All-Path bridges

Guillermo Ibanez (UAH), Jun Tanaka (Fujitsu), Vinod Kumar (Tejas Networks)

IEEE Interim meeting

Santa Fe, 9-12 May 2011

guillermo.ibanez@uah.es

FUJITSU

Universidad de Alcalá
Dpto. de Automática
Área de Ingeniería Telemática

La Suma de Todos
CONSEJERÍA DE EDUCACIÓN
Comunidad de Madrid
www.madrid.org

Junta de Comunidades de
Castilla-La Mancha

Objectives

- Advance in definition: scope, requirements, compatibility
- Address Singapore presentation issues:
  - Describe improved path repair
- Show load distribution results in a data center topology

Summary

- Introduction
- Defining the scope: All-Path
- Towards standardization
- Recapitulation from Singapore presentation
- Path repair requirements
- Improved path repair mechanisms
- Profiles/targets of All-Path protocol
  - Compatibility with IEEE 802 protocols
  - All-path Hosts, All-path M-in-M
- Load distribution results

Introduction

- A new family of transparent bridges is emerging
  - Modified backward address learning
  - No spanning tree needed
  - All possible paths are explored
  - Proposed names:
    - Generic bridges of this type: **All Path bridges**
    - Fast Path (a.k.a. ARP-Path, Broad-path):our bridges proposal.
      - Keep the name Fast-Path (or as separate words, Fastpath is a registered Trademark), to avoid further confusion?.

Defining the scope

- *All-Path* bridges: bridges that explore all possible paths with broadcast,multicast or replicated frames when setting a path or a tree (and select one or several paths on a frame arrival basis, not necessarily the fastest one).
  - Modified learning, filtering and forwarding

Towards standardization

- All-Path bridges use basic mechanisms that can be applied in a variety of bridges: 802.1D, current 802.1Q, 802.1aq
- Being a new concept, it makes sense (probably) first to explore it in basic bridges (802.1D)
- All-Paths mechanisms could be useful in 802.1aq's SPBV and SPBM bridges (e.g. multicast trees) but not a priority, unless a unique benefit is obtained.
  - Quite different variants of All-Path protocol would likely be required due to different requirements of SPBV and SPBM

All-path bridges as amendment to 802.1D

- Amend the IEEE 802.1D standard, adding alternative (optional) forwarding to the RSTP protocol
  - Just one step beyond RSTP
  - Coexisting with RSTP:
    - RSTP as independent protocol (per port protocol separation)
    - RSTP as backup protocol (RSTP could be used for path repair)
- As a VLAN independent solution to the Spanning Tree Protocol limitations.
  - Because VLAN dependent solution would fit more in 802.1aq work or as a separate 802.1Q amendment
- It seems the option adding most value and less disturbant

All-Path bridges requirements (tentative)

- Network size: small, medium (500 nodes),20 K hosts
- Zero (or near zero) configuration
- Single IP subnet
- RSTP compatible, although not 100% miscibility.
- VLAN/ 801.Q compatible
- Reconfiguration time: like RSTP or better

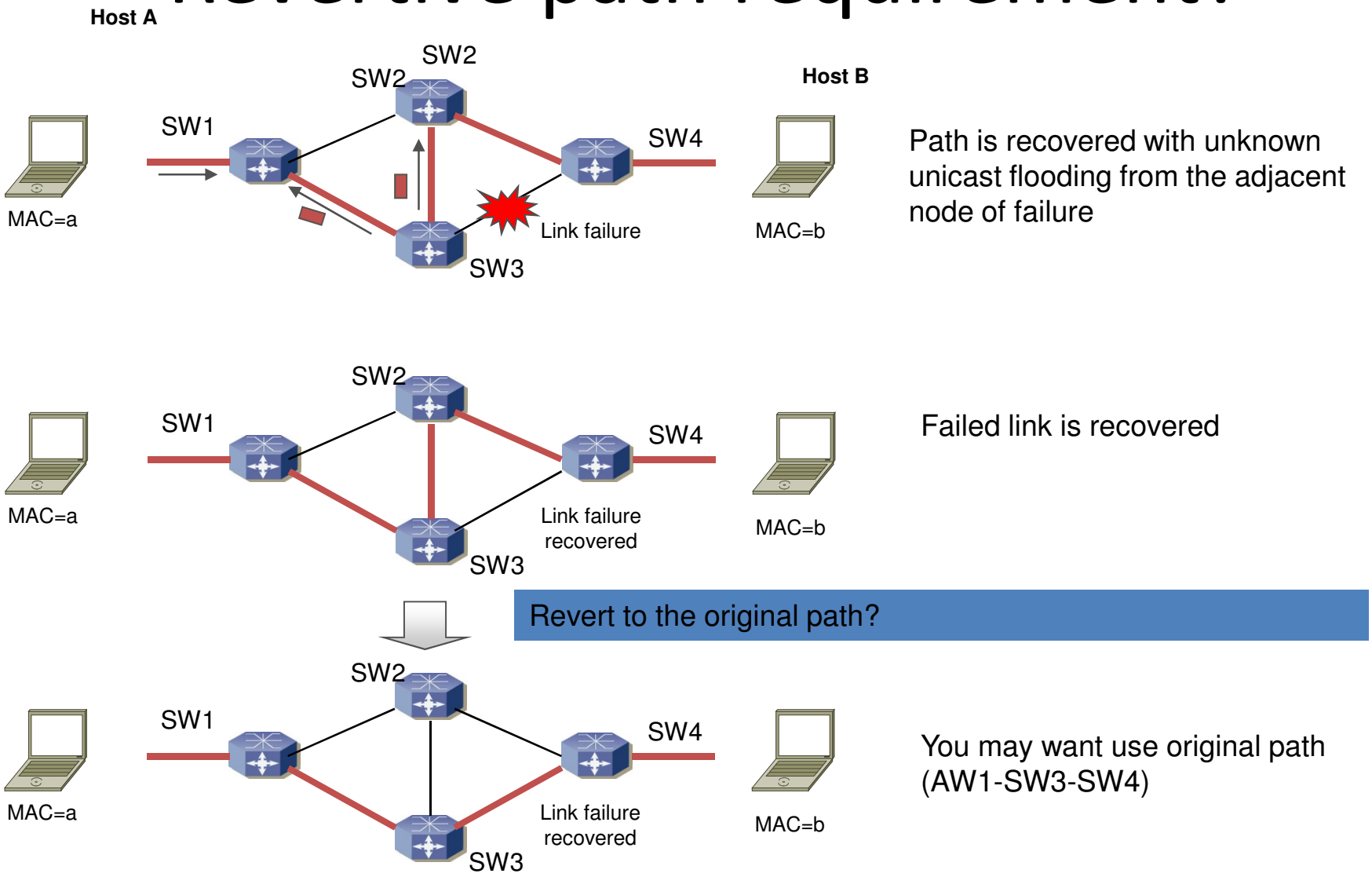
Recapitulation Singapore Broad Path Presentation

- Two main issues raised:
  - Processing effort of path repair. Two special broadcast frames (encapsulate unknown unicast frame) are required per active flow.
    - Path repair is performed on-demand, per path.
    - Unused paths do not need repair.
    - And processing load is distributed along the time.
    - ≈33 packets/sec for 10K flows (explained later)
  - Difficulty of determining by hardware the first port receiving the replicated frame.
    - Fixed (first port) versus dynamic (statistically first)
    - Means: Statistics, queue,locking circuits…
    - Open area for hardware research (hardware challenge)

Path Repair Requirements

- Definition: Path Repair is the process of reestablishing lost paths in switches, normally by MAC that are flushed after link, switch failure, initialization or expiration.
- Path repair shall not affect existing traffic, only exceptionally.
- Path Repair shall require low to moderate processing effort per bridge. (Up to 10 K active flows per link).
- Low to moderate message overhead
- Optional: it shall not produce frame loss
- Shall work in all topologies.
- Selectable alternative path repair mechanisms

Revertive path requirement?



Summary

- Introduction
- Defining the scope: All-Path/ New paradigm
- Path repair requirements
- **Improved path repair**
- Profiles/targets of All-Path protocol
  - Compatibilities with IEEE 802 protocols
  - All-path Hosts, All-path M M-in-M
- Load distribution results

Improved Path Repair Mechanisms

IEEE Interim Meeting G. Ibáñez

Santa Fe, 9-12 June 2011

Path Repair load

- Previously, two broadcasts per flow being repaired
- To be issued by the bridge at an average rate of **33 packets per second**:Assuming 10 K active flows at failed link, the number of arrived frames belonguing to different flows (destinations) must be at least 10000 MACs/ 300 seconds (5 minutes default value) expiration time = 33 different flows per second (to keep all addresses alive at the caché MAC table.
- Convenience of making this load lighter

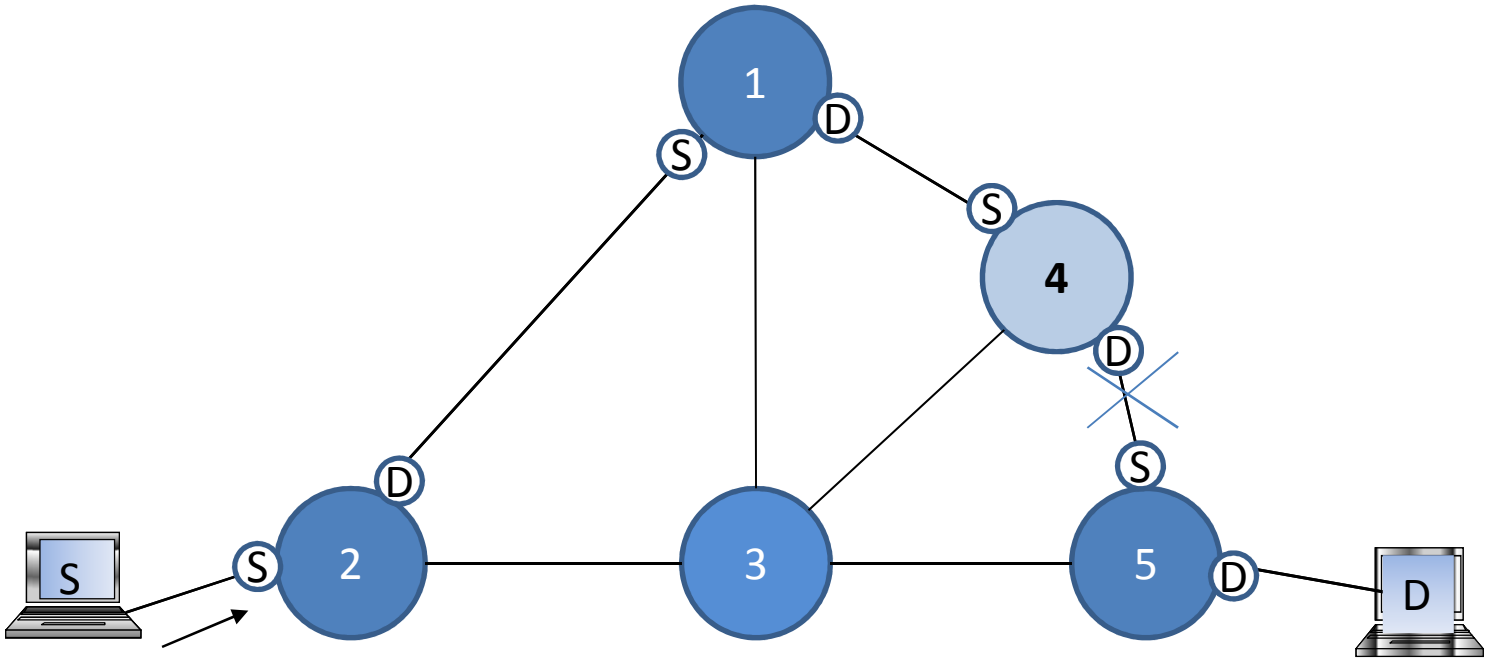
Improved path repair. Summary

- We propose two new mechanisms:
  - **Loop back** (reverse forwarding) of unknown unicast frames.
  - **Path proxy** mechanism at every bridge accelerates repair and reduces broadcast.

Reverse forwarding (frame loop back)

- Reverse Forwarding of unknown unicast frames.
  - When a frame arrives at a bridge and it doesn't know the destination of the frame, the bridge will send the frame backwards, using the port stored in its learning table for the source of the frame using an **reverse forwarding mode** (the forwarding logic is reverted: SA is handled as DA and viceversa).
- Reverse forwarding can be the normal forwarding mode for **looped back frames** (frames that arrive at a port that is already associated to its DA, i.e. frames that arrive at the port they should be normally output to).
- The frame is sent backwards at every bridge until the source edge bridge is reached.
- If a bridge does not have a port associated to DA, default repair is initiated (Path Fail broadcast)

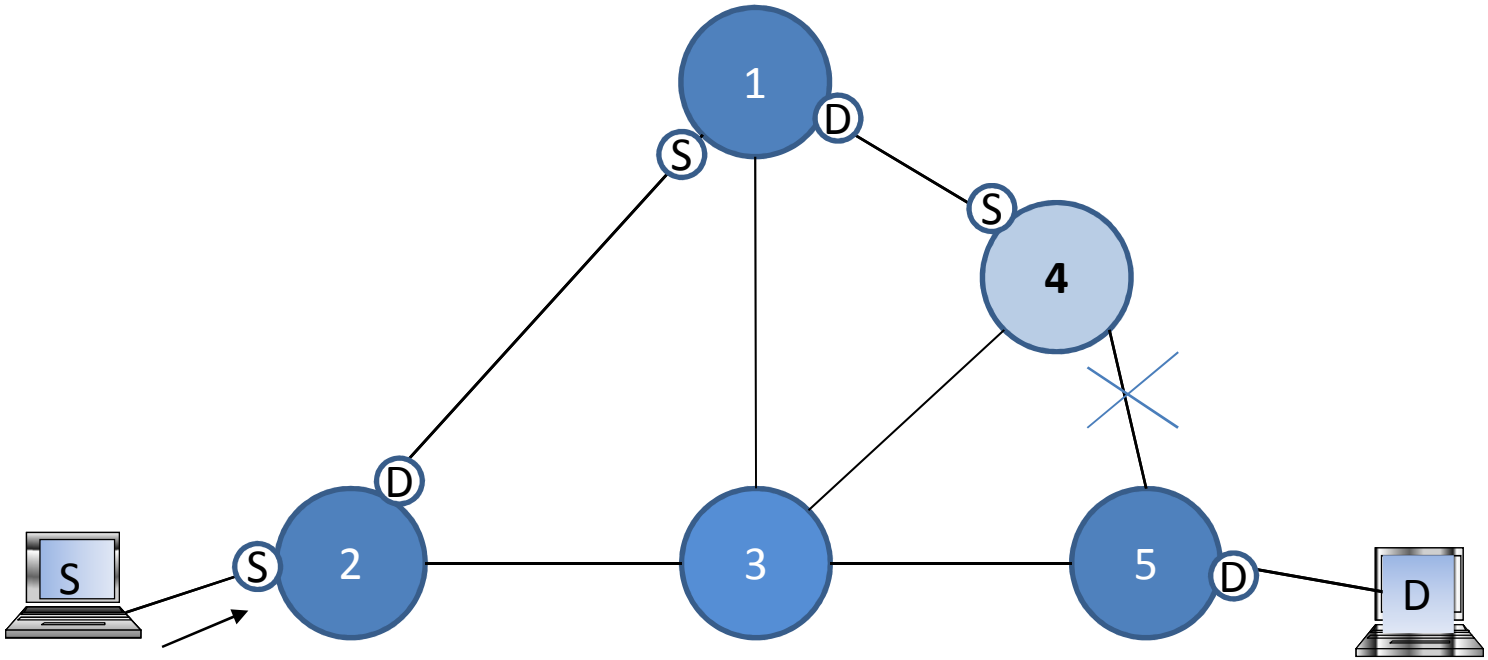
New path repair with reverse forwarding (link between bridges 4-5 fails)



S *Port locked to S*

D *Port locked to D*

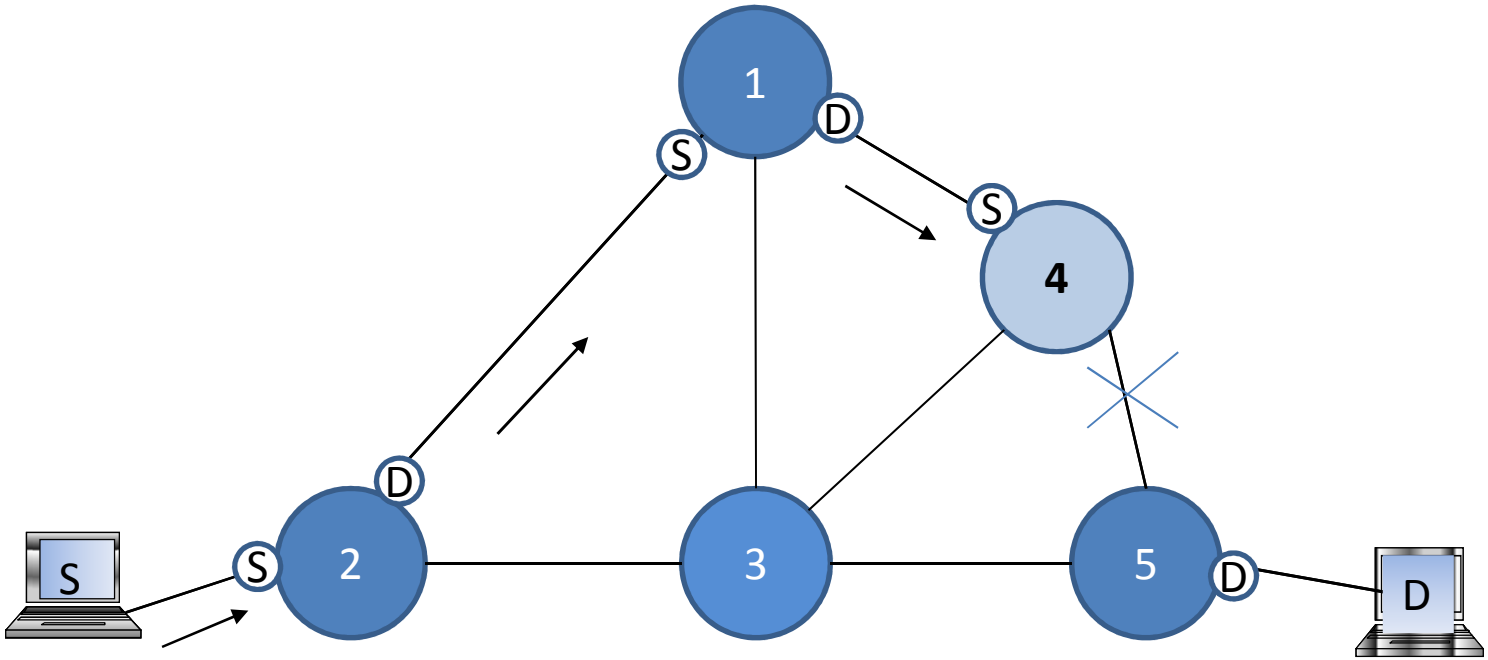
bridges 4 and 5 flush all MACs of related port after link failure



S *Port locked to S*

D *Port locked to D*

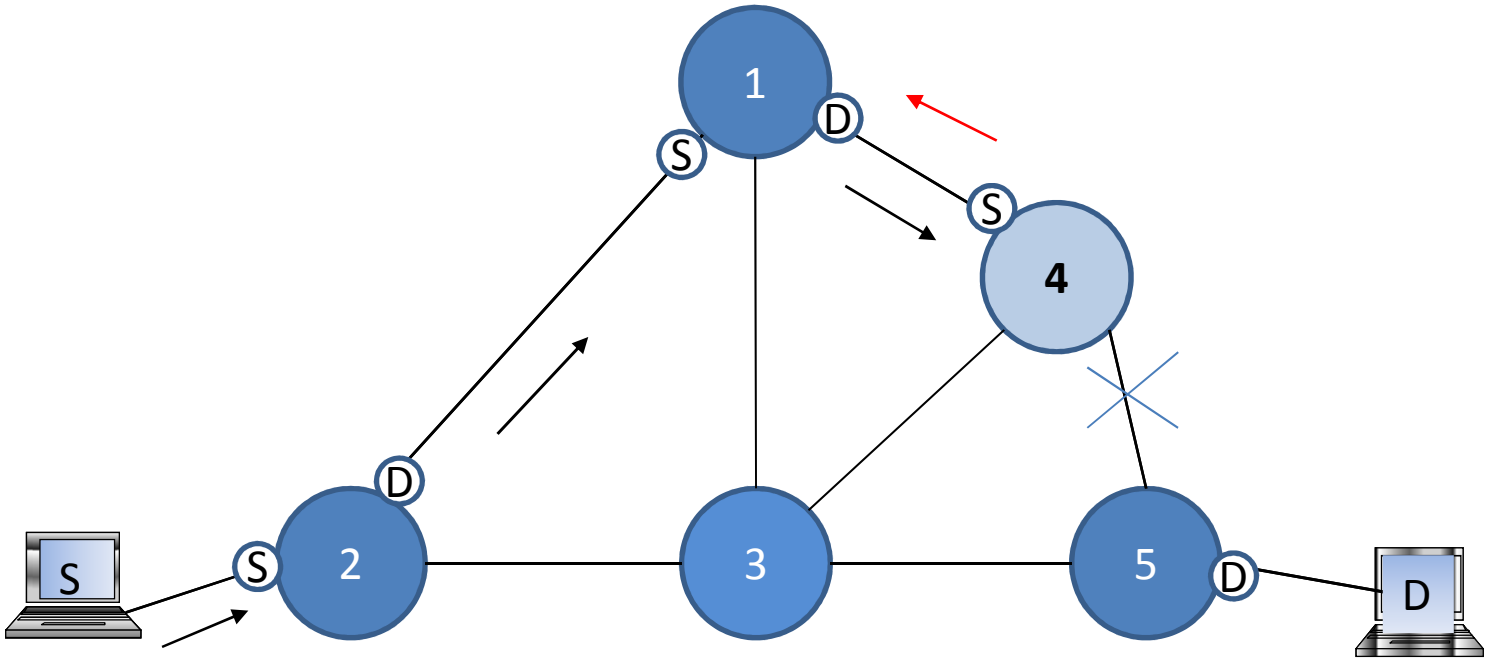
Unicast packet arriving



S *Port locked to S*

D *Port locked to D*

Unknown unicast frame is looped back towards S (SA,DA roles interchanged)

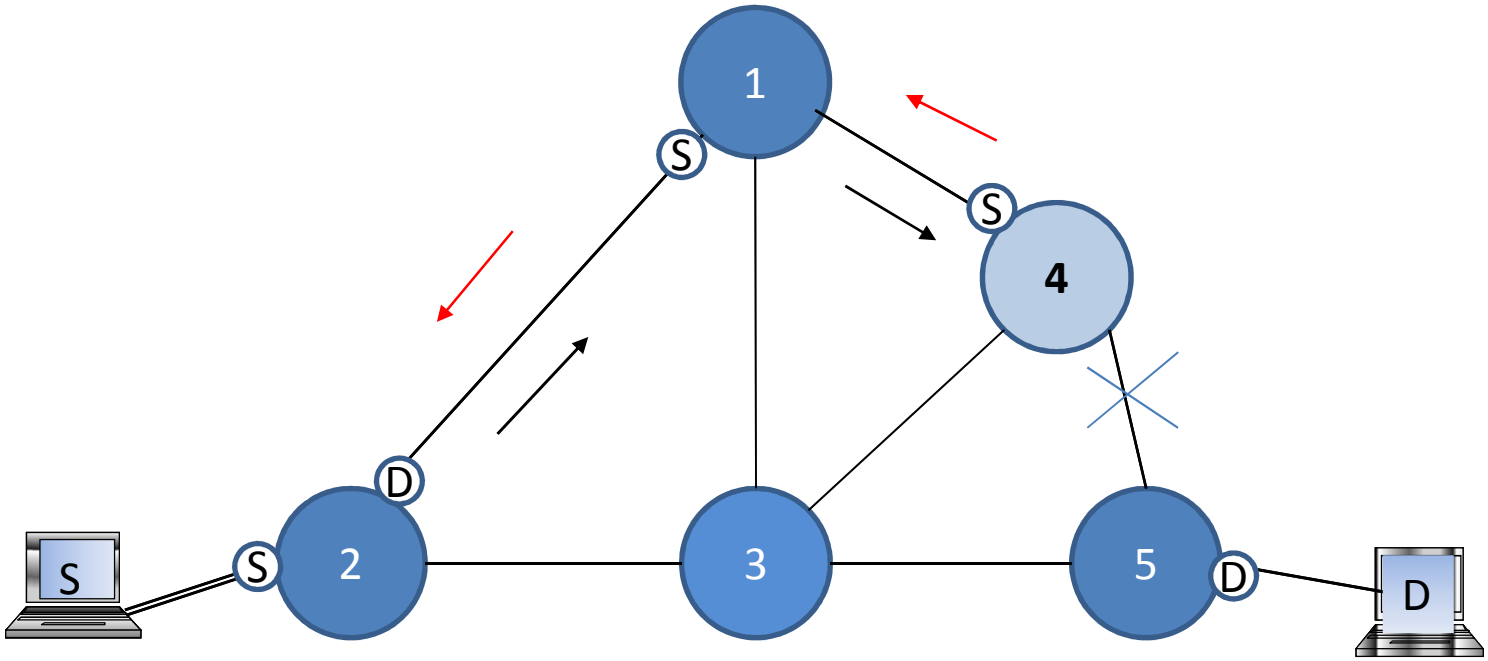


S *Port locked to S*

D *Port locked to D*

Looped back frame erases (optionally) the backwards path

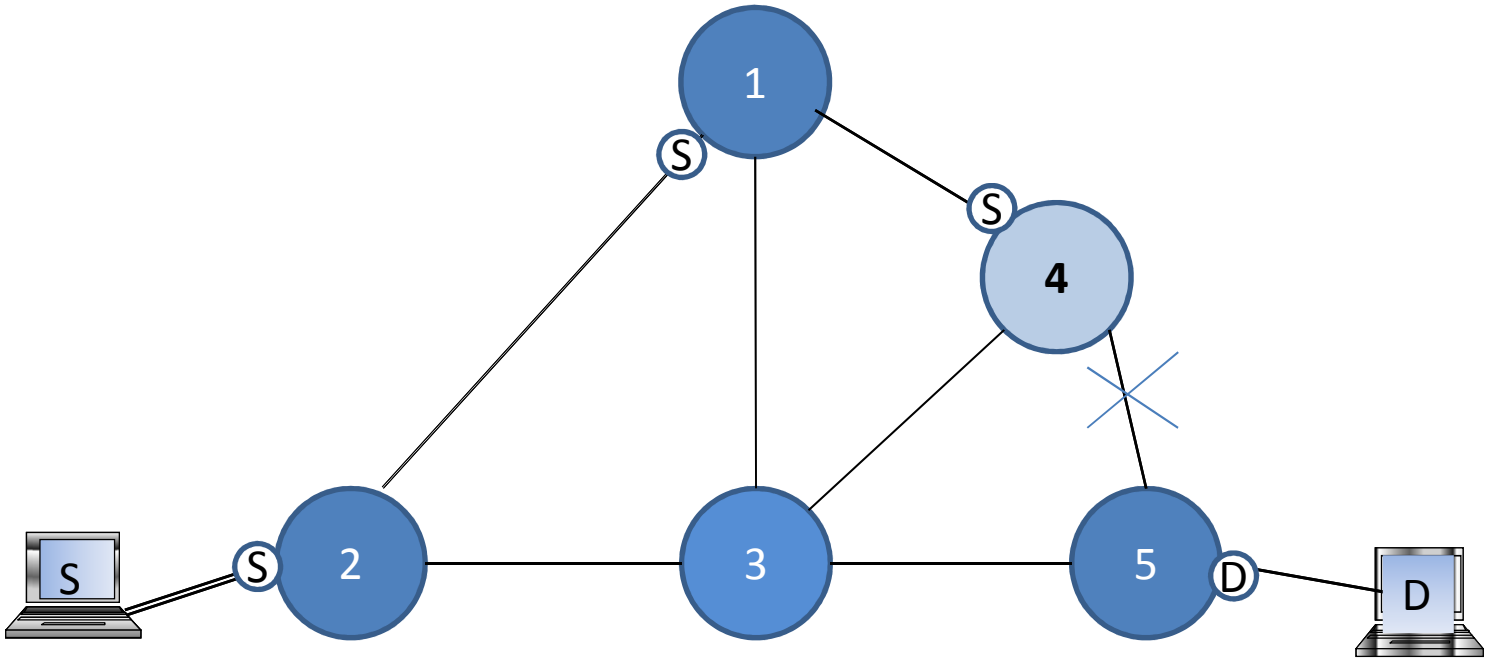
Looped frame reaches source bridge 2



S *Port locked to S*

D *Port locked to D*

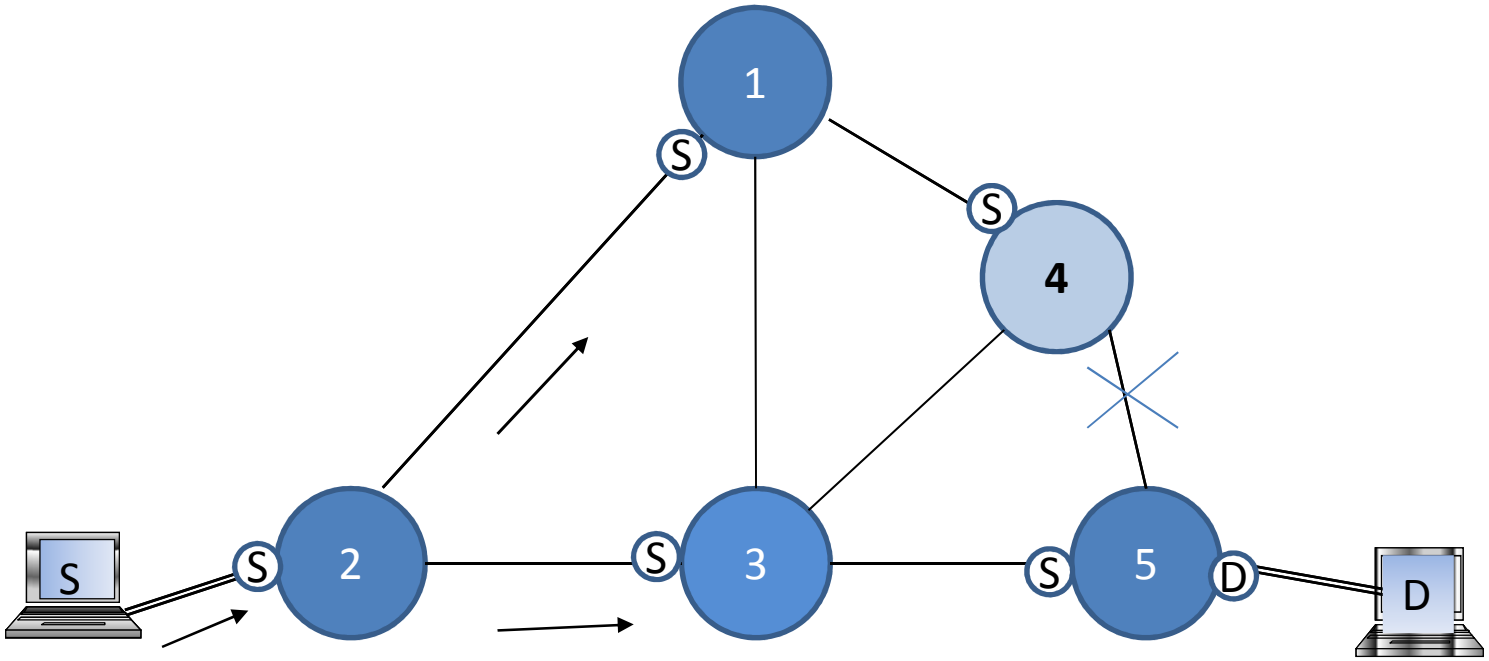
Looped frame reaches source bridge 2



S *Port locked to S*

D *Port locked to D*

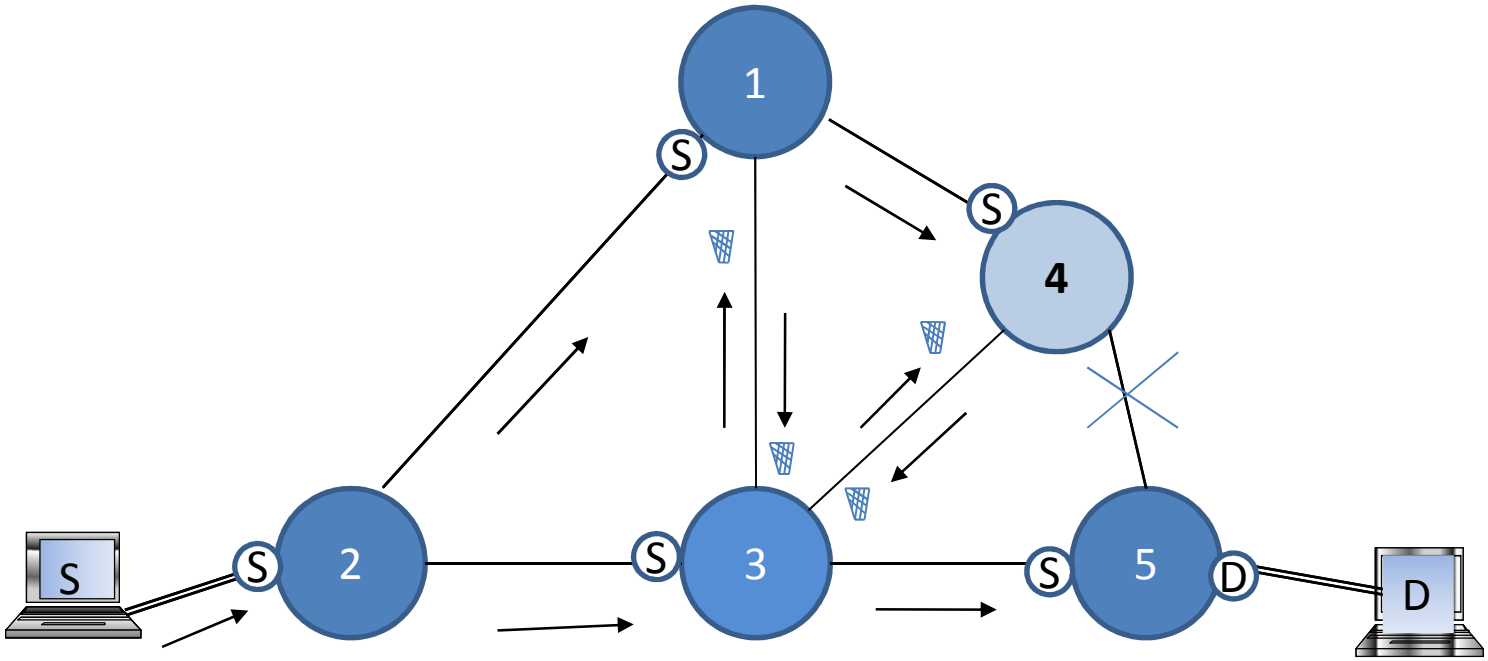
Source bridge sends ARP Request or Path Request packet to set new path



S *Port locked to S*

D *Port locked to D*

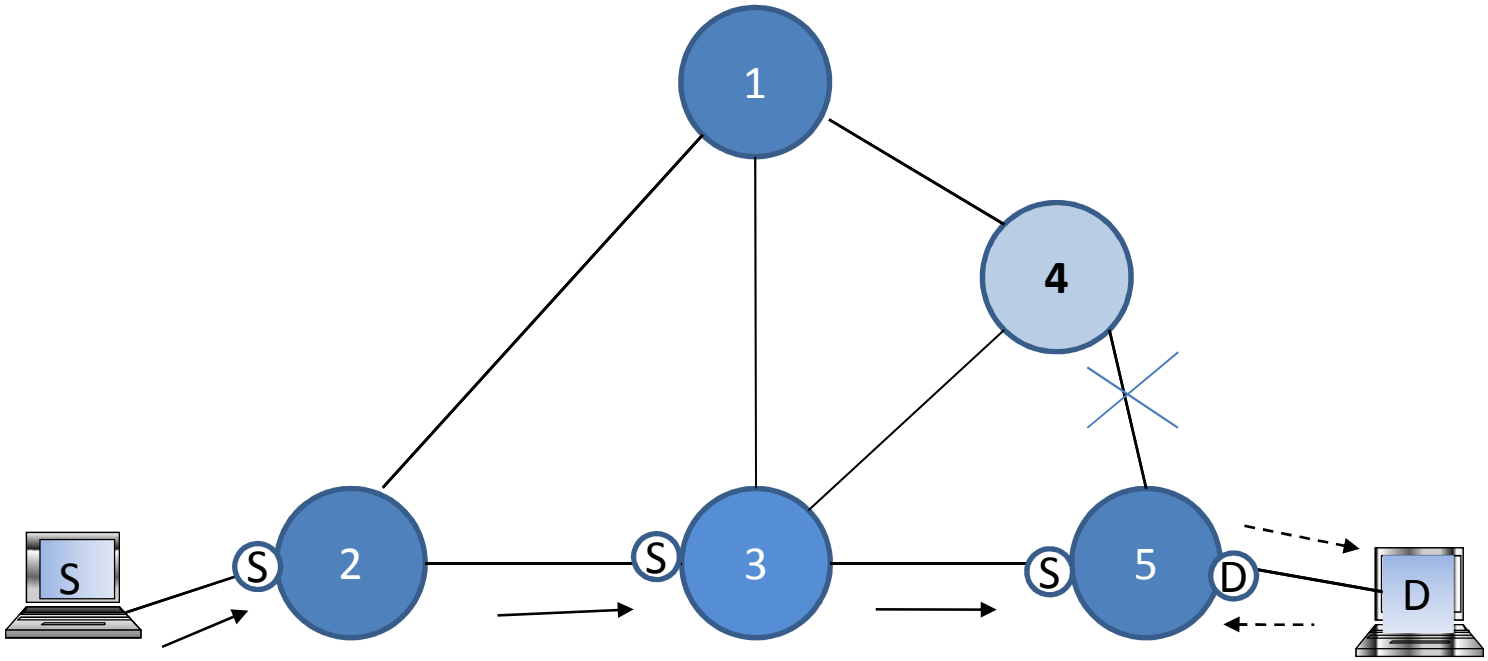
Broadcast propagates and selects paths to S



S *Port locked to S*

D *Port locked to D*

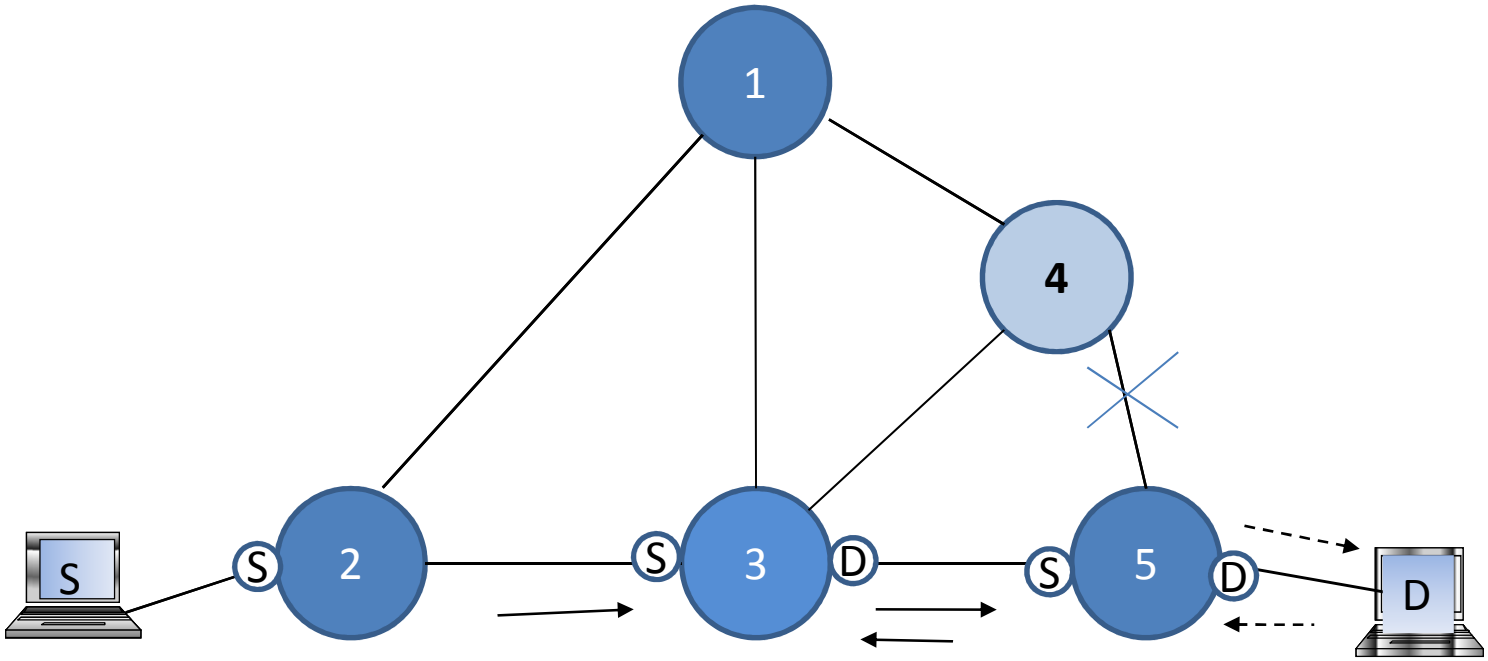
ARP Reply or Path Reply completes the path



S *Port locked to S*

D *Port locked to D*

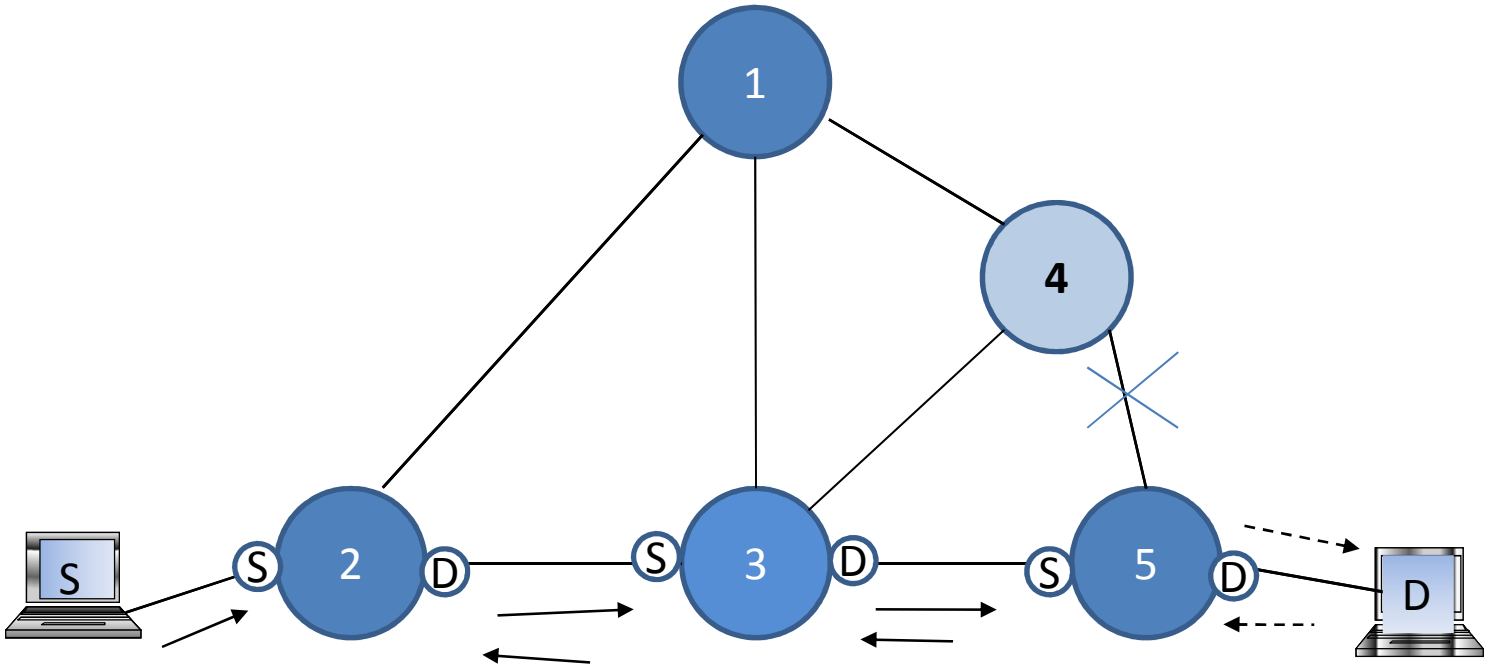
ARP Reply or Path Reply completes the path



S *Port locked to S*

D *Port locked to D*

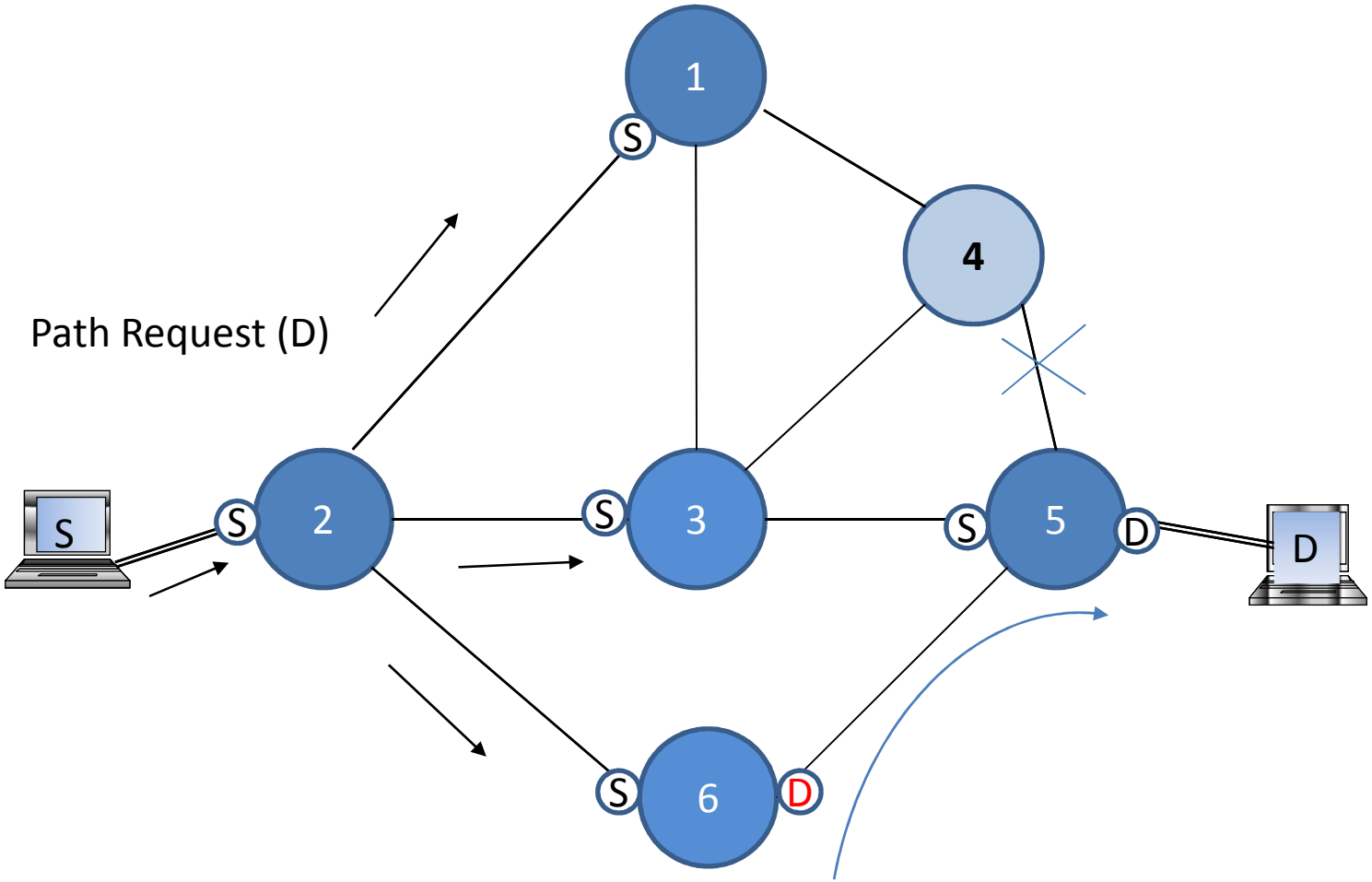
ARP Reply or Path Reply completes the path



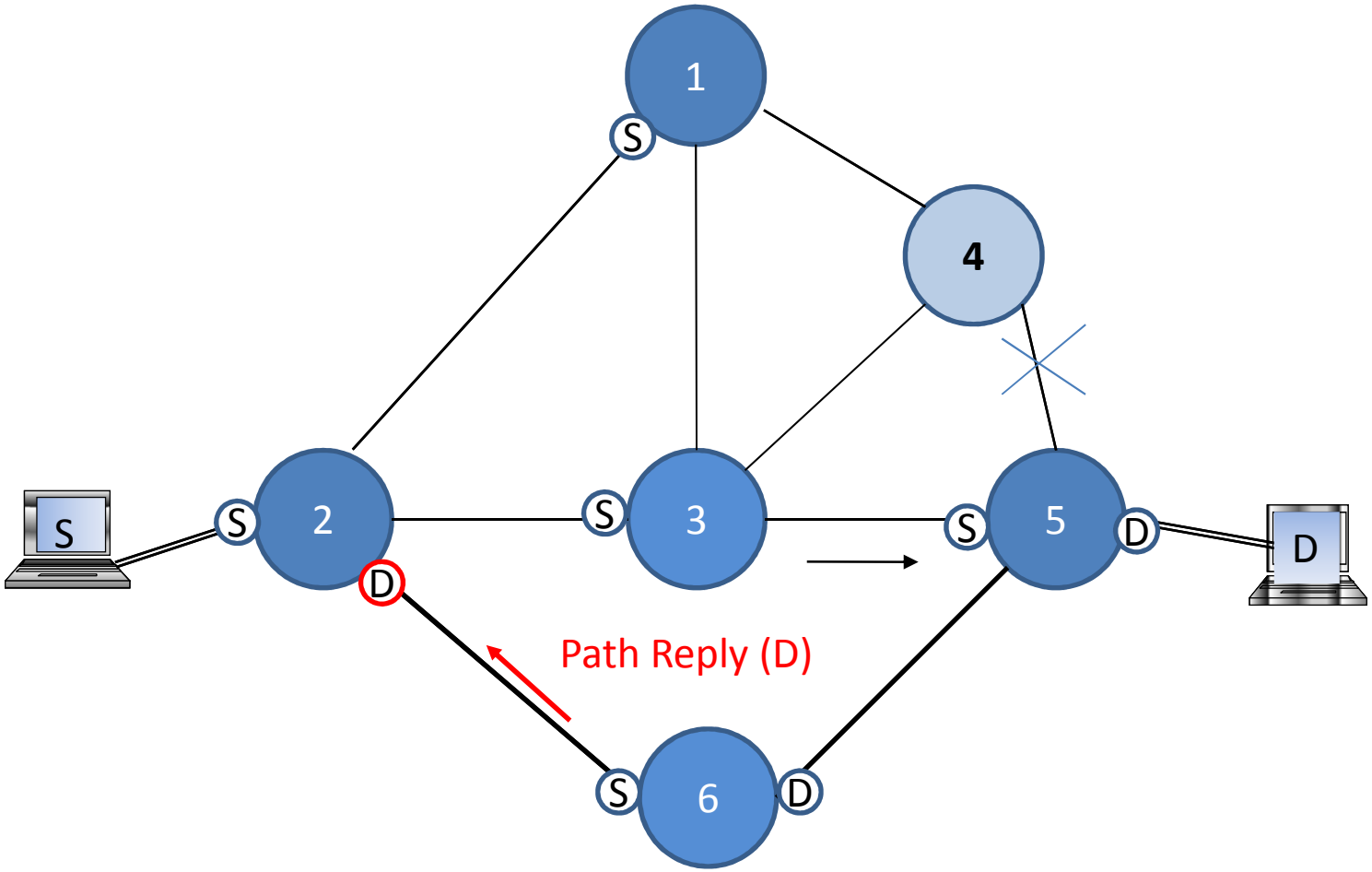
New Path Proxy function

- A bridge that receives a Path Request for a DA that it has associated to a port, different to the receiving port
- Responds with a Path Reply via the receiving port and does not further forward the Request
- It is like the ARP Proxy function, without the IP information, only MAC, and between bridges, iso between hosts.

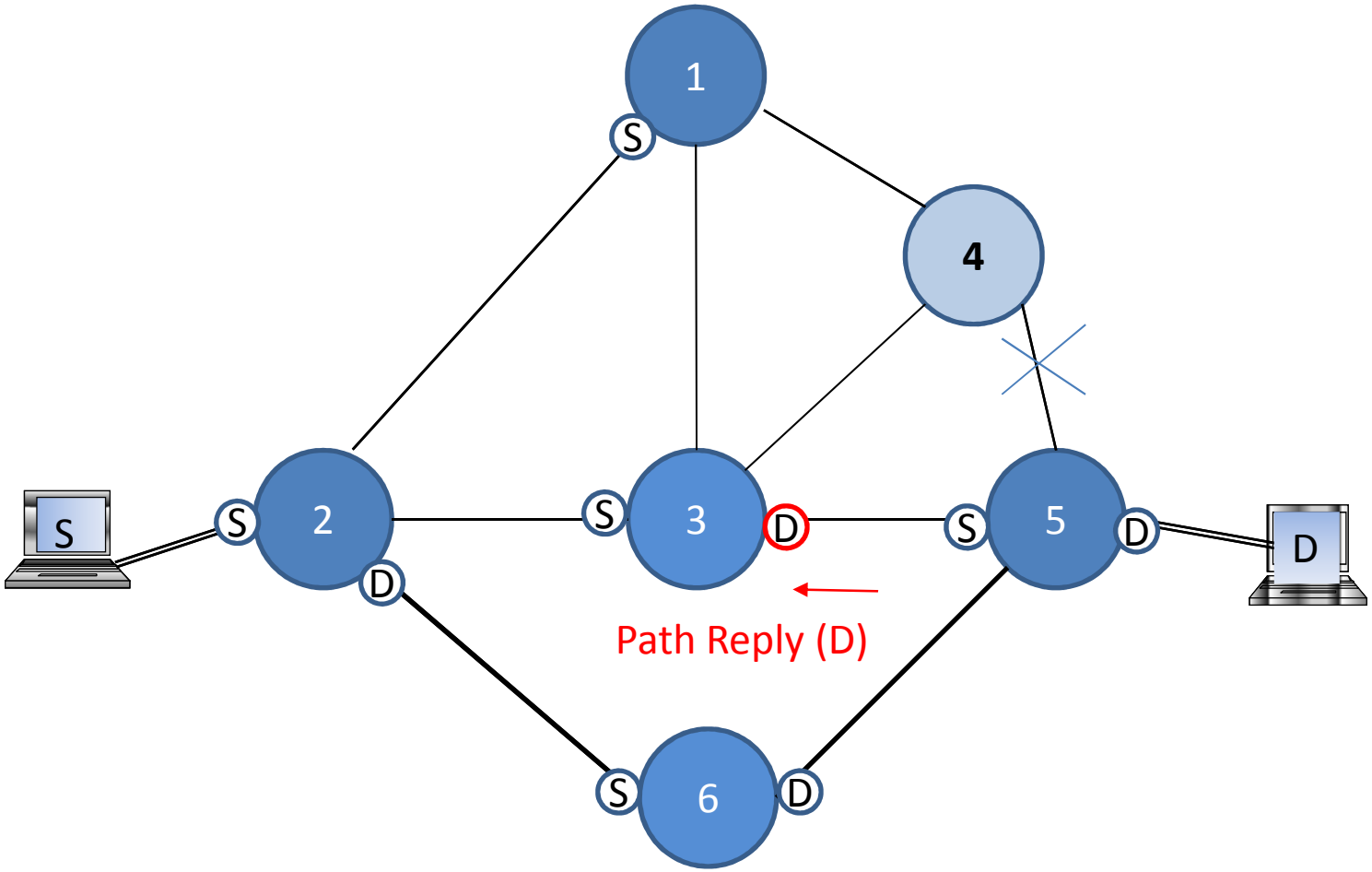
With Path Proxy:
If there is already a path to D at bridge 6 (used by other flows to D)…



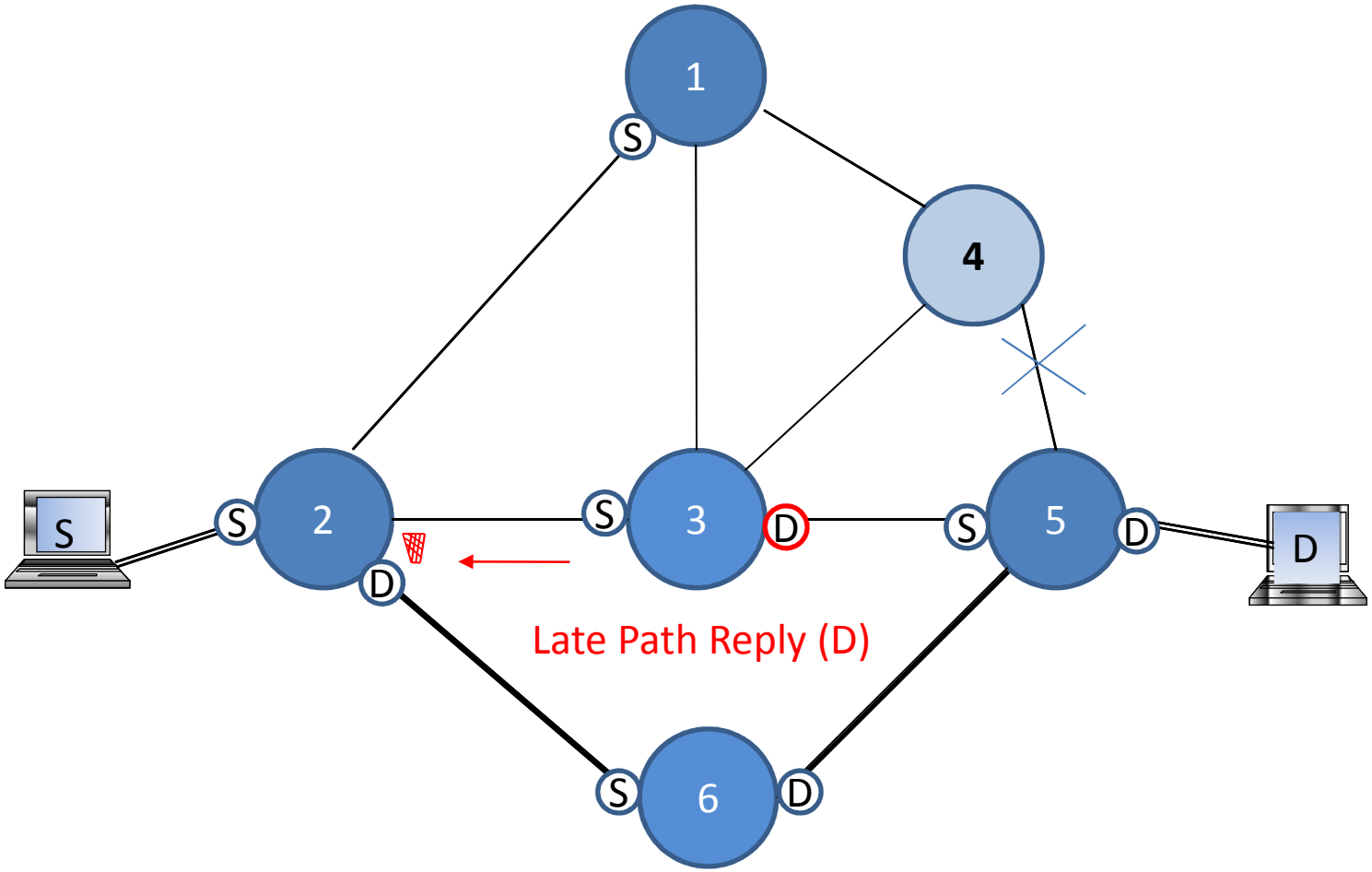
If there is a path to D at bridge 6… Bridge 6 responds with Path Reply



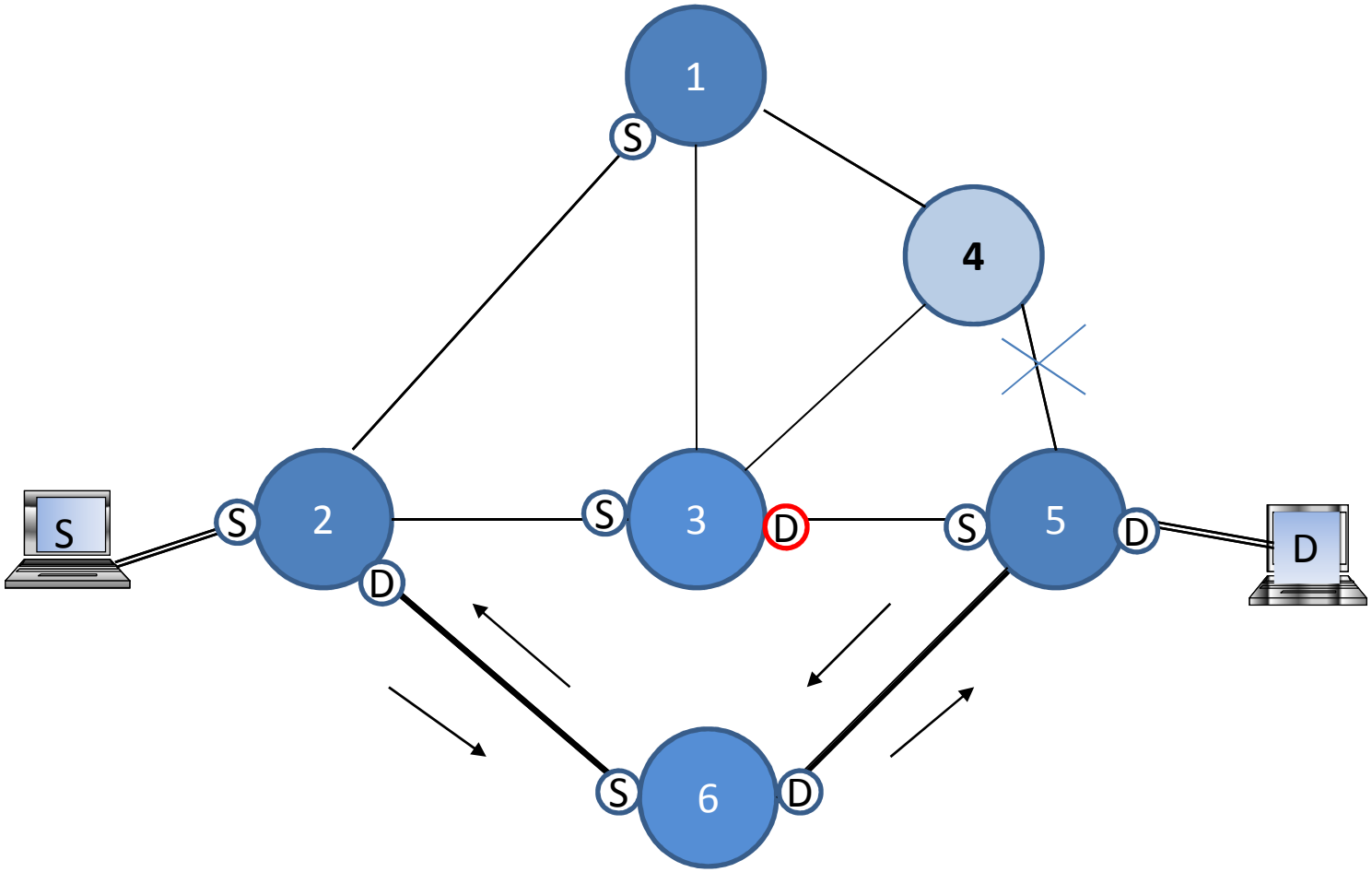
With Path Proxy:
If there is a path to D at bridge 6…



With Path Proxy:
Late Path Replys are discarded at 2



With Path Proxy:
Path repair completed in less hops



Resiliency of path repair to loss of path fail packet

- If first packet looped back is lost, for that destination only, path repair will not take place during repair timer duration.
- Several or all packets may be looped back towards source edge bridge to obtain resiliency.
- Repair table at source edge bridge to prevent further repair attempts for same destination.
- Broadcast frames of basic path repair ensure resiliency if there is no backward connectivity

Loss of Path Fail packet

- Path Fail packet is sent as broadcast to all bridges, it will reach several times each bridge.
  - Path fail is not lost unless there is no topology redundancy

- When Path_fail or Looped back frame is lost, no further repair will be attempted during the repair timer duration period, but consecutive received frames will trigger path repair again.

Improved Path Repair Mechanism

- Loop back frames is part of normal forwarding mechanism:
  - Forward back, flush (unlearn, optional) MAC and set DA to repair in repair table)
  - **No special frames to be created**, frames are just looped back **unmodified**.
  - Requires substantially less processing than Path Fail pkt. Only incremental processing is needed to populate the address at Repair Table at bridge (if used).
  - Repair efforts for flushed MACs of link are distributed among source bridges

Unicast unknown replication vs broadcast Path/ARP Request

- In a bridge receiving unknown unicast frame:
- Unicast frame replication:
  - Serves as path repair in most topologies (not all). Topologies with redundancy on the forward path to destination .
  - Requires processing/discarding of all unicast duplicate frames
  - Additional broadcast load till the bridges learn addresses
- ARP Request or Special Path Repair packets:
  - Most topologies (not all, same reason as for unicast)
  - ARP Request: block/discard like other broadcast packets
  - Special Path Repair packets: Require special process at every bridge to check if DA is directly connected to the bridge.

Replication of unknown unicast frames

- Replication of unknown unicast frames seems an alternative mechanism to Path Repair, closer to the mechanism of Standard Bridges (flood when unknown),
- Unknown unicast frames might be handled in similar way (wrt locking and learning) as ARP Request frames
  - This mechanism can be used in case of link failure to forward the frames till destination. Most topologies will find an alternative path towards destination.
- With Fast-Path, one of the mechanism must be chosen

  -On the fly Replicate of unknown unicasts

  -or loopback and ARP/Path Request

Replication of unknown unicast at source edge bridge (lossless forwarding)

- A flow that is forwarded back encapsulated towards source bridge, might be replicated (unknown unicast frames) at source edge bridge to reach the destination,
- Until new paths are created by issuing gratuitous ARP Request from edge, Path Request or a replicated unicast ARP Reply.

Summary

- Introduction
- Defining the scope: All-Path/ New paradigm
- Recapitulation from Singapore presentation
  - Path repair requirements
  - Improving path repair
- **Profiles/targets of All-Path protocol**
  - Multicast summary
  - Compatibilities with IEEE 802 protocols
  - All-path Hosts, All-path M M-in-M
- Load distribution results

Multicast

- Unicast and multicast routes are congruent in Fast Path and are set up simultaneously.
  - Multicast address lock/learn timers
- Multicast trees are instantly created from source (full flooding plus discard).
- Multicast traffic sources keep active their unicast tree to destination
- Multicast traffic does not need repair (frames create and maintain their tree)
- Compatible with IGMP snooping to optimize (prune) multicast trees.
- Compatibility with MRP to be checked (inputs wellcome)

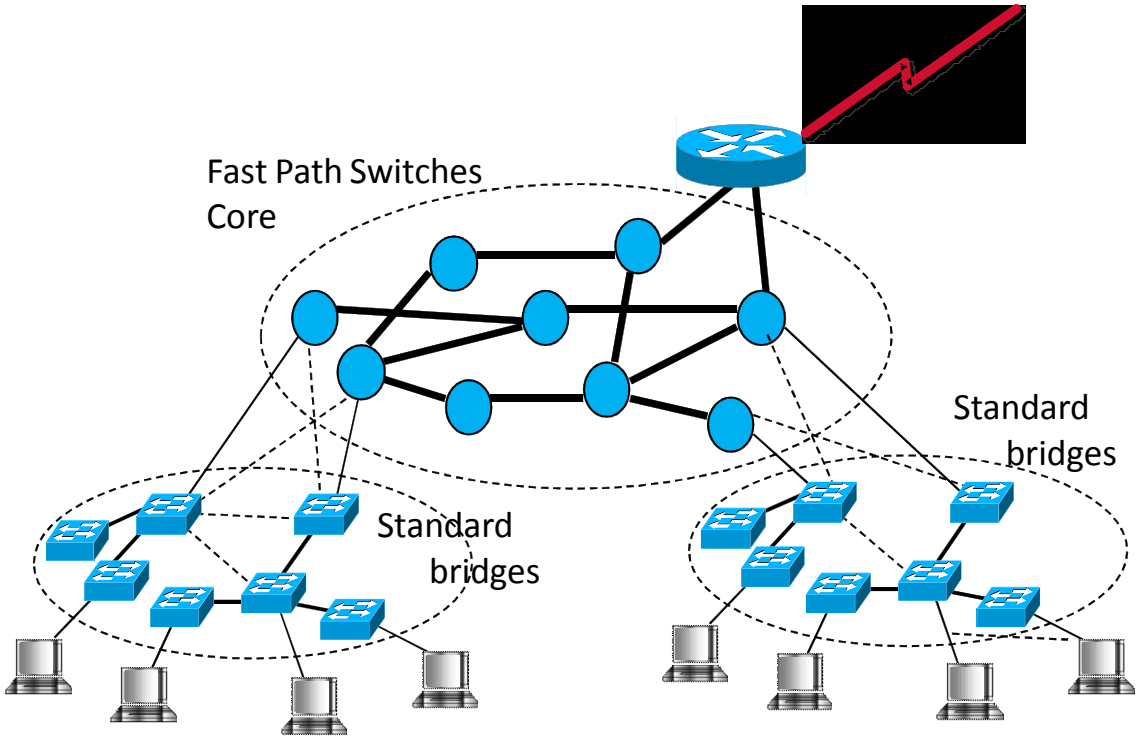
IEEE protocols compatibility

- Core-Island compatibility with 802.1D bridges
  - Compatibility with 802.1Q bridges:
    - plain 802.1Q
    - 802.1aq like (VLAN associated to edge bridge)
  - Further work required to analyze compatibility with (inputs/requirements wellcome):
    - CFM
    - PBB
    - MRP
    - ECMP
    - VEPA

Ideas for Fast Path interworking with 802.1D bridges

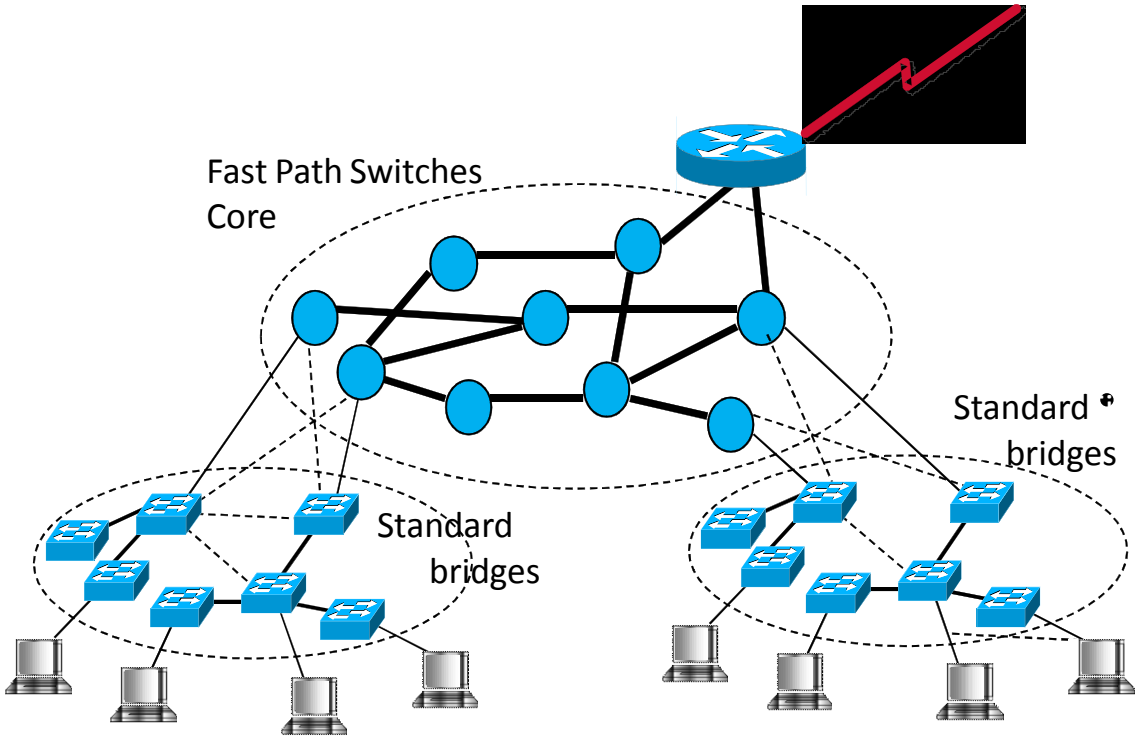
- Base: port protocol migration similar to RSTP/STP
- Bridge operates as All Path if connected to othe All Path bridges
  - Migrates to plain bridge (RSTP) operation if not in the main core
    - And ports connected to Fast Path core are preferred as candidate root ports
- Discovery protocol between All Path bridges creates a core of interconnected All Path bridges (LLDP may help)
  - Continuity and uniqueness of the BP core must be ensured

Compatibility with standard bridges in core-island mode



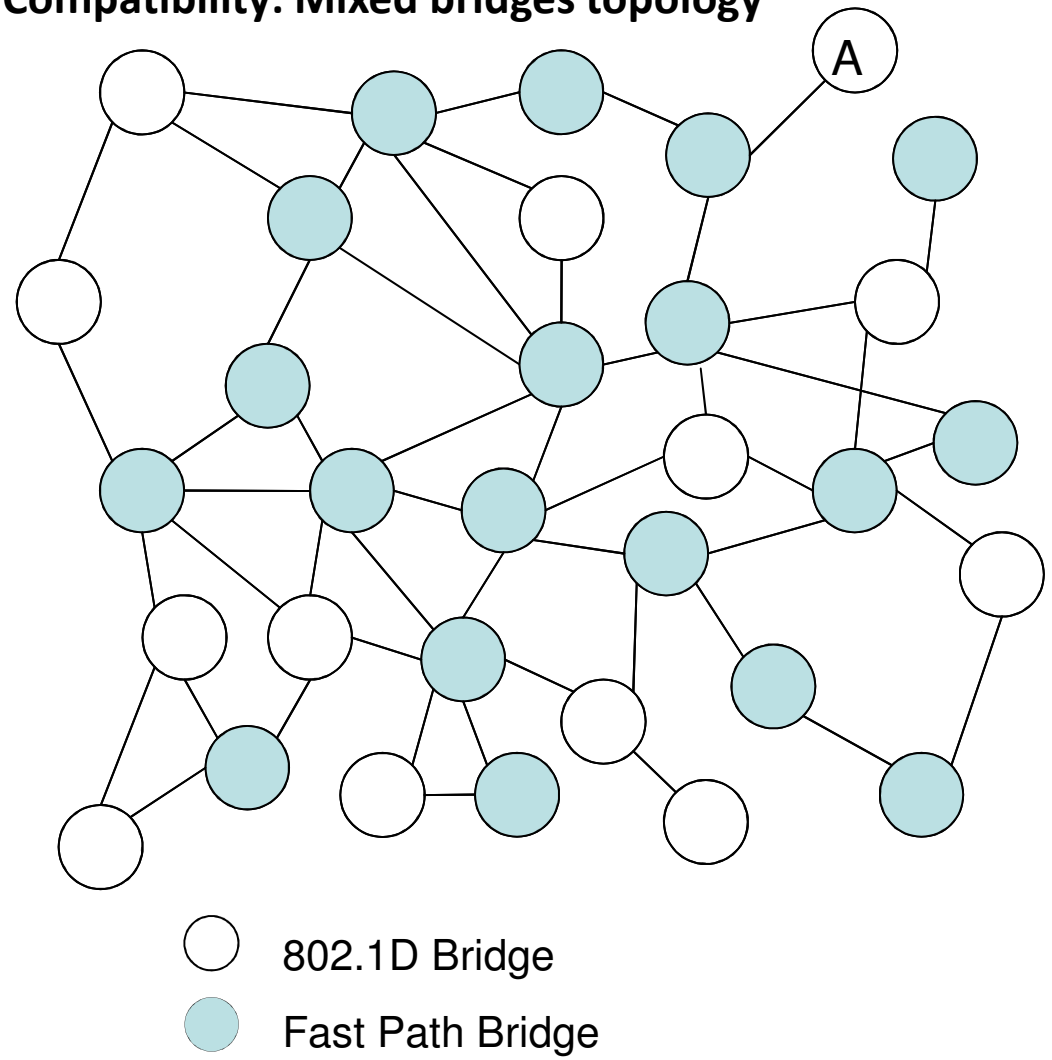
- Fast Path bridges become root of spanning trees of standard bridges (announce a high priority virtual root bridge)
- Islands split in two or more trees if connected with redundant links to core

Compatibility with standard bridges in core-island mode

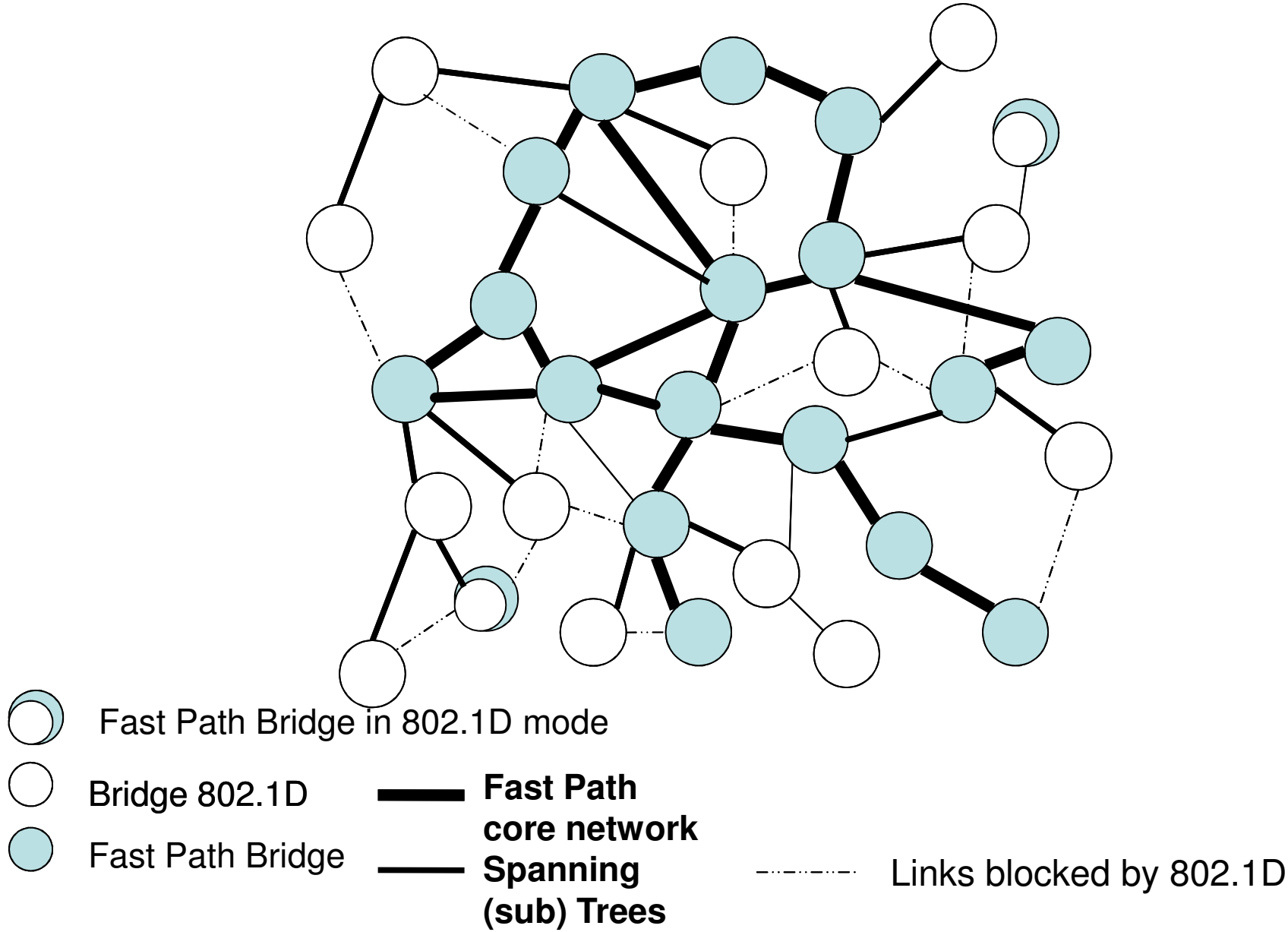


- Every port detects type of connected bridge and selects working mode: STD or Fastpath (via protocol migration mechanisms) depending on its connectivity (i.e. hears FP BPDUs or not)
- LLDP protocol may help

Compatibility: Mixed bridges topology



Compatibility: Active topologies



All Path vs SPBM

- Address locking principle may also be applied to B-MACs
- Backbone bridges may set up trees instanta-neously sending a B Set-Tree multicast frame
- Backbone bridges may confirm with ack new tree link to root bridge at every hop to ensure path symmetry
- Unknown unicast frames (B-MAC) may be replicated
- Tree topology is unpredictable, but resilient

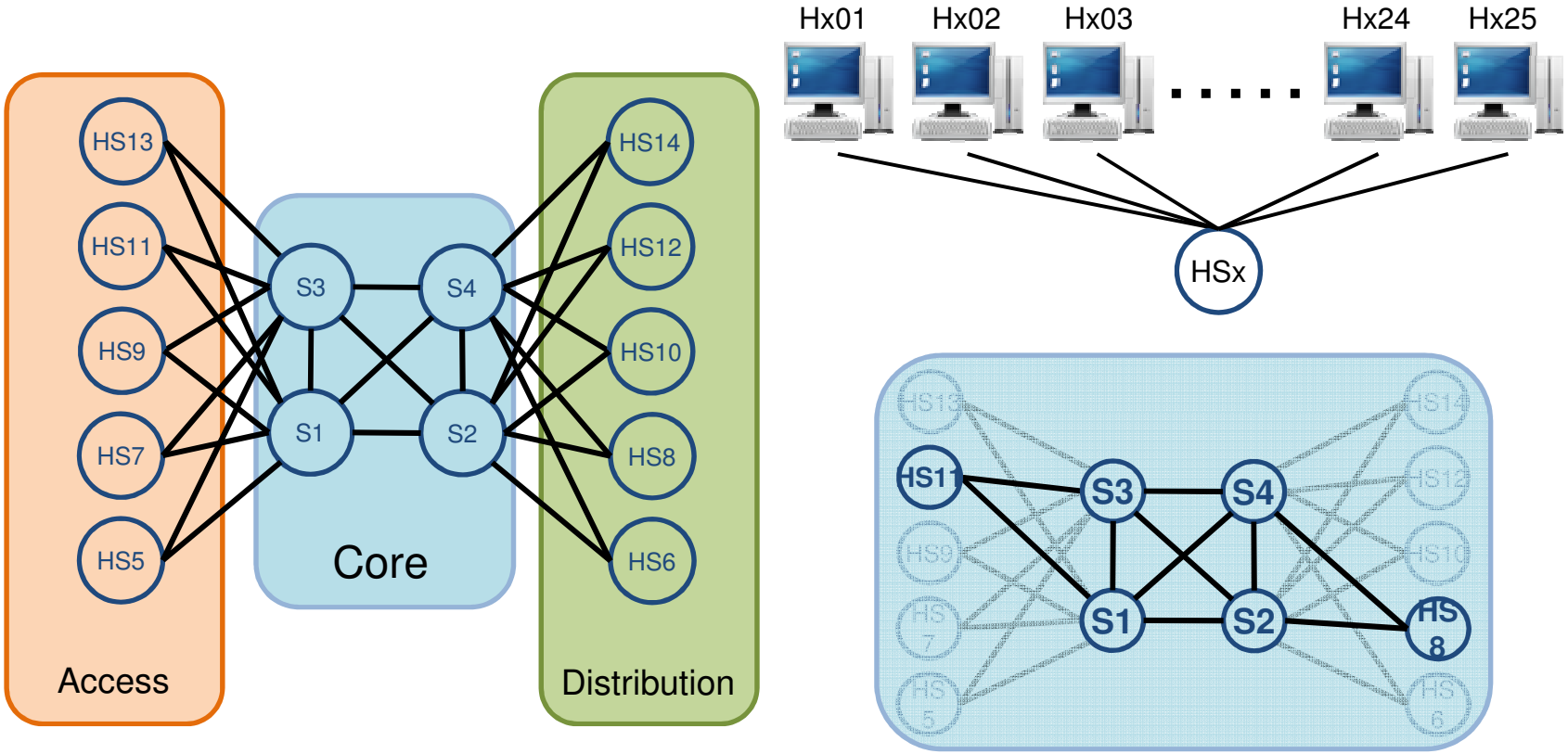
Summary

- Introduction
- Defining the scope: All-Path/ New paradigm
- Recapitulation from Singapore presentation
  - Path repair requirements
  - Improving path repair
- Profiles/targets of All-Path protocol
  - Compatibilities with IEEE 802 protocols
  - All-path Hosts, All-path M M-in-M
- **Load distribution results**

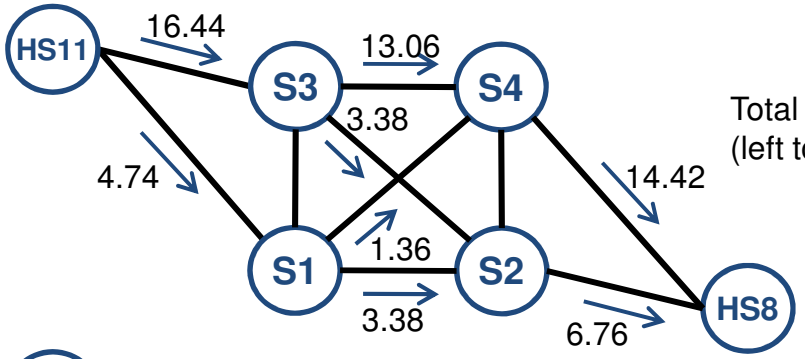
Load distribution

- Path diversity is a native feature of ARP-Path: variations in latency produce variations in the paths set up from hosts connected to the same edge bridge with host connected with another edge bridge at different times.
- Data center topology with alternative paths and increasing loads shows well the load distribution.
- Coarse load distribution at low loads
- Finer with high loads.

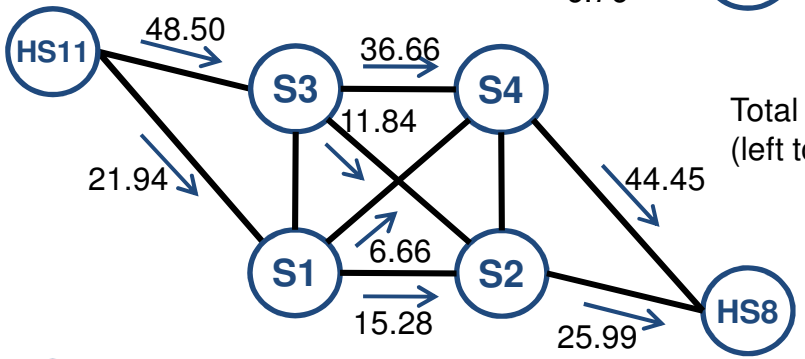
Load distribution. Path diversity



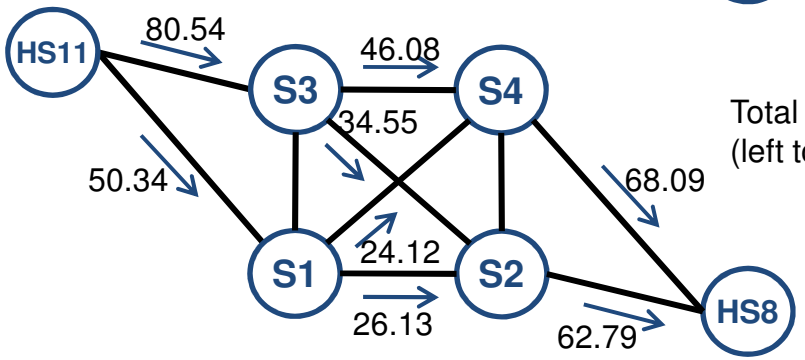
Traffic from 25 hosts connected to HS11 to 25 hosts connected to HS8. UDP of increasing length to force link loads333. All links 100 Mbps. Omnet++ simulator.



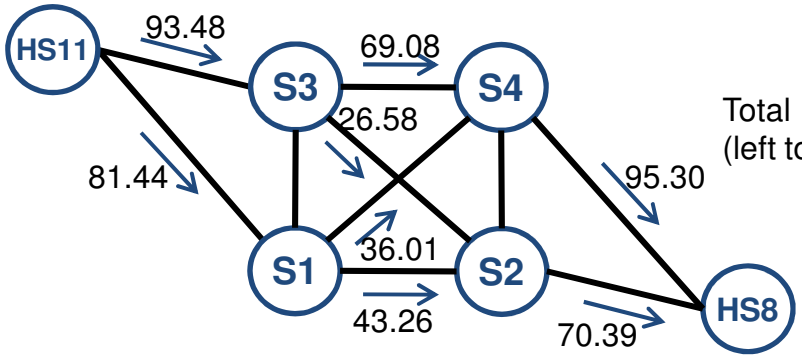
Total average traffic
(left to right): **21.18 Mbps**



Total average traffic
(left to right): **70.44 Mbps**



Total average traffic
(left to right): **130.88 Mbps**



Total average traffic
(left to right): **174.92 Mbps**

Conclusions

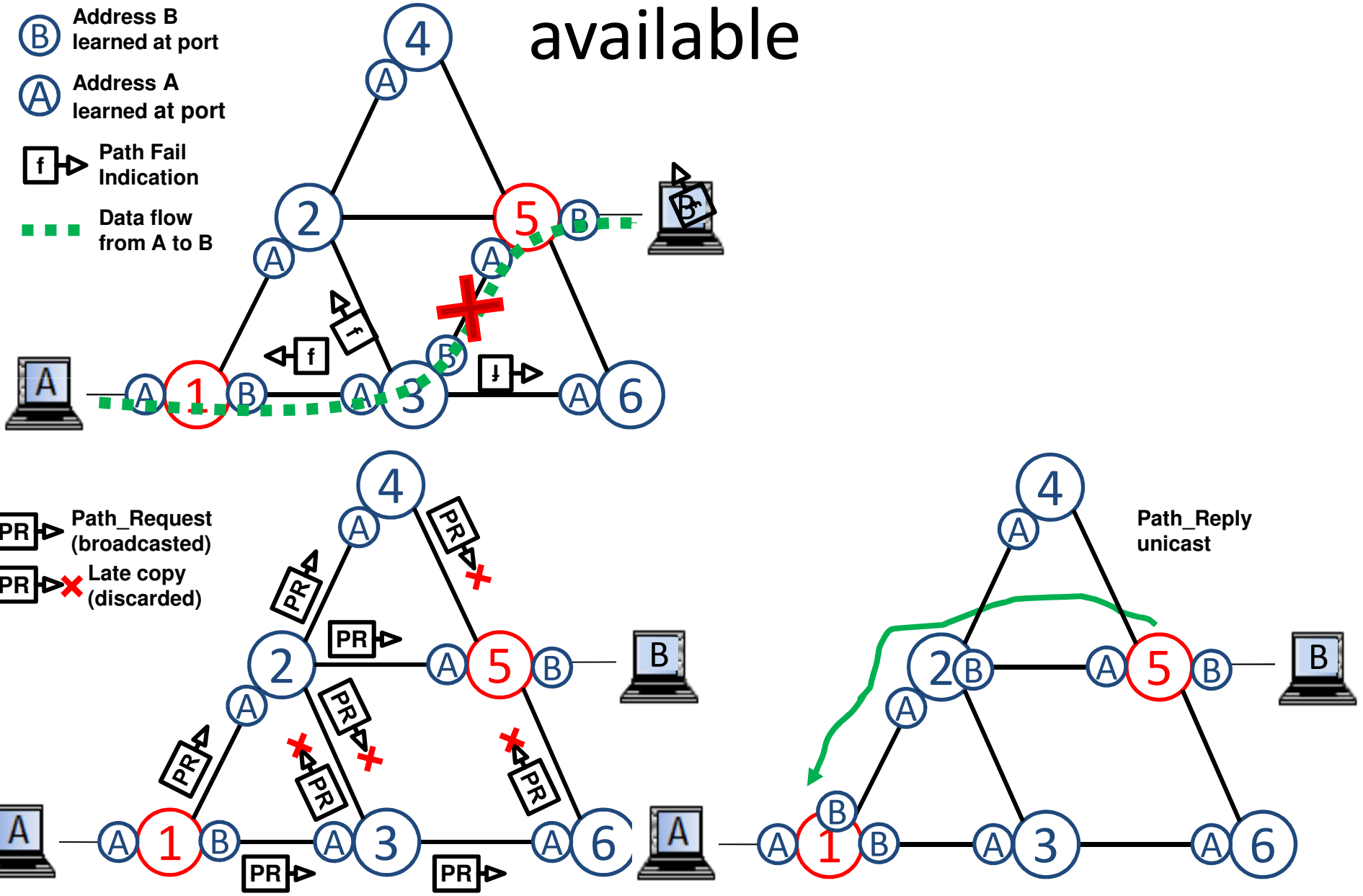
- Processing effort for Fast-Path on-demand path repair is moderate
- Mechanisms like **reverse forwarding** and **paths proxy** reduce repair processing load per node and speed up repair
- Unknown unicast replication as protocol variant.
- Scope and requirements to be agreed.
  - Suggested as 802.1D evolution.
  - Suitability to SPBM or SPBV only if it provides extra benefits
- Inputs wellcome. Any form of collaboration, like reviewing documents, will help.
- Good results in load distribution at medium and high loads.

Back up slides

Alternate (=old) path repair

- The repair mechanism consists of 2 special messages. First, a "*Path_Fail*" message that will be transmitted back to the bridge serving the host that originated the unicast frame. Second, a "*Path_Request*" message which will lock ports analogously to the ARP Request step, to start creating the new alternative path. A "*Path_Reply*" message, analogue to the ARP Reply will confirm the path

Path Repair when no backward path is available



Replication of unicast frames withouth MAC learning

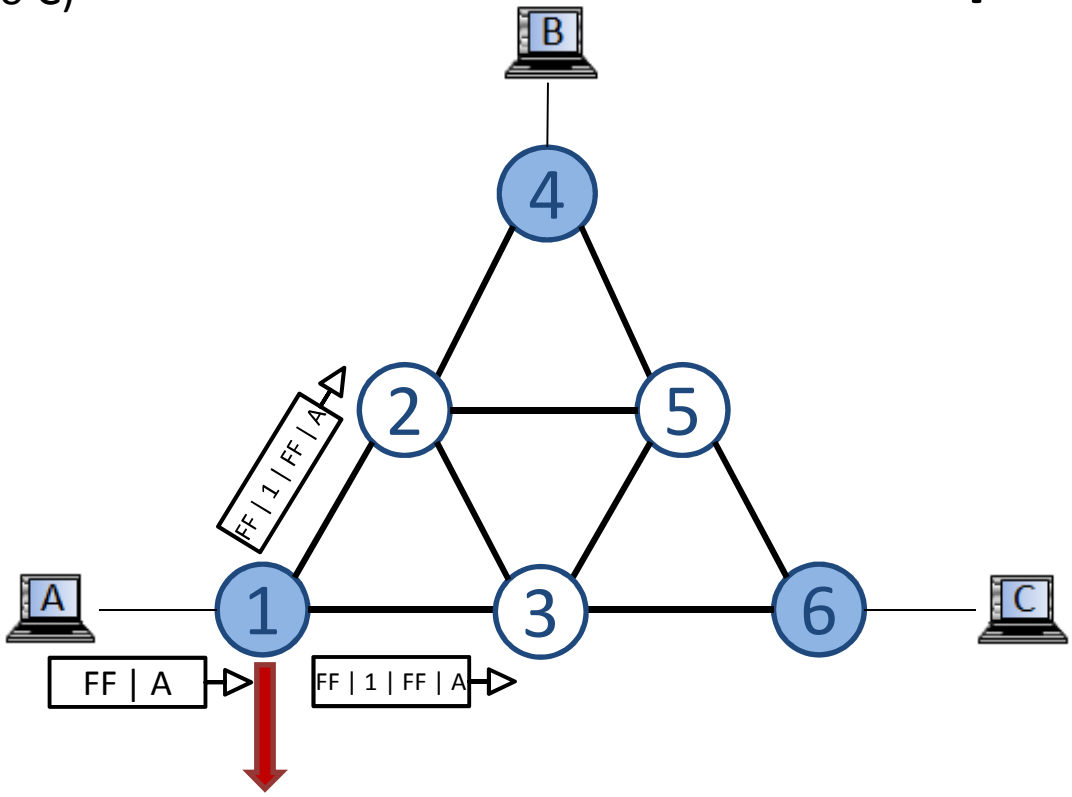
- This would serve frames without path to reach destination, but the path to destination is not built
- If there is no MAC learning with unicasts, the return path is not built, then the process will repeat in the opposite direction
- No paths are built
- A mechanism to trigger repair from destination are needed. This can be the backward Path Fail

MAC in MAC (SPBM)

- Example of ARP with MAC in MAC shown
- Special Backbone multicast packets can be used to encapsulate ARP Request and ARP Reply in the backbone

ARP Request
(from A to C)

Mac in MAC example



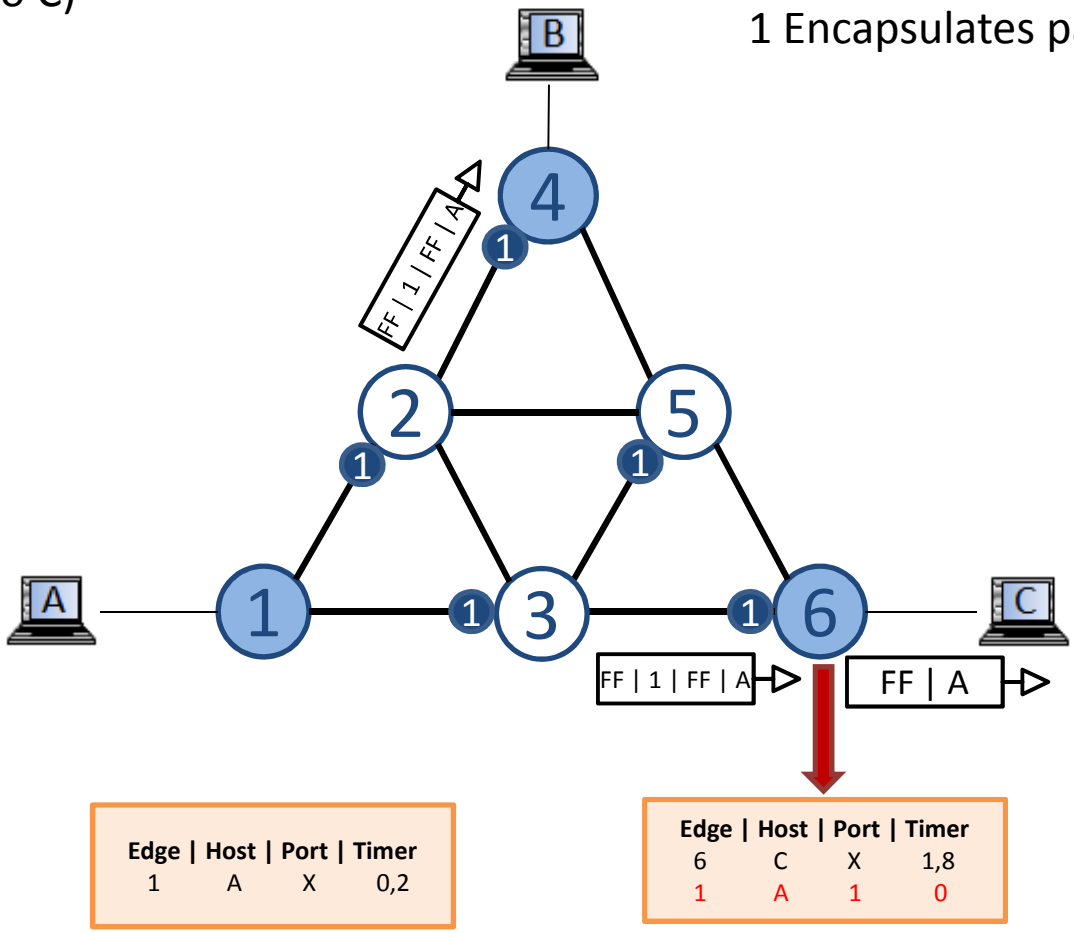
'locked' port, that later will become 'learnt'
(it means a table entry: **mac | port | timer**)

Ethernet frame: **dst mac | src mac | data**

Table

Translation table
(entry: **edge mac| host mac| port | timer**)

ARP Request
(from A to C)

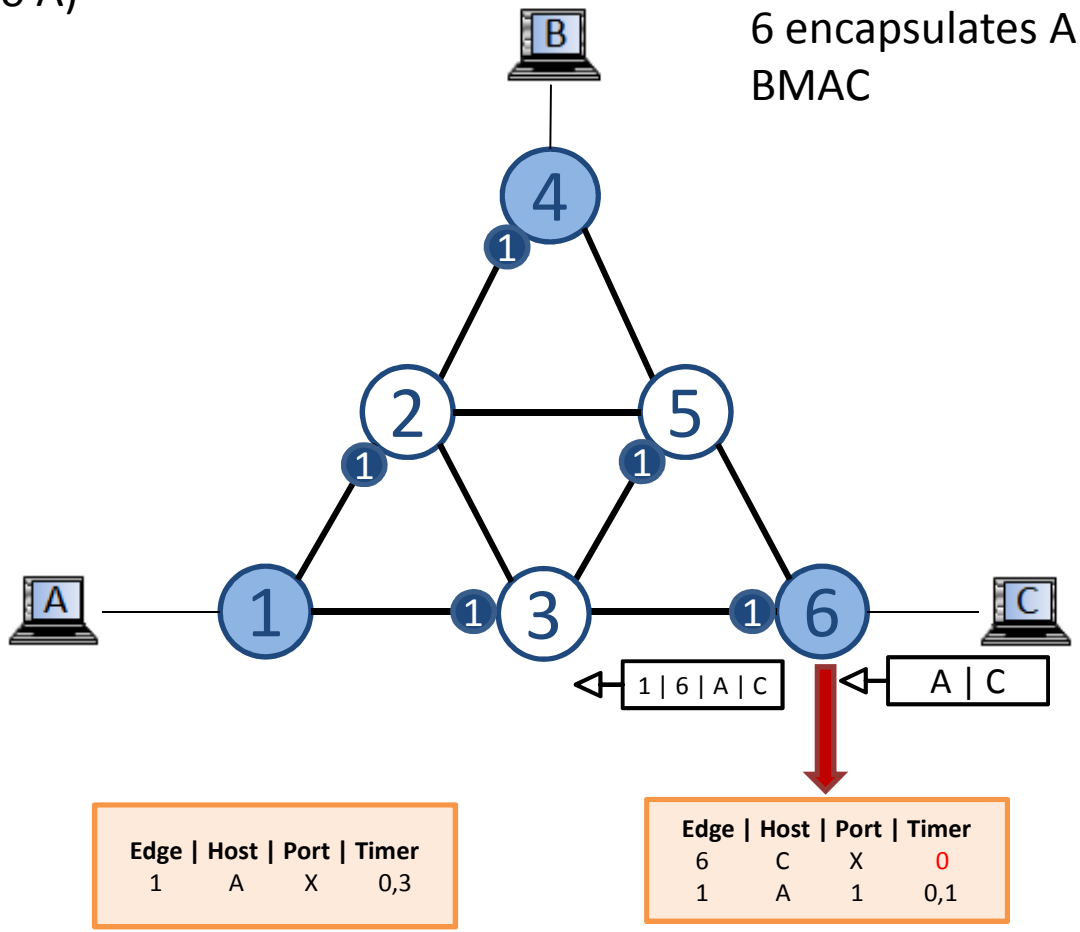


'locked' port, that later will become 'learnt'
(it means a table entry: **mac | port | timer**)

Ethernet frame: **dst mac | src mac | data**

Table Translation table
(entry: **edge mac| host mac| port | timer**)

ARP Reply ‘ucast’
(from C to A)



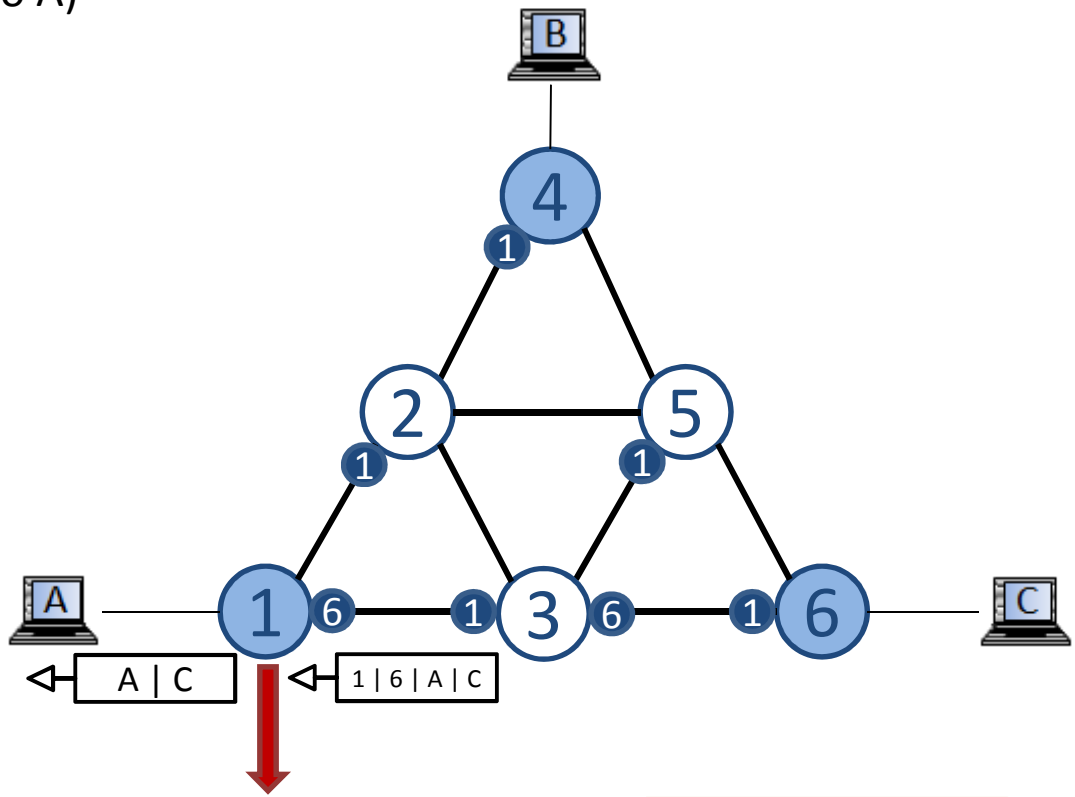
6 encapsulates ARP Reply with its BMAC

‘locked’ port, that later will become ‘learnt’
(it means a table entry: **mac | port | timer**)

Ethernet frame: **dst mac | src mac | data**

Table Translation table
(entry: **edge mac| host mac| port | timer**)

ARP Reply ‘ucast’
(from C to A)

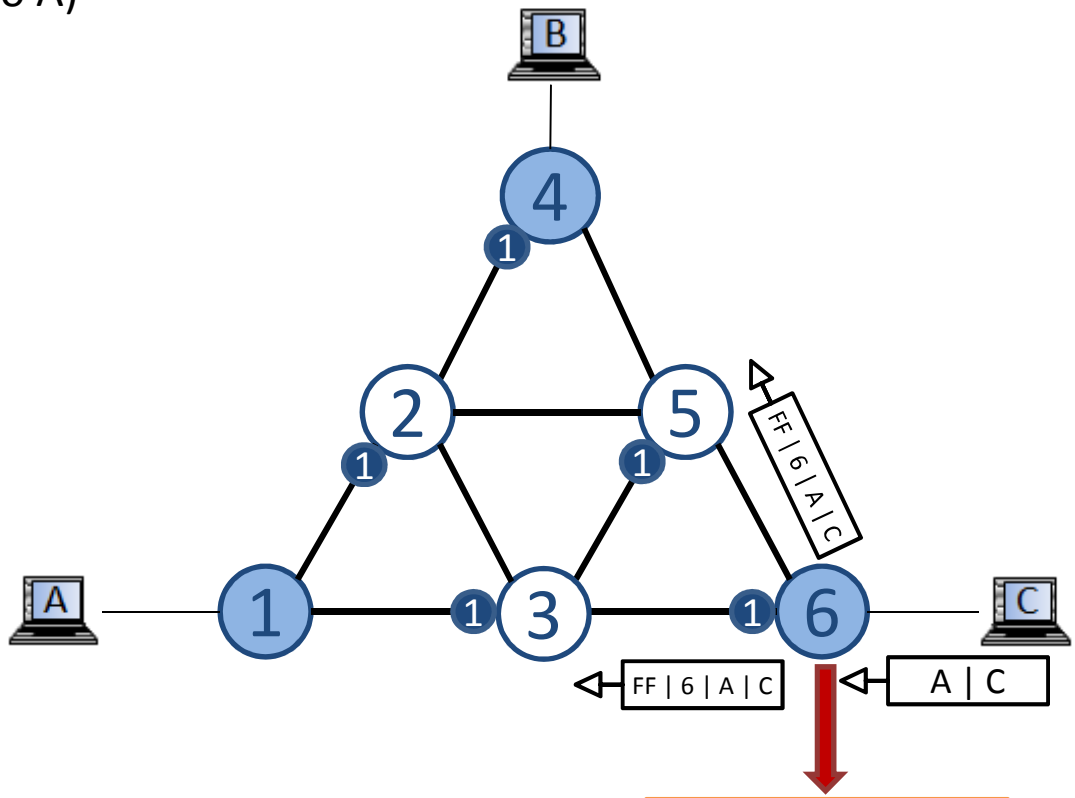


‘locked’ port, that later will become ‘learnt’
(it means a table entry: **mac | port | timer**)

Ethernet frame: **dst mac | src mac | data**

Table Translation table
(entry: **edge mac| host mac| port | timer**)

ARP Reply ‘bcast’
(from C to A)



Edge	Host	Port	Timer
1	A	X	0,3

Edge	Host	Port	Timer
6	C	X	0
1	A	1	0,1

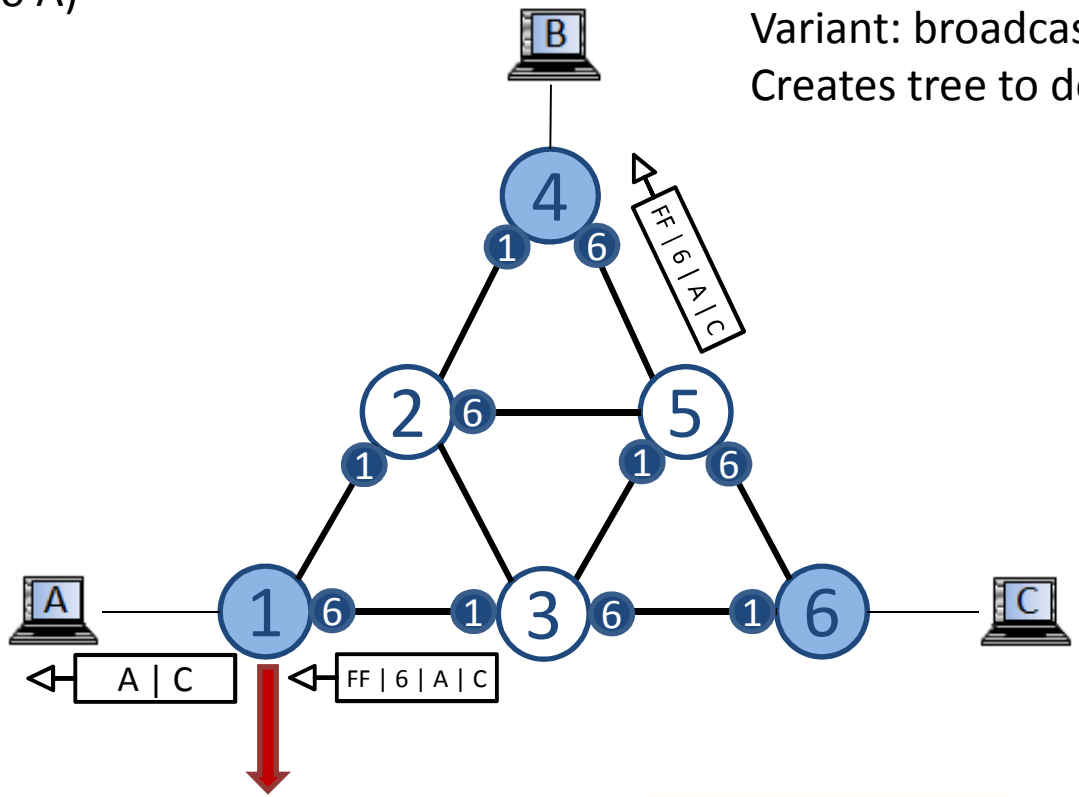
‘locked’ port, that later will become ‘learnt’
(it means a table entry: **mac | port | timer**)

Ethernet frame: **dst mac | src mac | data**

Table Translation table
(entry: **edge mac| host mac| port | timer**)

ARP Reply ‘bcast’
(from C to A)

Variant: broadcast of reply packet
Creates tree to destination



‘locked’ port, that later will become ‘learnt’
(it means a table entry: **mac | port | timer**)

Ethernet frame: **dst mac | src mac | data**

Table Translation table
(entry: **edge mac| host mac| port | timer**)