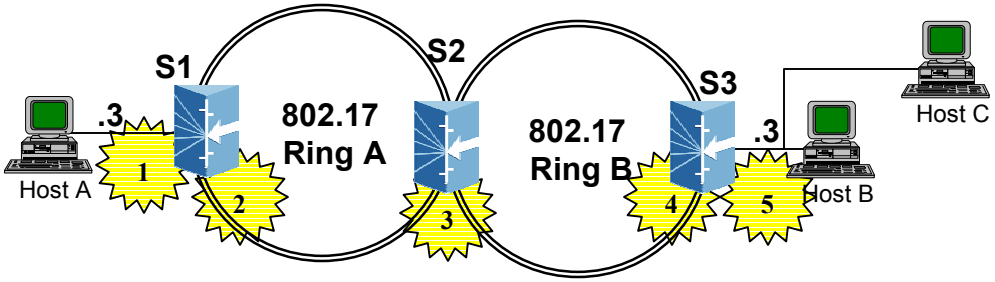



```
Case event
{ receive 802.17 Frame off Ring }
   /* Do 802.17 MAC specified packet reception logic */
   If ( Pass Frame to Bridge Relay )
      /* Build Service Indication Primitive */
      :
      If ( frame.PT == Single_Encap )
         { Single_Encap } -> Indication.EncapID
      Indication.DispatchToBridgeRelay()
   End If

{ Receive Service Request Primitive }
   Convert( Request.EncapID ) -> frame.PT
   :
End Case
```

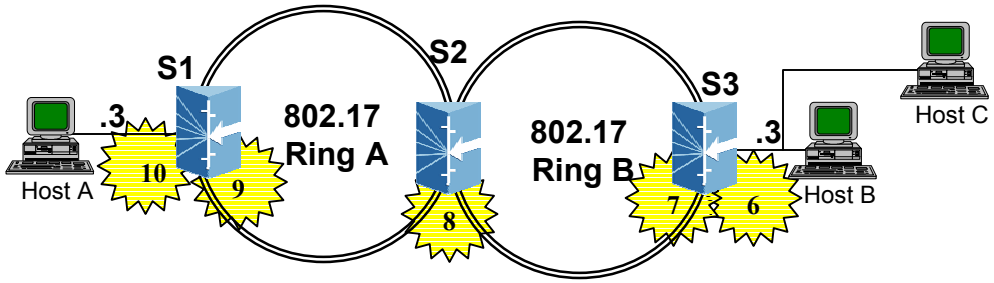
Example #1: High Level Walk-Thru

Station S1 and S3 are configured to be Encapsulated Bridges. Staiton S2 is configured to be a Transparent Bridge (802.1D). Assume no DB learning has occurred.



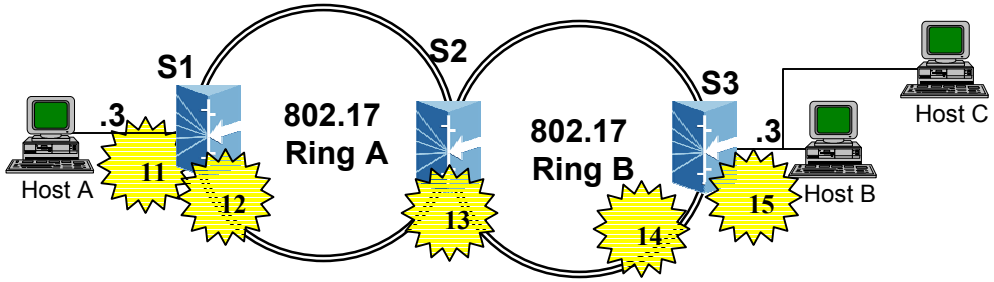
Step	Description	Packet Format	FDB	Encap DB
1	HostA sends packet destined to HostB	B | A	N/A	N/A
2	RPR S1 receives packet and Bridges/Relays onto RingA.	FF | S1 | B | A	*[A, p1]*	*[S1, A]*
3	RPR S2 receives packet and process as a TB.	FF | S1 | B | A	*[S1, PortA]*	N/A
4	RPR S3 receives packet.	FF | S1 | B | A	-	*[S1, A]*
5	HostB receives packet.	B | A	N/A	N/A

Example #1: High Level Walk-Thru



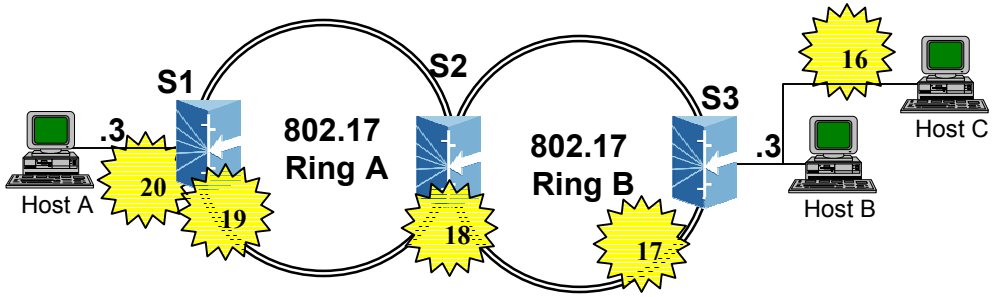
Step	Description	Packet Format	FDB	Encap DB
6	HostB responds to HostA.	A | B	N/A	N/A
7	RPR S3 receives packet and Bridges/Relays onto RingB.	S1 | S3 | A | B	*[B, p3]*	[S1, A] *[S3, B]*
8	RPR S2 receives packet and process as a TB.	S1 | S3 | A | B	[S3, PortB] *[S1, PortA]*	N/A
9	RPR S1 receives packet.	S1 | S3 | A | B	[A, p1]	[S3, B] *[S1, A]*
10	HostB receives packet.	B | A	N/A	N/A

Example #1: High Level Walk-Thru



Step	Description	Packet Format	FDB	Encap DB
11	HostA responds to HostB	B | A	N/A	N/A
12	RPR S1 receives packet and Bridges/Relays onto RingA.	S3 | S1 | B | A	[A, p1]	[S3, B] [S1, A]
13	RPR S2 receives packet and process as a TB.	S3 | S1 | B | A	[S3, PortB] [S1, PortA]	N/A
14	RPR S3 receives packet.	S3 | S1 | B | A	[B, p3]	[S1, A] [S3, B]
15	HostB receives packet.	B | A	N/A	N/A

Example #1: High Level Walk-Thru



Step	Description	Packet Format	FDB	Encap DB
16	HostC sends packet destined to HostA.	A | C	N/A	N/A
17	RPR S3 receives packet and Bridges/Relays onto RingA.	S1 | S3 | A | C	[B, p3] *[C, p3]*	[S1, A] [S3, B] *[S3, C]*
18	RPR S2 receives packet and process as a TB.	S1 | S3 | A | C	[S3, PortB] [S1, PortA]	N/A
19	RPR S1 receives packet.	S1 | S3 | A | C	[A, p1]	[S3, B] [S1, A] *[S3, C]*
20	HostA receives packet.	A | C	N/A	N/A