Fast Path Ethernet Switching

Revisiting Backward Learning Switches

LANMAN Workshop 7th May 2010

Guillermo Ibanez, J.A. Carral, Alberto García-Martínez (UC3M)
José M. Arco, Diego Rivera, Arturo Azcorra (UC3M
GIST research group. Telematic Eng. Area. Dpt. Automatica.
Universidad de Alcalá (Madrid). Projects: MEDIANET, EMARECE
guillermo.ibanez@uah.es

Universidad de Alcalá
Dpto. de Automática
Área de Ingeniería Telemática

UNION EUROPEA
FONDO SOCIAL EUROPEO

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

Junta de Comunidades de
Castilla-La Mancha

- Introduction (the routing bridges problem)
- Path Set Up in Fast Path Switches
- Path Repair
- Linux Proof of Concept
- Simulation results
- Compatibility with standard bridges
- Conclusions

Evolution of Transparent Switches

- Standardized:
  - From Spanning Tree Protocol to Rapid STP and Multiple STP (MSTP).
- Lack of scalability and ineficiency of bridges with spanning tree
  - Limitation of spanning tree (RSTP) protocols: size, links blocked, paths are not shortest paths
  - Complexities of configuring and scaling MSTP
- Burden of administering IP addresses in campus networks
  - Increasing importance of using a single IP subnet in the campus or datacenter
  - Server/network virtualization increases its importance
- **NEED of a self configuring single IP subnet network campus architecture**
- Proposals under standardization:
  - Shortest Path Bridges (IEEE 802.1aq). 2005.
  - Routing Bridges (TRILL, IETF). 2004.
  - *Both use link state routing IS-IS to build rooted trees and path computation*
- Other*: Seattle. Uses also link-state routing.*

Motivation

- Shortest path bridges and Routing bridges are conceptually **hybrid** devices (bridge + layer 2 router): Combination of transparent bridge and link state routing protocol (IS-IS) is used to build trees and find routes.
- Drawbacks: Computational complexity, host lists interchange between bridges, and need of a synchronizing mechanism to prevent frame loops.
- Have all the possibilities of backward learning bein explored on transparent switches?
- ***Our statement: Transparent bridges with backward learning may achieve shortest path if they use full topology***
  - ***without spanning tree***
  - ***without link state routing.***

Transparent bridges with shortest paths How?

- *By enhancing the transparent bridge mechanisms of forwarding, learning and filtering.*

1.- Relax the restrictions of diffusion of frames to a tree topology:

  - *Enable frame difusion and learning over all links : Fast path. (Find path by ARP broadcast flood)*

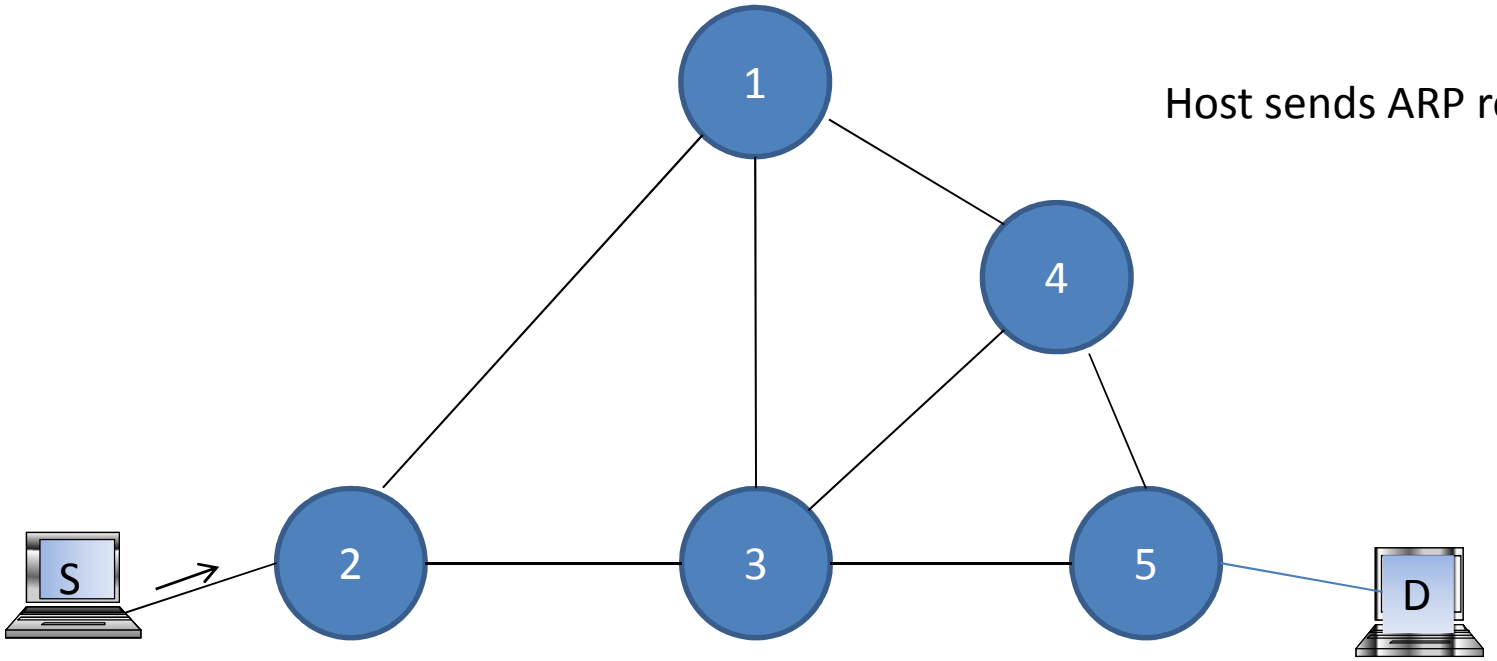
2.-But we need an effective frame loop prevention mechanism*:*

  – *Discard all broadcast frames not received via port associated to source (like IP Reverse Path Forwarding Multicast)*

Fast path basics

- Establish unicast paths just by controlled flooding of a broadcasted frame: ARP Request.
- A temporary tree is established towards the source by learning-locking the source address to the input ports that receive it first
- The path is then confirmed as a bidirectional, symmetric path, after reception of the unicast ARP reply frame from destination host .

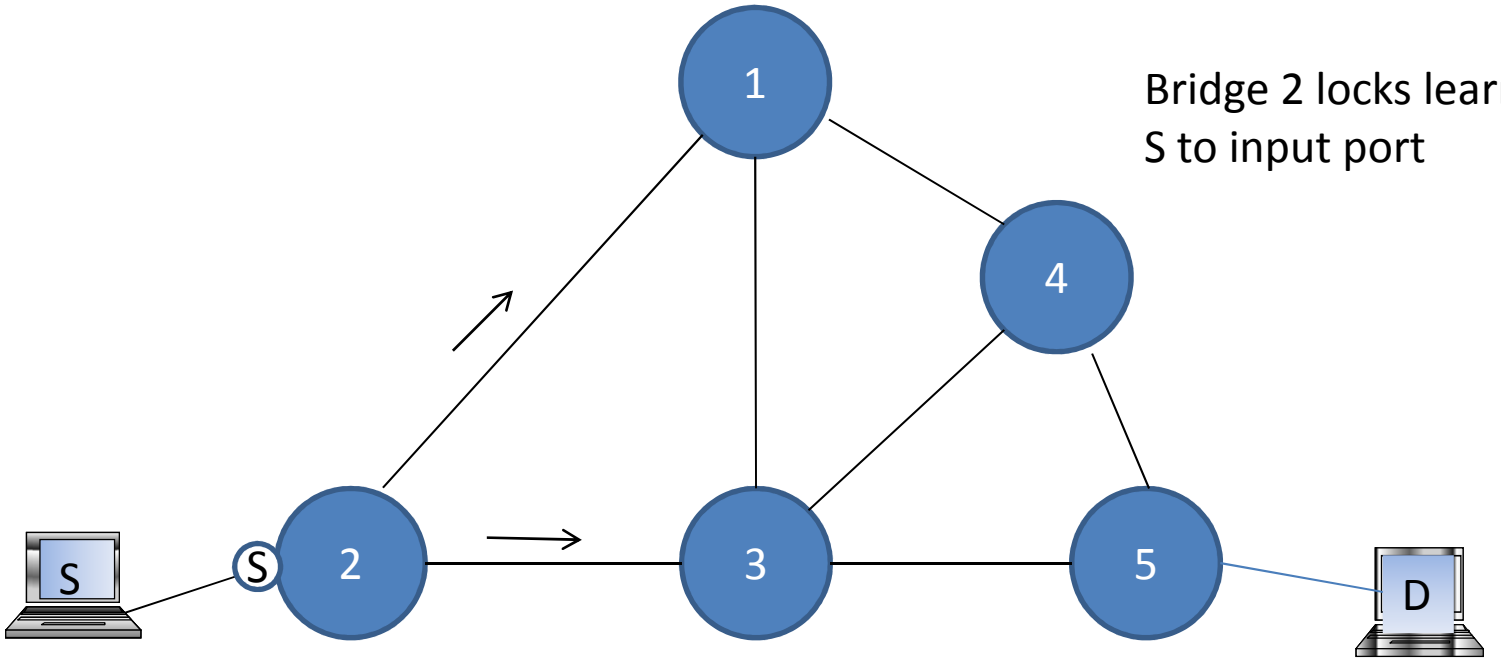
Path set up by host-I
ARP Request



Host sends ARP request frame

ARP Request (broadcasted)

Path set up host-II. ARP is flooded

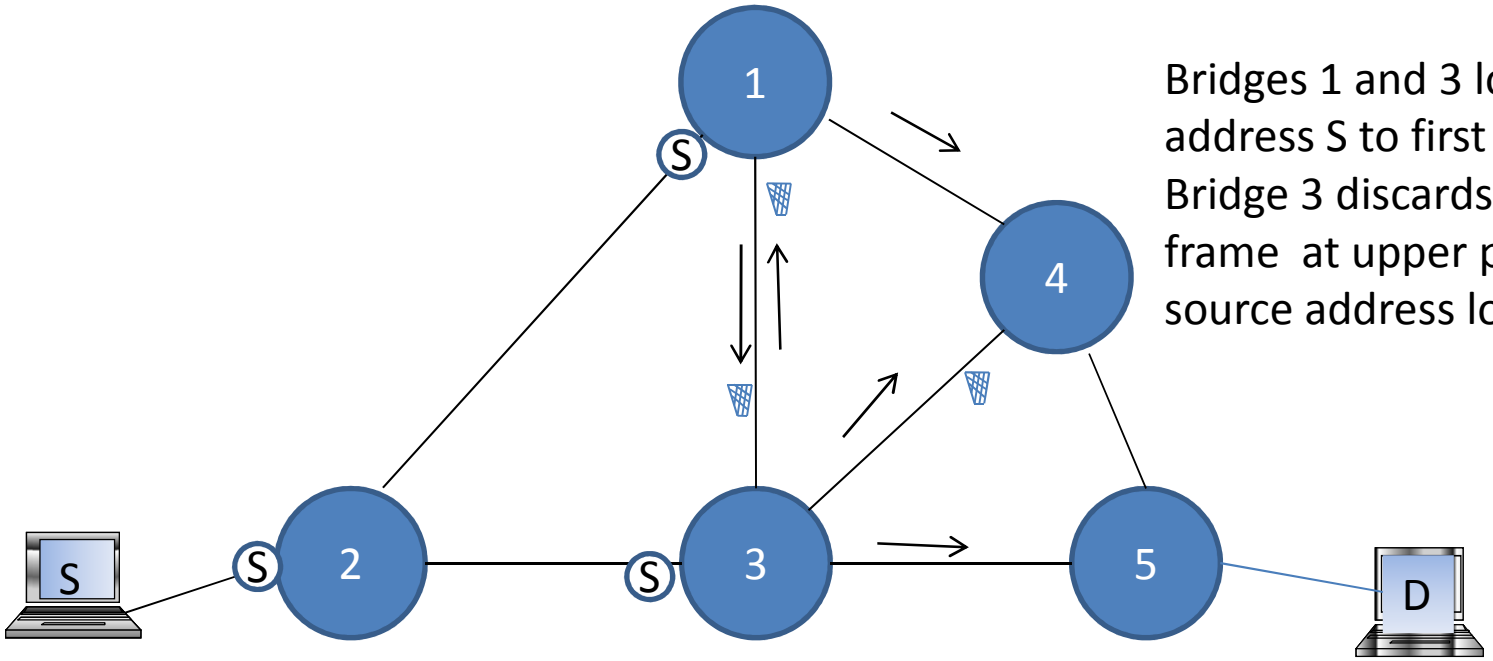


Bridge 2 locks learning of address S to input port

S *Port locked to S*

→ *ARP Request (broadcasted)*

Path set up by host-III. ARP propagates



Bridges 1 and 3 lock learning of address S to first input (left) port
Bridge 3 discards late (duplicate) frame at upper port because source address locked

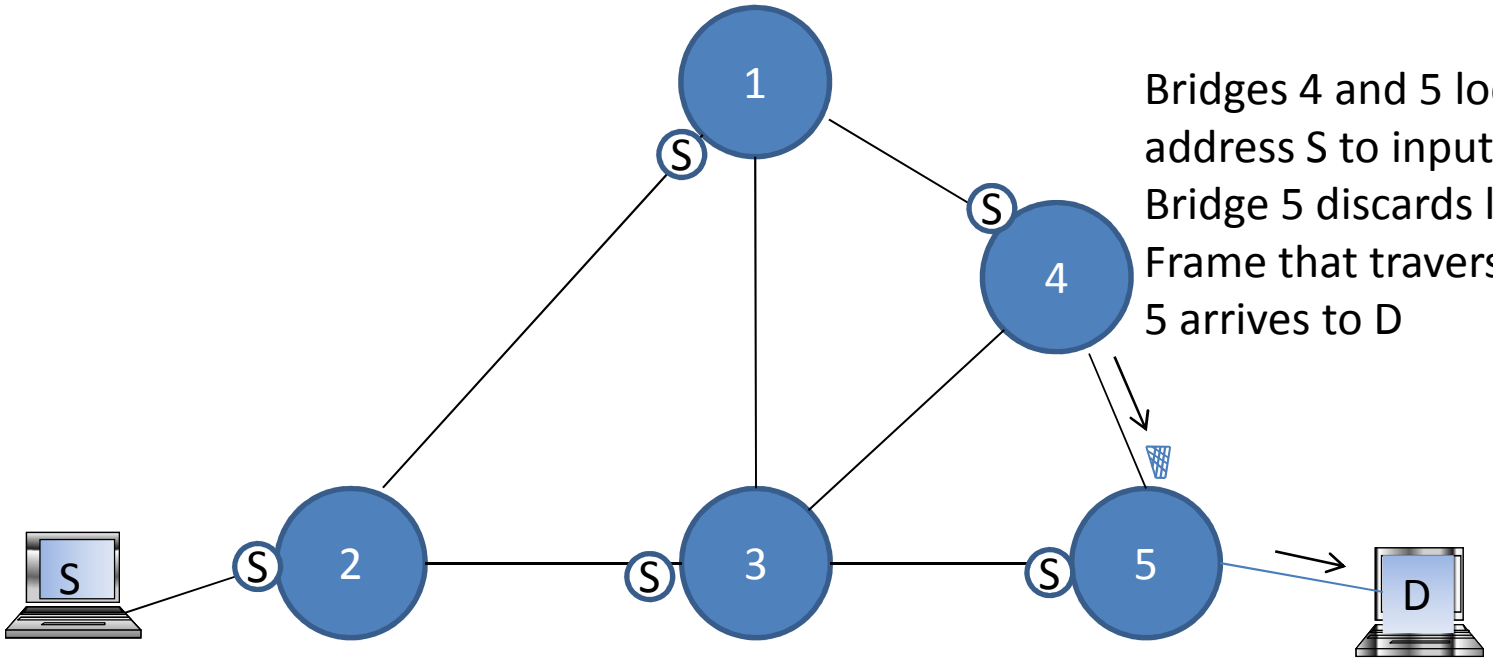
S *Port locked to S*

D *Port locked to D*

ARP (path) request (broadcasted)

Late frame discarded

Path set up by host-IV
ARP Request reaches destination



Bridges 4 and 5 lock learning of address S to input port
Bridge 5 discards late frame
Frame that traversed bridges 2-3-5 arrives to D

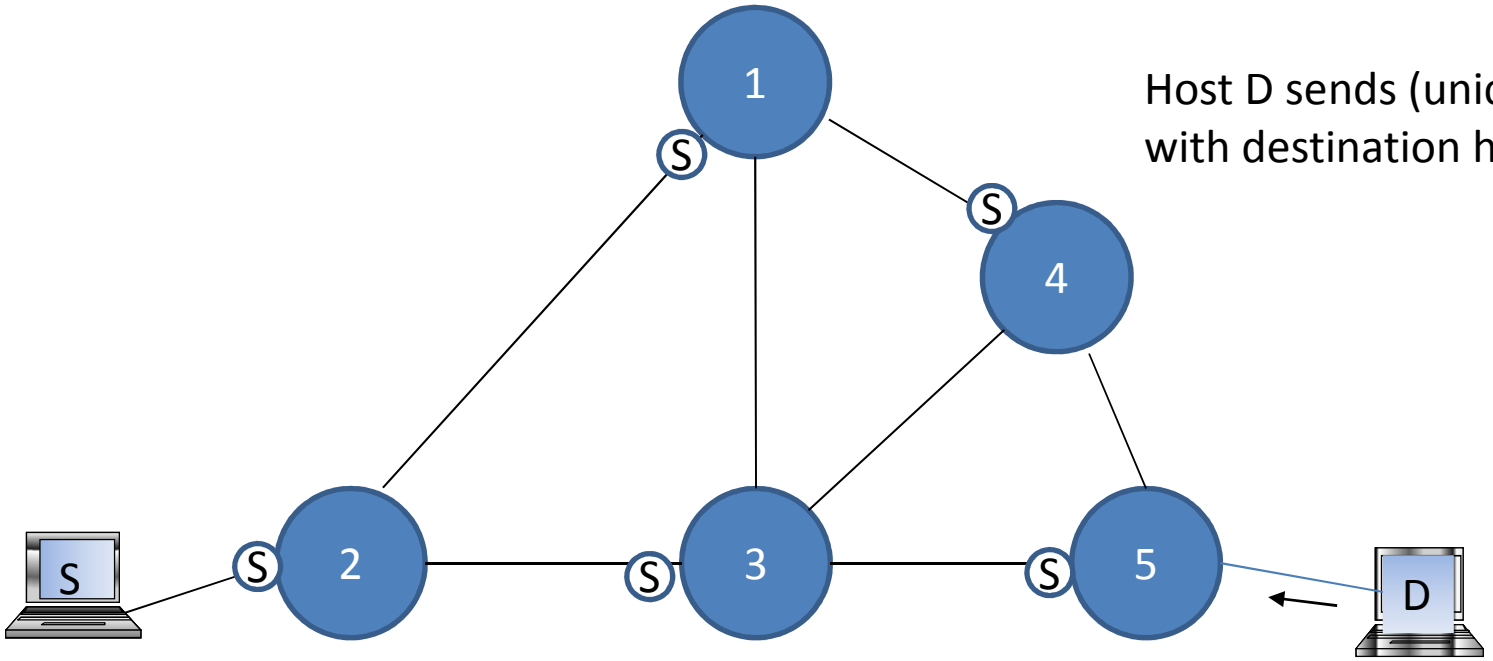
S *Port locked to S*

D *Port locked to D*

→ *ARP (path) request (broadcasted)*

← *ARP (path) reply (confirm) (unicast)*

Path set up by host-V.
ARP Reply (unicast)



Host D sends (unicast) ARP Reply with destination host S

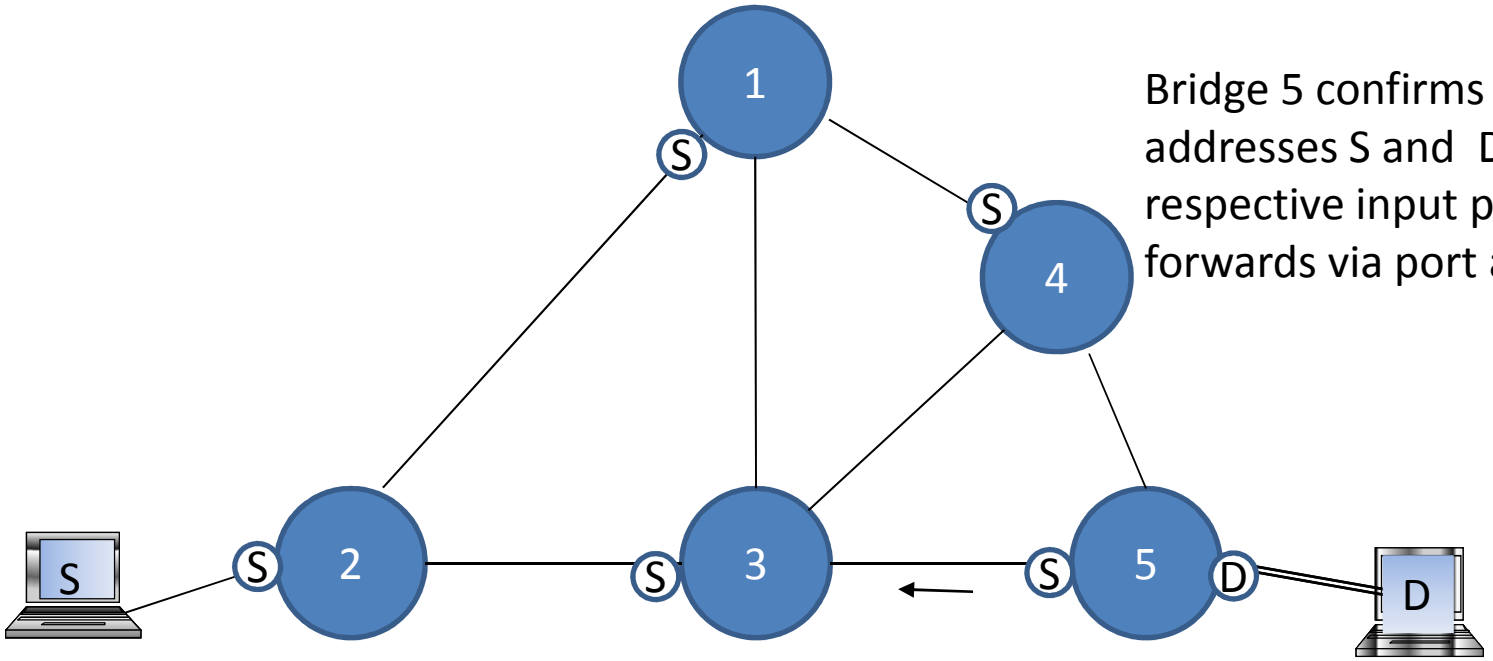
S *Port locked to S*

D *Port locked to D*

→ *ARP (path) request (broadcasted)*

← *ARP (path) reply (confirm) (unicast)*

Operation (path set up by host-VI)



Bridge 5 confirms learning of addresses S and D associated to respective input ports and forwards via port associated to S

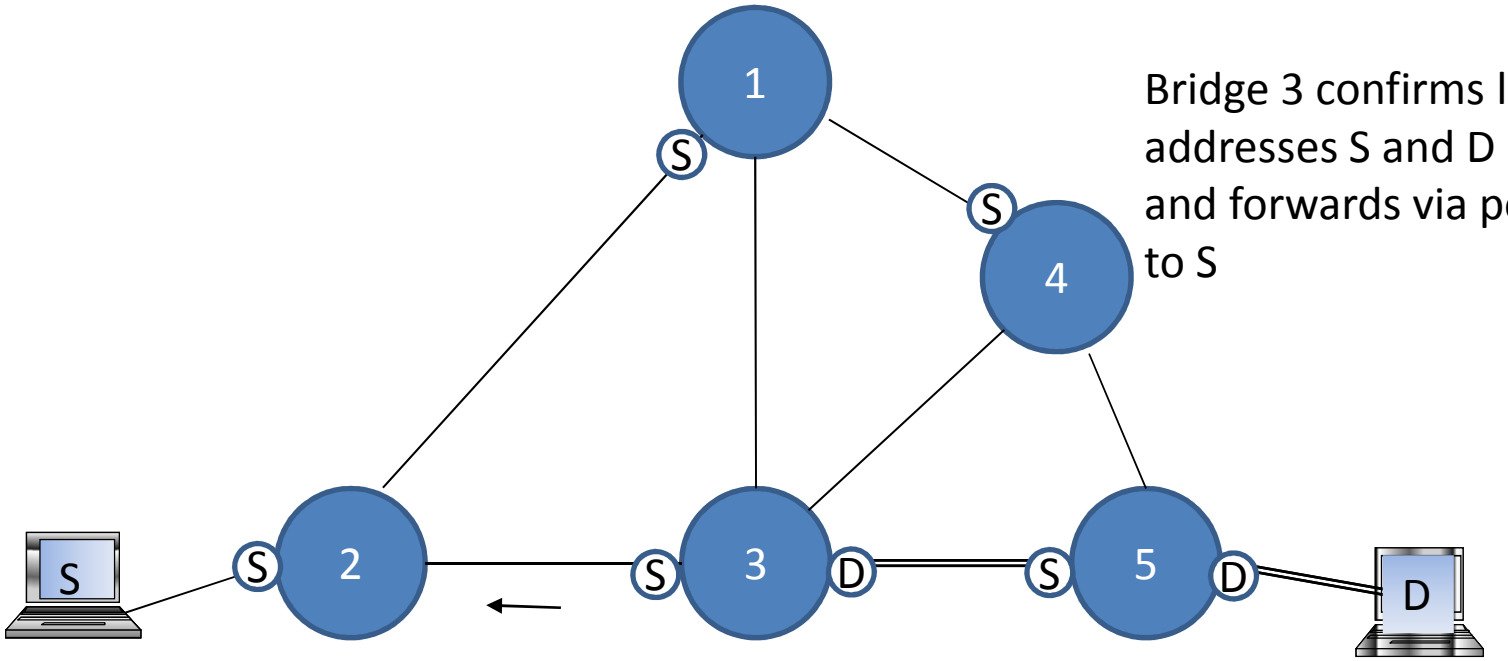
S *Port locked to S*

D *Port locked to D*

→ *ARP (path) request (broadcasted)*

← *ARP (path) reply (confirm) (unicast)*

Operation (path set up by host-VII)



Bridge 3 confirms learning of addresses S and D at input ports and forwards via port associated to S

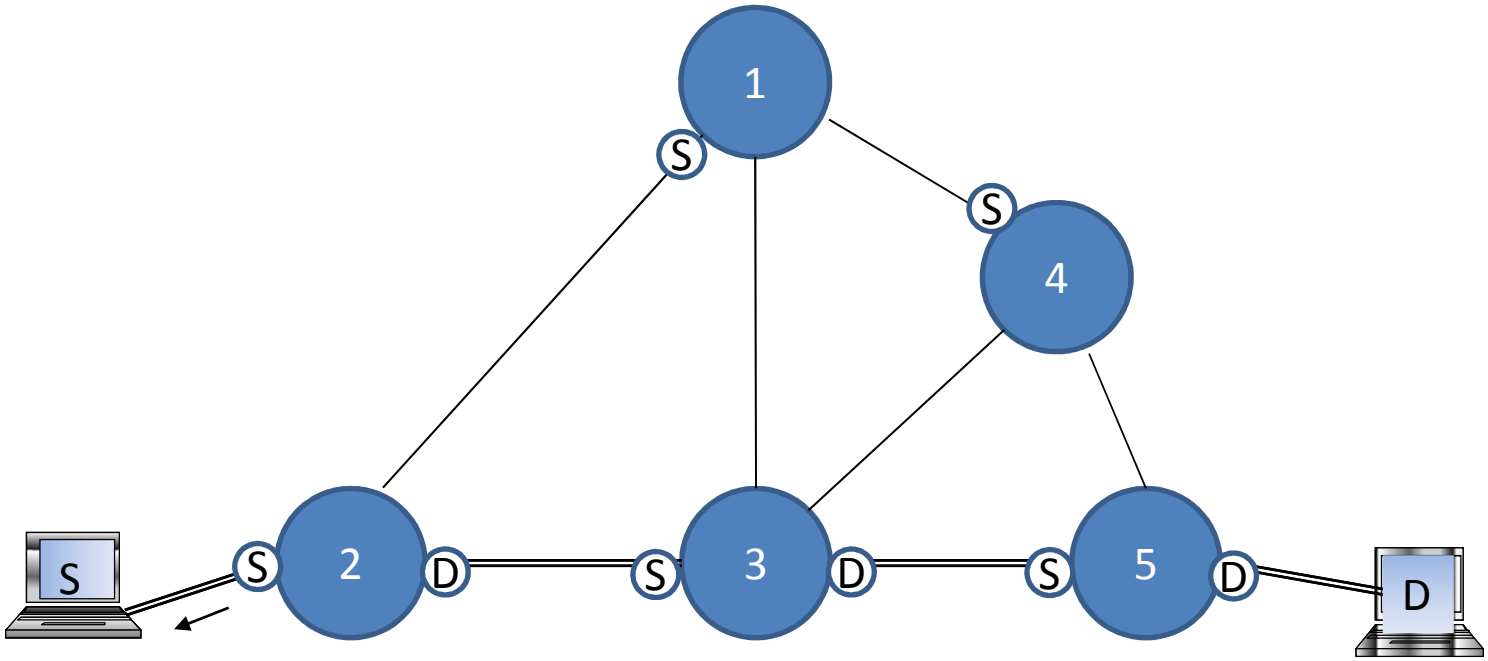
S *Port locked to S*

D *Port locked to D*

→ *ARP (path) request (broadcasted)*

← *ARP (path) reply (confirm) (unicast)*

ARP Reply arrives at S and completes the path set up



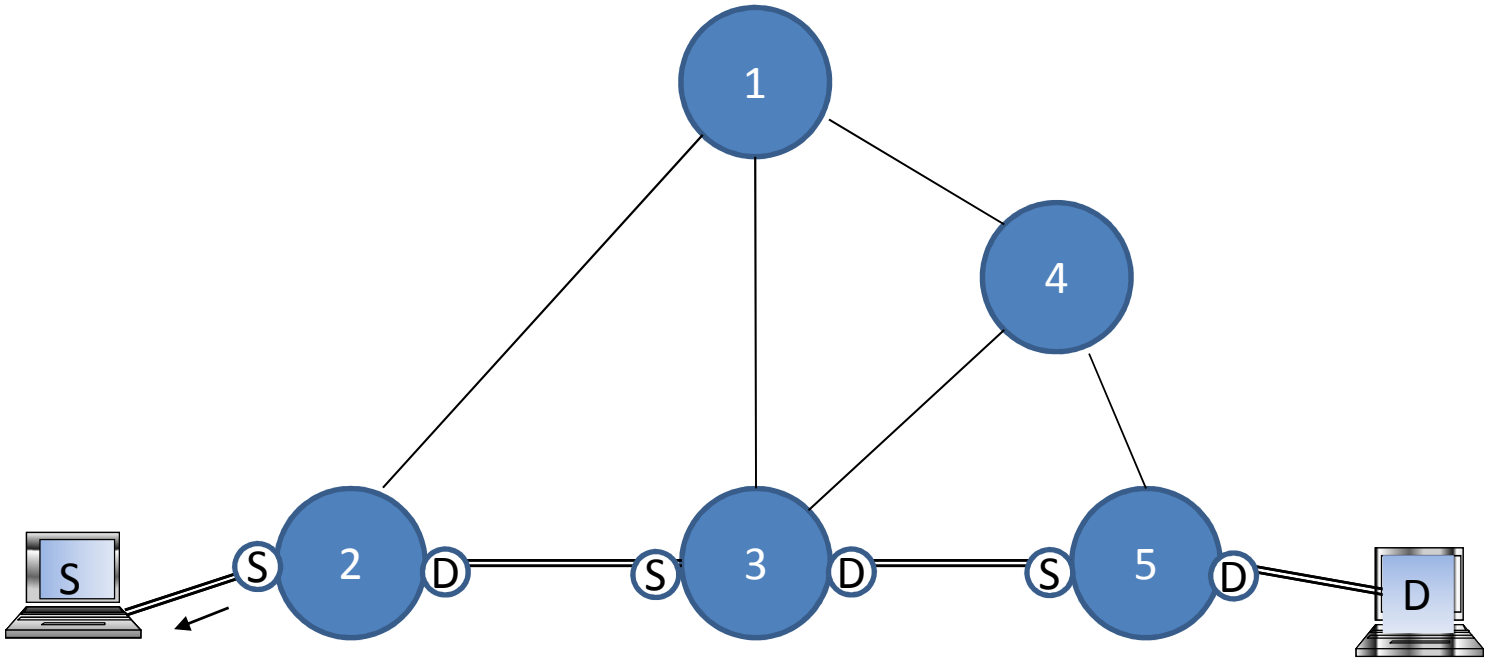
S *Port locked to S*

D *Port locked to D*

→ *ARP (path) request (broadcasted)*

← *ARP (path) reply (confirm) (unicast)*

Unconfirmed addresses learned expire



S *Port locked to S*

D *Port locked to D*

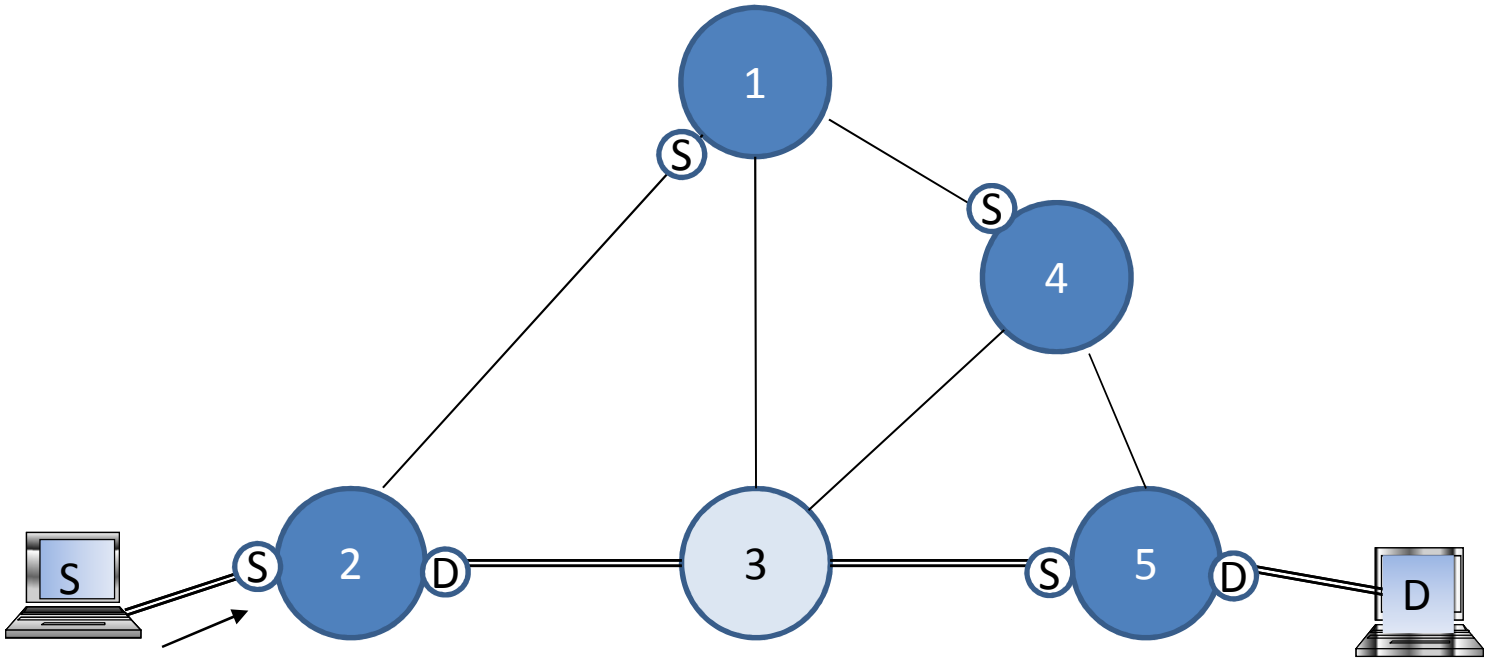
→ *ARP (path) request (broadcasted)*

← *ARP (path) reply (confirm) (unicast)*

Path repair

- Unicast frame with unknown source address arrives at a bridge
- Several variants
  - ARP Request reissue from first bridge w/o path
  - Encapsulate frame on broadcast frame (with all Fast Path bridges multicast destination address and return via input port towards source bridge, who reissues ARP Request.
  - Combinations and variants of the above

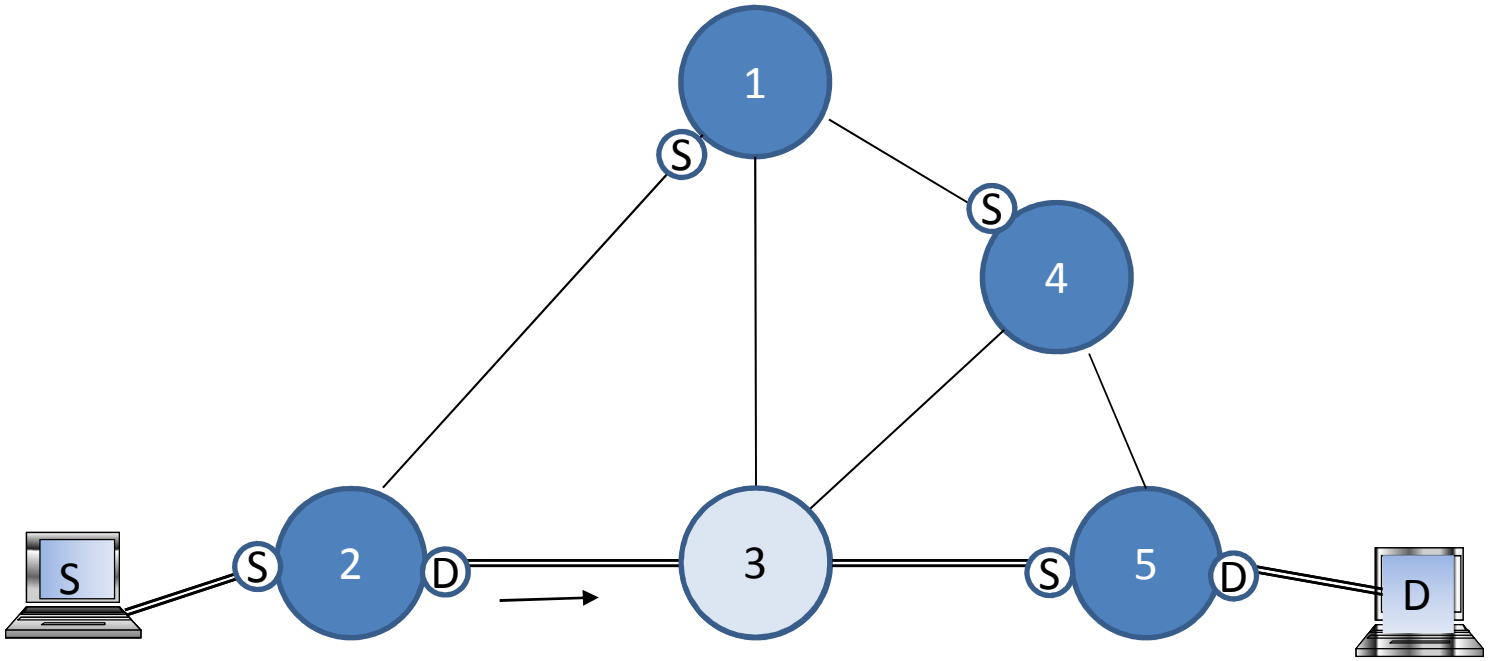
Path repair (bridge 3 flushed all MACs by initialization)



S *Port locked to S*

D *Port locked to D*

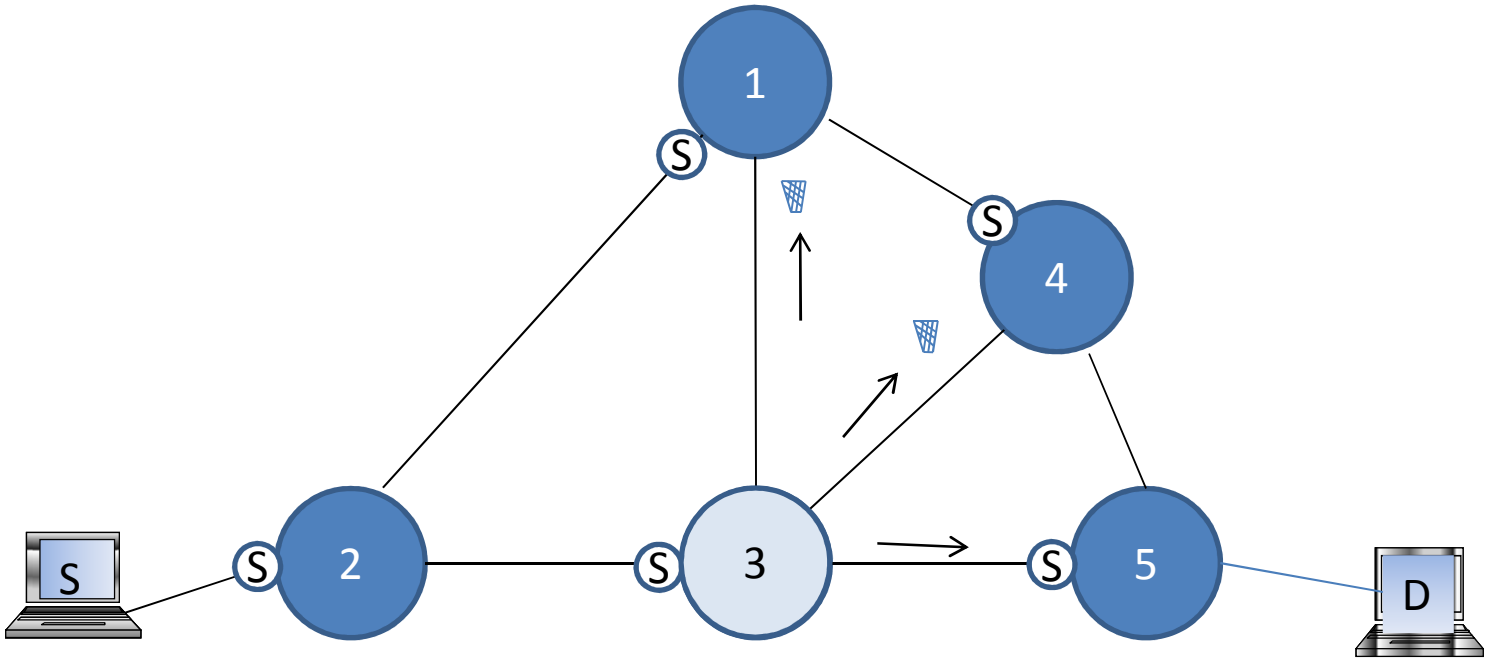
Path repair (bridge 3 had all MACs flushed by initialization)



S *Port locked to S*

D *Port locked to D*

Path repair (bridge 3 issues ARP request)



S *Port locked to S*

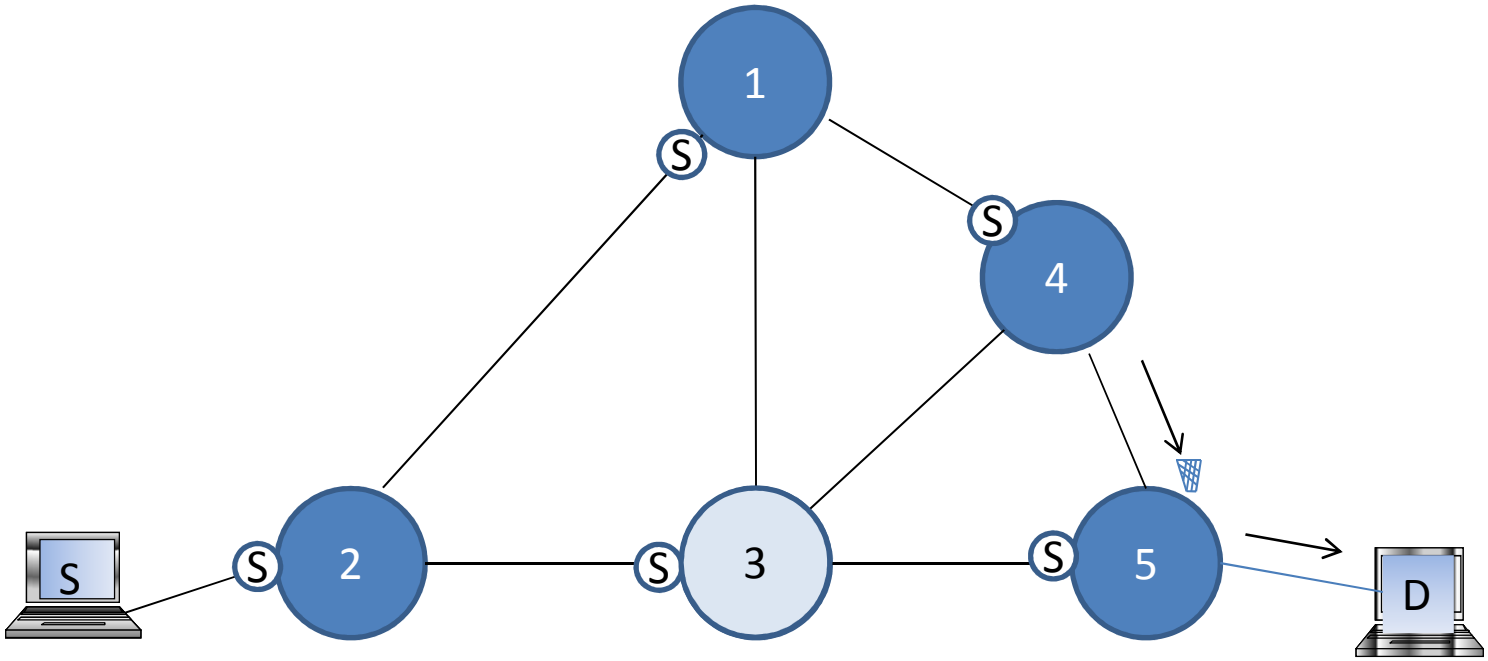
D *Port locked to D*

ARP (path) request (broadcasted)

Late frame discarded

ARP (path) reply (confirm) (unicast)

Path repair



S *Port locked to S*

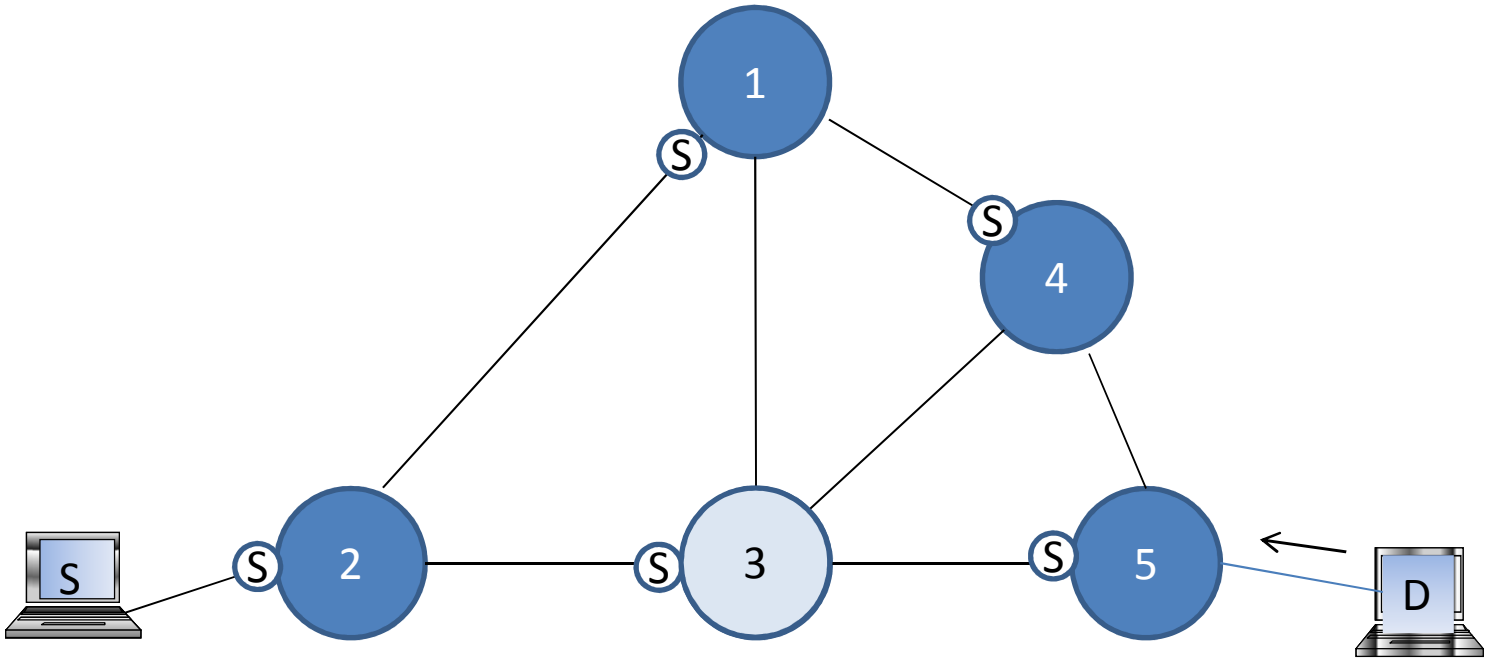
D *Port locked to D*

ARP (path) request (broadcasted)

Late frame discarded

ARP (path) reply (confirm) (unicast)

Path repair completing



S *Port locked to S*

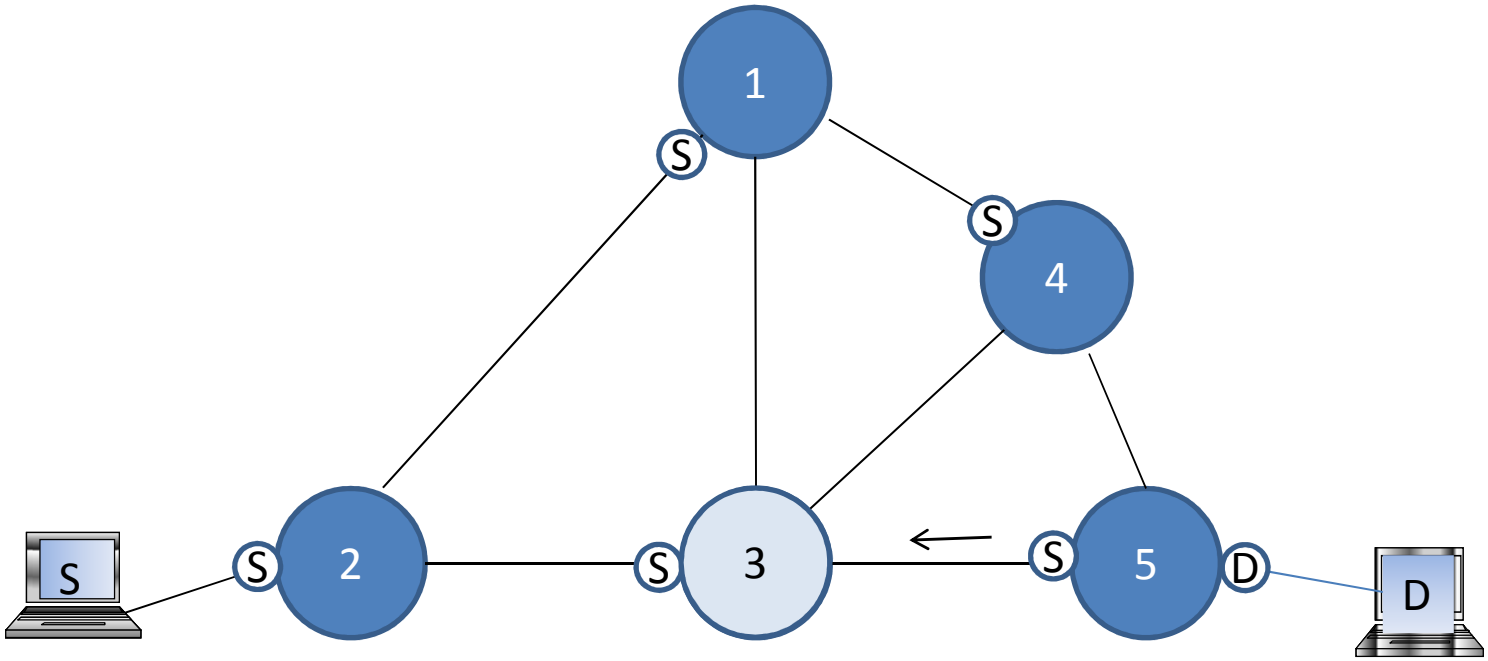
D *Port locked to D*

ARP (path) request (broadcasted)

Late frame discarded

ARP (path) reply (confirm) (unicast)

Path repair completing



S *Port locked to S*

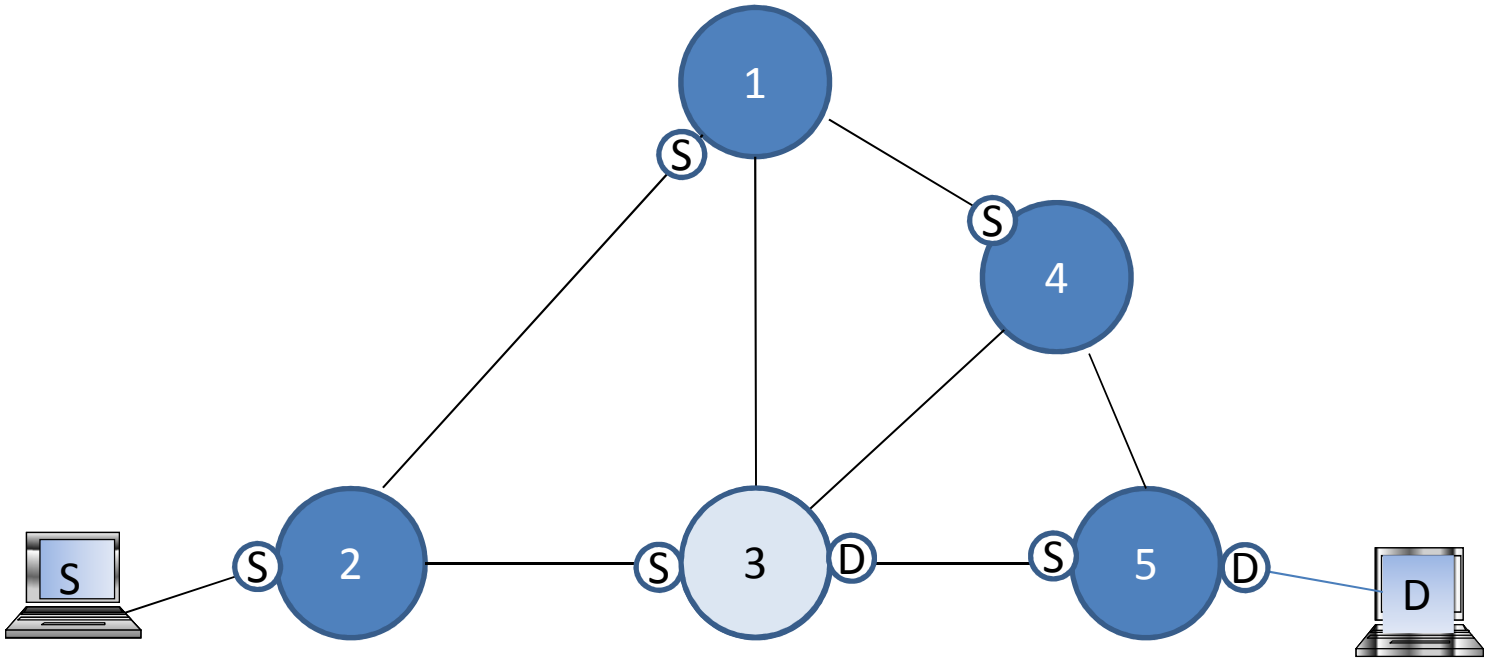
D *Port locked to D*

ARP (path) request (broadcasted)

Late frame discarded

ARP (path) reply (confirm) (unicast)

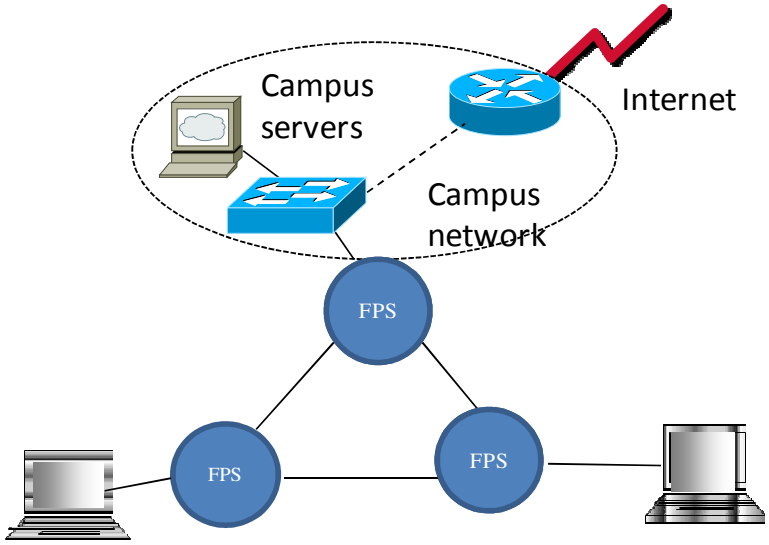
Path repair completed



S *Port locked to S*

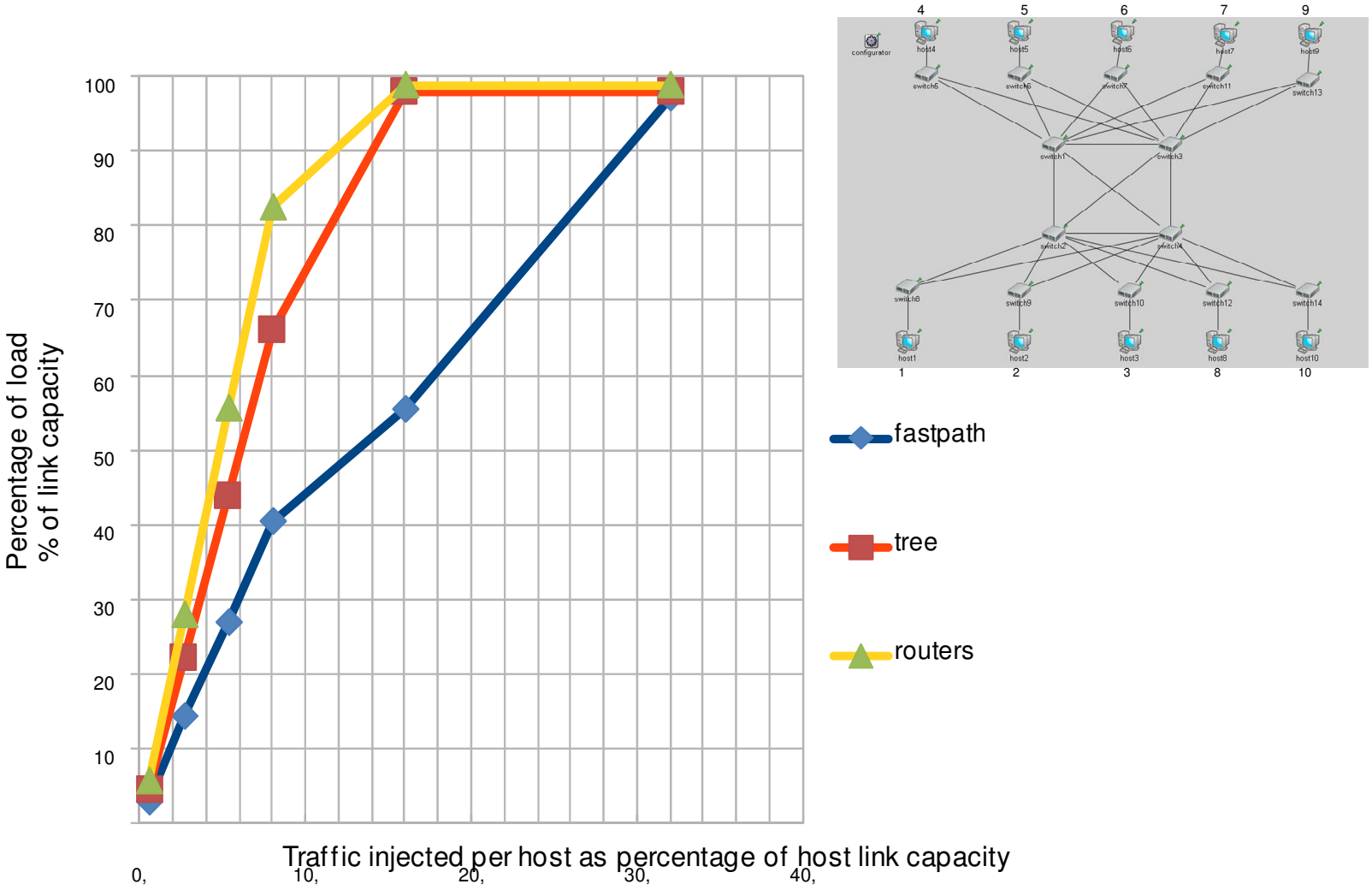
D *Port locked to D*

Proof of concept (Linux ebtables)



- Functionally simple to code and implement
- Services of campus network operate smoothly (DHCP, video streaming)
- Delays similar to hardware switches (on kernel part)

Enterprise network throughput



Throughput of paneuropean reference network

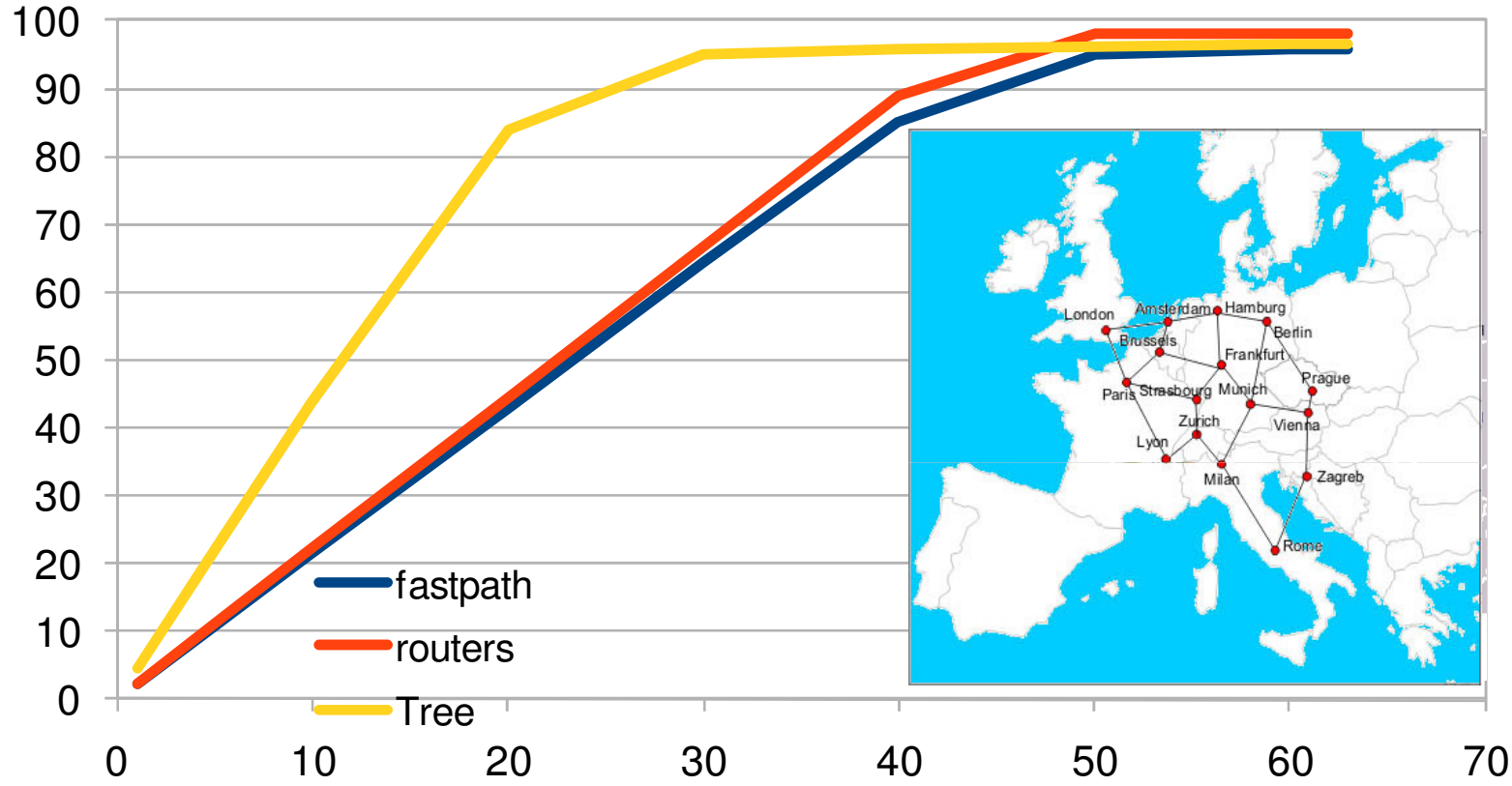
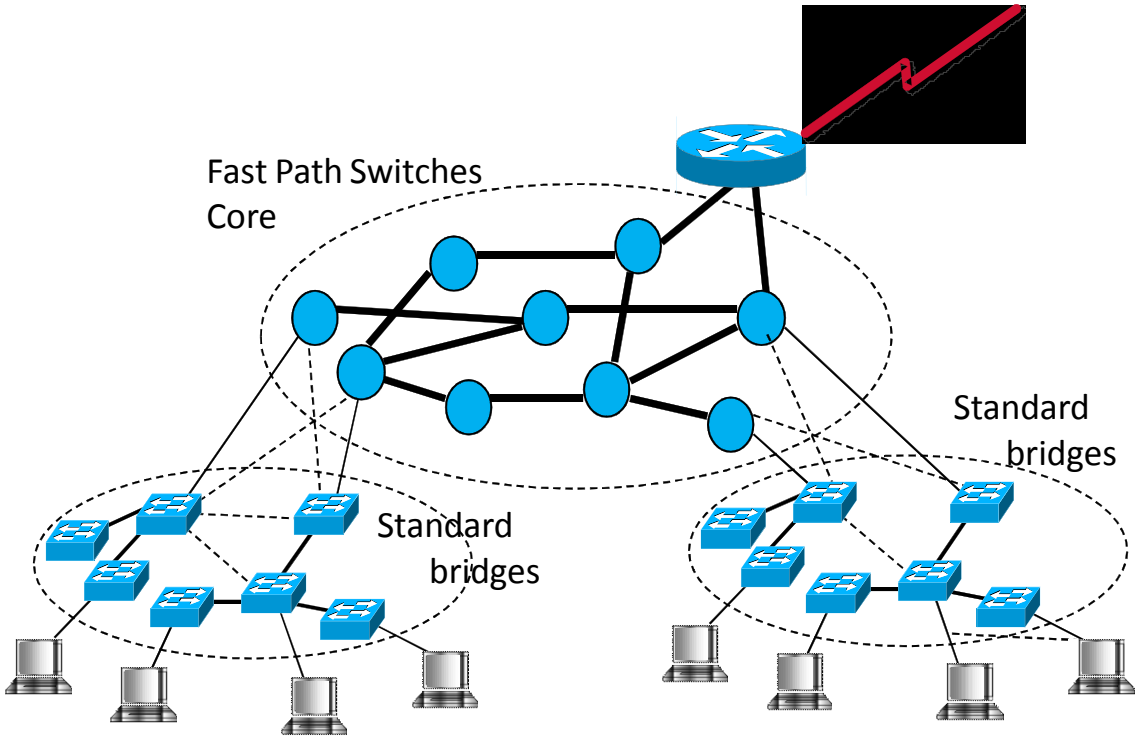


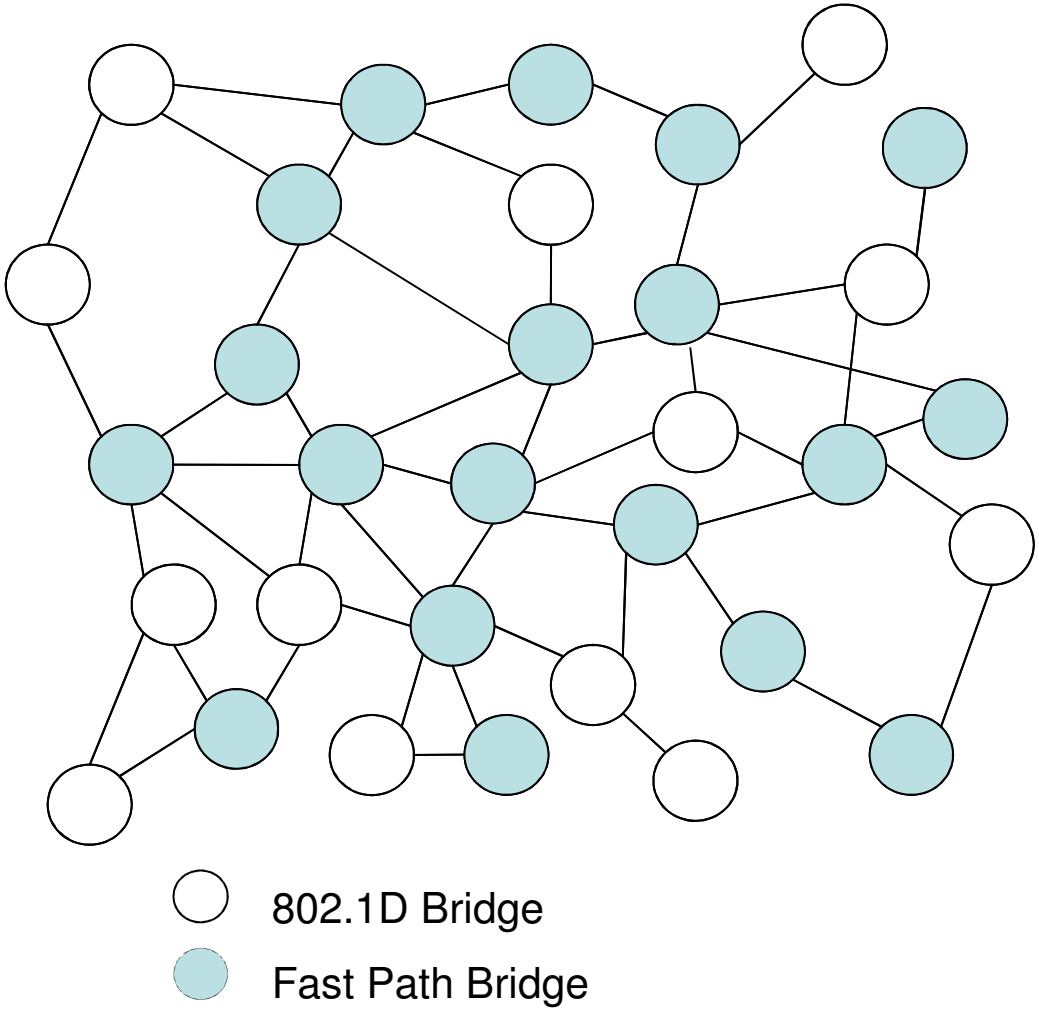
Figure 9. Throughput comparison of pan european network in % of most loaded link versus % of average traffic load applied at the sending host link

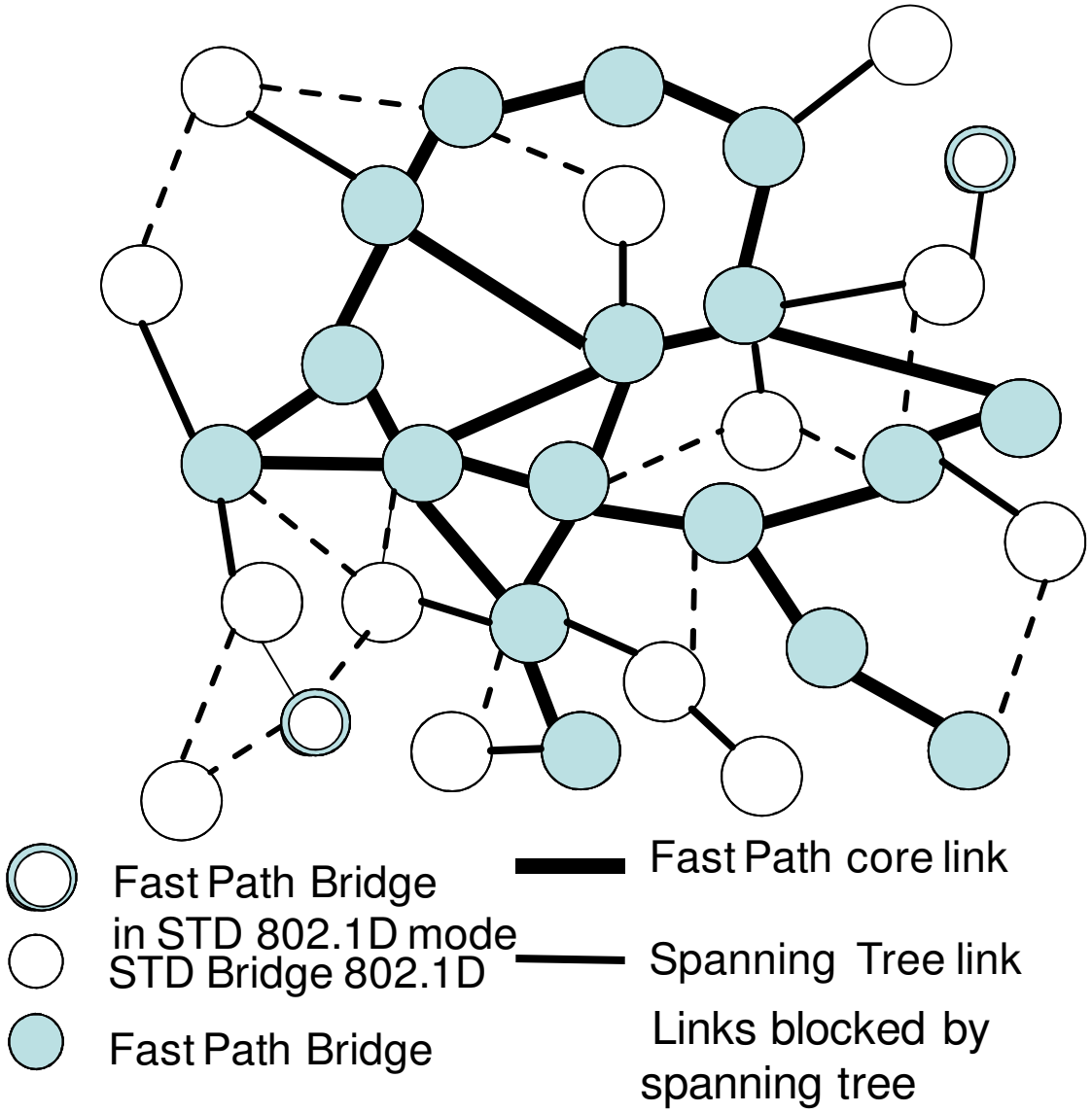
Compatibility with standard bridges in core-island mode



- 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: Mix of bridges topology example




Fast Path Bridge
in STD 802.1D mode
STD Bridge 802.1D
Fast Path Bridge
Fast Path core link
Spanning Tree link
Links blocked by
spanning tree

Conclusions

- *Shortest Path Transparent Bridging is possible and simpler **without** routing protocols and spanning trees*
- *Fast Path Ethernet switches perform equal or better than shortest path routing in terms of throughput with lower complexity*
- *Broadcast of ARP Requests is (d-1 times) increased to set up paths, can be mitigated incorporating Etherproxy functionality on Fast Path switches*
- *Compatible with IEEE 802.1 Bridges*

Future work

- Continue study of load distribution
- Etherproxy functionality on Fast Path Switches
- Scalability tests
- Optimization of user multicast traffic
- Specific Datacenters scenarios
- Proof of concept on Openflow/NetFPGAs

Some references

- IEEE 802.1D-2004 IEEE standard for local and metropolitan area networks-Media access control (MAC) Bridges. http://standards. ieee.org/getieee802 /802.1.html.
- M. Seaman. Shortest Path Bridging. http://www.ieee802.org/ 1/files/public/docs2005/new-seaman-shortestpath-par-0405-02.htm.
- Transparent interconnection of lots of links (TRILL) WG. Available on line at: http://www.ietf.org/html. charters/trill-charter.html
- Khaled Elmeleegy, Alan L. Cox: EtherProxy: Scaling Ethernet By Suppressing Broadcast Traffic. INFOCOM 2009: 1584-1592
- Ebtables. http://ebtables.sourceforge.net
- Omnet simulator. http://www.omnetpp.org