

Advanced Computer Networks

Interconnection Layer 2: bridges and VLANs

Prof. Andrzej Duda
duda@imag.fr

<http://duda.imag.fr>

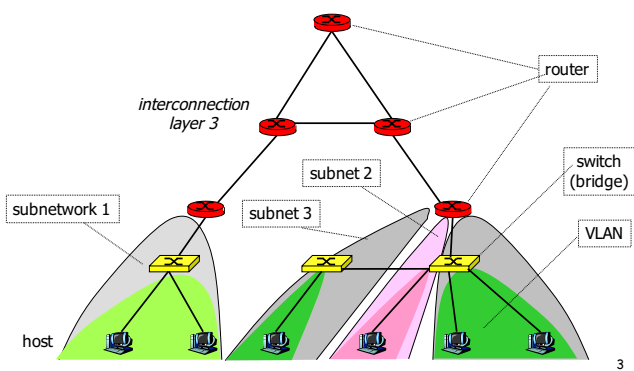
1

Contents

- Transparent bridges
- Spanning Tree Protocol (STP)
- Rapid Spanning Tree Protocol (RSTP)
- VLANs

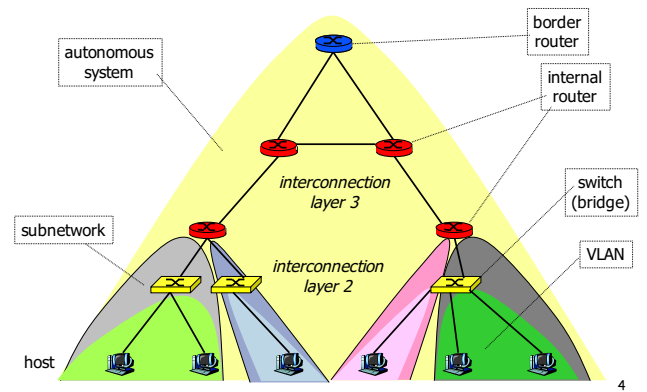
2

Interconnection structure



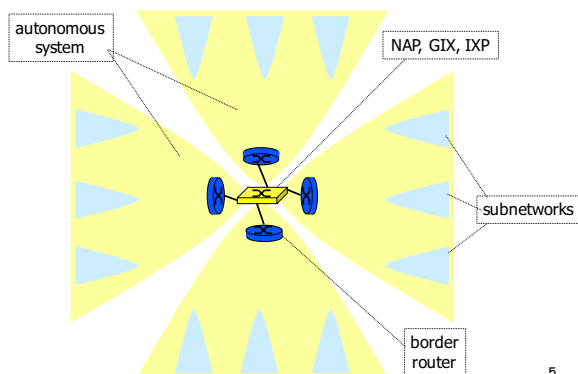
3

Autonomous systems



4

Internet



5

Interconnection of AS

- Border routers
 - interconnect AS
- NAP or GIX, or IXP
 - exchange of traffic - peering
- Route construction
 - based on the path through a series of AS
 - based on administrative policies
 - routing tables: aggregation of entries
 - works if no loops and at least one route - routing protocols (EGP - External Routing Protocols)

6

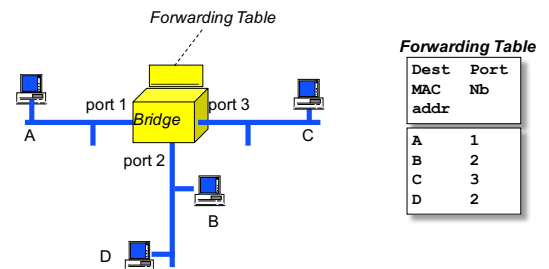
Transparent Bridging (TB)

- Bridges are intermediate systems that forward MAC frames to destinations based on MAC addresses
- Interconnect systems beyond one LAN segment, keeping main characteristics of LAN
 - without additional addresses
 - MAC addresses used to identify end systems
- End systems ignore that there are transparent bridges
 - bridge is transparent
 - MAC frames not changed by bridges
 - frames not sent *to* bridge, but rather: bridge is promiscuous
 - listens to all frames and retransmits if needed

7

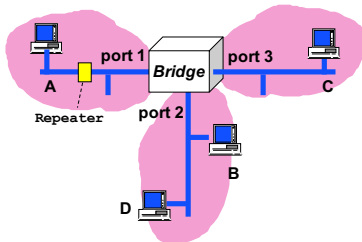
Transparent Bridging (TB)

- Administrator creates the forwarding table
- TB operation
 - connectionless forwarding, unstructured addresses



8

LB: Learning Bridge



Dest MAC addr	Port Nb

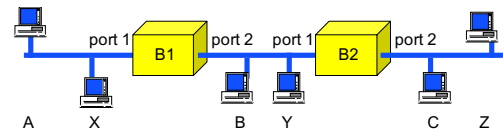
Dest MAC addr	Port Nb
A	1
B	2
C	3
D	2

- Bridge builds forwarding table by reading all traffic
 - bridges are plug and play: no address configuration (no IP address needed)
 - table built by **learning** from SA field in MAC frame
 - a table entry has limited life (**MaxLife**, 5 minutes)
- Flooding if destination address unknown or group address

10

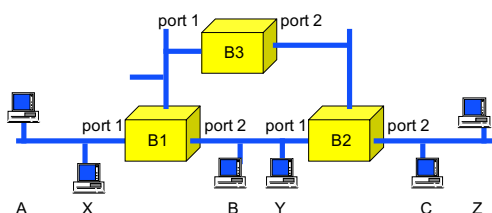
Several Learning Bridges

- Can the learning bridge be extended to a network of bridges?
- How does B2 see the network?



Loops

- What happens when A sends a frame to B?
 - assume empty forwarding table



11

Loop-Free topology

- Learning bridge works well on Loop-Free topology only
 - Bidirectional graph: node = bridge, edge = connection through LAN
 - Loop free - bidirectional graph = bidirectional tree
 - examples: line, star
 - On a tree, there is only one path from A to B
 - **Proposition:** If bridge topology is loop-free, then there exists only one path from any end system to any bridge
 - Loop-free topology is required and sufficient for

12

Spanning Tree Bridges

- Based on learning bridge:
 - table driven forwarding, flooding if unknown DA or multicast, learning
- Forces topology to a tree
 - Spanning Tree algorithm run by all bridges
 - Some ports blocked to prevent loops
 - ports that are allowed to forward frames (in either way) are said to be "in the forwarding state" or called "forwarding ports"
- Interconnection of bridges
 - several parallel paths for reliability
 - Spanning Tree algorithm chooses one path at a given instant

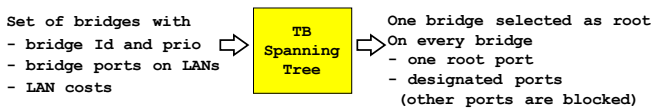
13

Forwarding Method

Individual PDU forwarding	Copy all frames on all forwarding ports Frame received on port i -> /* port i is forwarding */ If DA is unicast, is in forwarding table with port j and j is a forwarding port then copy to port j else flood all forwarding ports i Update forwarding table with (i, SA)
Control method	Maintain spanning tree and port states Learn addresses on reading traffic

14

TB Spanning Tree Specification



- Bridges viewed as a bidirectional graph (nodes = bridges)
- Selection of the root bridge
 - lowest priority with lowest identifier
- Spanning Tree = shortest path tree from root to all bridges
 - edge costs set by management, high cost = less traffic
 - based on distributed Bellman-Ford (distance vector)
 - $cost_to_root = best_announced_cost + local_cost$

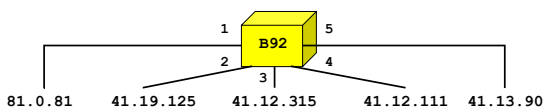
15

Spanning Tree Specification

- Root** port on one bridge = port towards root, shortest path
 - in case of equal costs, lowest id chosen
- Designated bridge
 - one per LAN
 - it has the shortest path to root via root port
- Designated** ports
 - all ports for which the bridge is designated
 - connect LANs to the spanning tree
- Ports other than root or designated are **blocked**
- Configuration messages
 - $rootId.cost_to_root.senderId.port$ (41.13.92.3)
 - simplified: $rootId.cost_to_root.senderId$

16

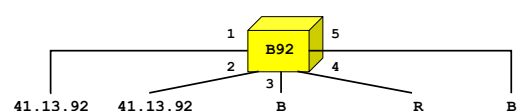
Spanning Tree example



- Best root: 41
- Best cost: $12 + 1 = 13$, on port 3 or 4 ($cost=1$)
- Root port:** 4, because $111 < 315$
- New message: 41.13.92
- Ports 1 and 2 are **designated**: 41.13.92 is better than 81.0.81 and 41.19.125
- Port 3 and 5 are **blocked**: 41.13.92 is not better than 41.12.315 nor 41.13.90

17

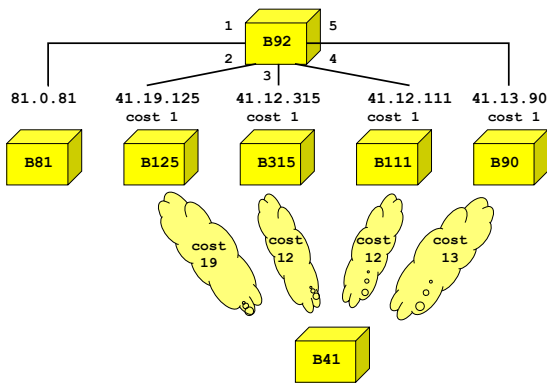
Spanning Tree example



- Message 41.13.92 sent periodically on ports 1 and 2
- Ports 1, 2, 4 participate in forwarding (they are in the Spanning Tree)

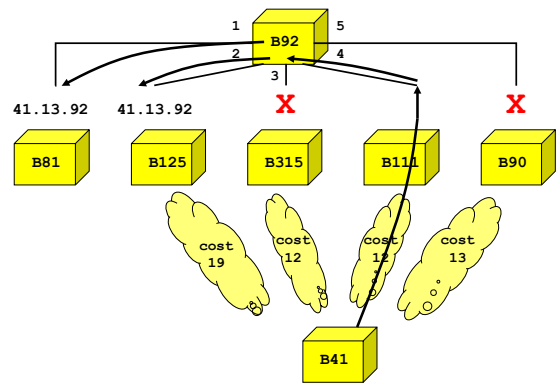
18

Spanning Tree example



19

Spanning Tree example



20

STP - Spanning Tree protocol

- IEEE 802.1D
- Distributed in all bridges
- Bridges exchange messages with neighbours in order to both
 - elect a root
 - determine shortest path tree to root
 - root port = port towards root on shortest path tree
 - designated ports = connect LANs to the spanning tree
 - designated bridge = one per LAN, has shortest path to root via root port

21

STP (IEEE 802.1d)

- Each bridge has a Bridge Identifier number, based on MAC address + configurable offset
- Bridge with smallest Bridge Identifier is the "root"
- Each link has a cost

Link Bit Rate	Cost
4 Mb/s	250
10 Mb/s	100
16 Mb/s	62
45 Mb/s	39
100 Mb/s	19
155 Mb/s	14
622 Mb/s	6
1 Gb/s	4
10 Gb/s	2

22

Bridge PDUs

- Control method uses control frames called Bridge PDUs (BPDUs)
 - 802.3 encapsulation, LLC frame with SAP = x42
 - MAC DA = all bridges (multicast) 01 80 C2 00 00 00
- BPDUs are not forwarded by bridges
 - unlike all other frames BPDUs are sent by one bridge to all bridges on the same LAN segment
 - reminder: a data frame is never sent to bridge by end system
- Configuration BPDU contains
 - root Id
 - cost to root (from sender of config BPDU)
 - id of sender
 - port number (omitted in the examples)

23

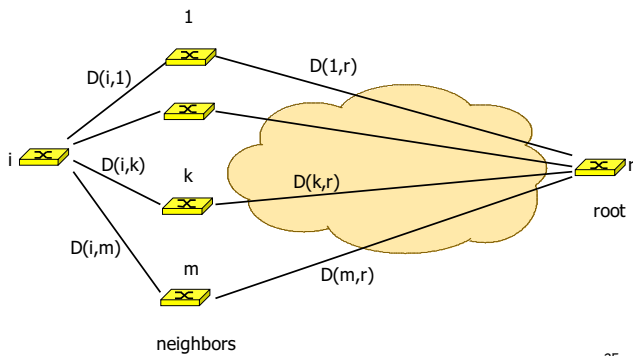
Initialization of Spanning Tree

- Bridge initially assumes self as a root
- Bridge computes own new config BPDU based on received information
 - determine best root so far
 - distance to root with Bellman-Ford

$$\text{distance } D \text{ from me to root} = \min [D(\text{me, neighbor}) + D(\text{neighbor, root})]$$
- On every port, Bridge transmits config BPDU until it receives a better config BPDU on that port
 - better = closer to root (lower cost or lower Id)
- On every port, bridge maintains copy of best config BPDU sent or received

24

Bellman-Ford algorithm



Basic ST Procedure

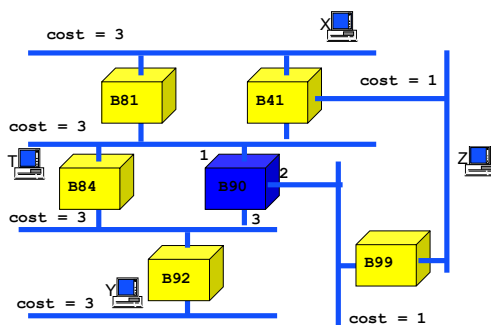
```

config BPDU received on any port or port enabled ->
compute new root;
compute new cost to root; /* Bellman Ford */
build new_config_BPDU;
for all ports i do
    if new_config_BPDU better than stored_config[i]
        then store and send on port i;
end

compute root port /* smaller distance to root */
designated ports = ports where config BPDU was sent
blocked ports = other ports

r.c.s better than r'.c'.s' iff
    (r<r') or (r=r' and c<c') or (r=r' and c=c' and s<s')
    
```

Complex example



Initialization of Spanning Tree

- Bridge B90 prepares config BPDU 90.0.90 and sends on all ports; B90 configuration tables:

1 90.0.90	1 < 41.0.41	1 41.0.41	2,3 > 41.3.90	1 41.0.41
2 90.0.90	2 90.0.90	2 90.0.90		2 41.3.90
3 90.0.90	3 90.0.90	3 90.0.90		3 41.3.90

2 < 41.1.99	1 41.0.41	3 > 41.2.90	1 41.0.41
	2 41.1.99		2 41.1.99
	3 41.3.90		3 41.2.90

Root Port	: 1	Root Port	: 2
Designated Ports	: 2,3	Designated Ports	: 3
Blocked Ports	:	Blocked Ports	: 1

message received on port 1: 1 < 41.0.41 message format: rootId.cost_to_root.senderId

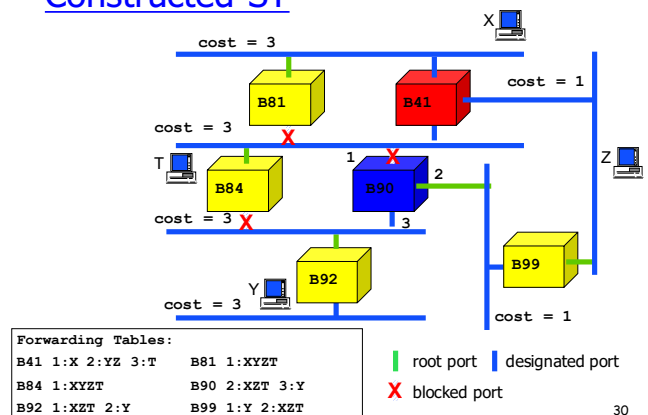
28

Comments

- When receiving a message we compare the cost (with the local cost included), but we store the message received (without the cost)
- On receiving 41.0.41 on port 1:
 - 41.3.41 < 90.0.90? yes -> 1 becomes root
 - new config msg = 41.3.90
 - 41.3.90 < 90.0.90? yes -> 2 becomes designated
 - 41.3.90 < 90.0.90? yes -> 3 becomes designated
- On receiving 41.1.99 on port 2:
 - 41.2.99 < 41.3.41? yes -> 2 becomes root
 - new config msg = 41.2.90
 - 41.2.90 < 41.3.90? yes -> 3 becomes designated
 - 41.2.90 < 41.0.41? no -> 1 becomes blocked

29

Constructed ST



STP Topology Management

- Topology change can be
 - local** - a configuration msg changes the state of a port (one port changes into the Forwarding or Blocking state)
 - global** - topology update mechanism via root
- Detection
 - configuration message is too old (the path to the root is no longer available)
 - receive a new better configuration
- When topology change detected
 - inform root
 - restart spanning tree computation
 - force bridges to use a shorter timeout interval (purge the forwarding table)

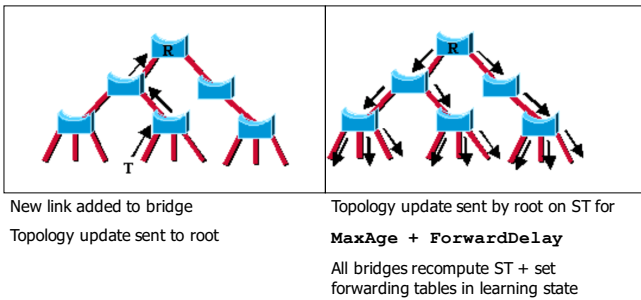
31

Topology change

- When one bridge detects a topology change
 - bridge sends topology update BPDU towards root and enters Listening state (upstream bridges repeat BPDU up to root)
 - root forwards new config BPDU with "topology change flag" set during **ForwardDelay** (15 s) + **MaxAge** (20 s)
 - causes all bridges to use the short timeout value for the forwarding table (see later)
 - until BPDU from root received with "topology change" flag cleared

32

Example



Source: CISCO RSTP White Paper

33

Configuration monitoring

- root sends a configuration message every **HelloTime** (2 s)
- message received with **Age**, retransmitted with **Age += 1**
- if **Age = MaxAge** (20 s), delete the stored configuration and restarts basic ST procedure

```

Root sends config BPDUs every HelloTime;

Bridge B receives config BPDU on root port i ->
  Reset timer Age on stored_config[i]
  for all designated ports j
    B sends own config BPDU
    B resets timer Age on stored_config[j]

Bridge B timeouts (MaxAge) stored_config[j]->
  delete stored_config[j];
  B performs basic ST procedure;
    
```

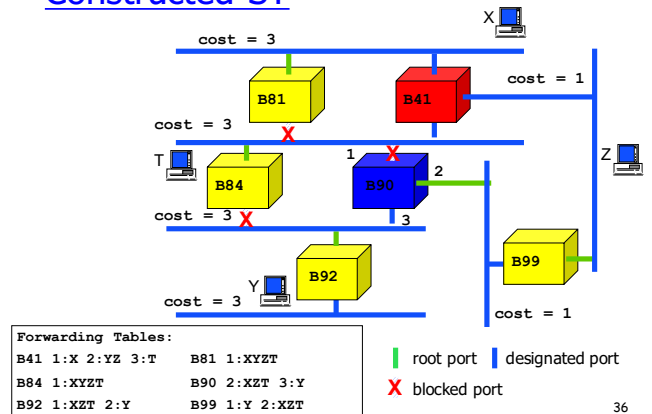
34

Timers

- Timers used in topology management
 - HelloTime** (2 s): time interval between Config BPDUs sent by the Root Bridge.
 - ForwardDelay** (15 s): time interval that a bridge port spends in both the Listening and Learning states
 - MaxAge** (20 s): time interval that a bridge stores a BPDU before discarding it
 - recommended values for a spanning tree of diameter 7
- Time to update
 - detect and rebuild: $35\text{ s} = 20\text{ s} + 15\text{ s}$
- Time to change from blocking to forwarding state
 - detect, rebuild, and learn addresses: $50\text{ s} = 20\text{ s} + 15\text{ s} + 15\text{ s}$

35

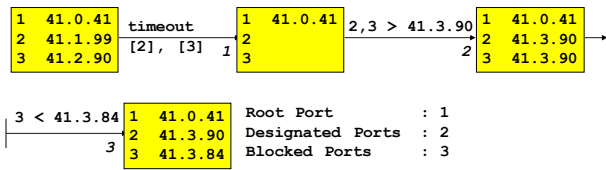
Constructed ST



36

Example

- B99 powered off; stored config at B90:



- Spanning Tree after failure?

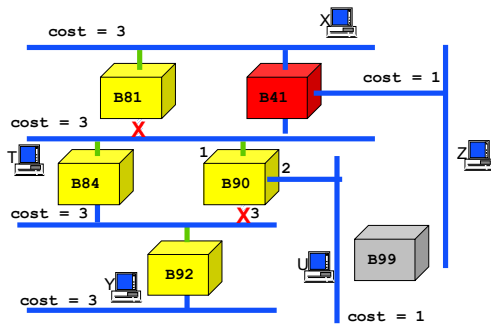
37

Comments

- After timeout:
 - 41.3.41 is the best configuration -> 1 becomes root
 - new config msg = 41.3.90
 - 2 and 3 becomes designated
- On receiving 41.3.84 on port 3:
 - 41.6.84 < 41.3.41? no -> 1 stays root
 - new config msg = 41.3.90
 - 2 stays designated
 - 41.3.90 < 41.3.84? no -> 3 becomes blocked

38

ST after failure



39

Synchronization with Forwarding

- Topology changes cause loops or loss of connectivity
 - during reconfiguration, topology is not yet (in general) loop free
 - even transient loops should be avoided
- Solution: Forwarding state is not immediately operational
 - pre-forwarding states:
 - Listening (accept config msgs, no forwarding): wait for stabilization of ST (**ForwardDelay**, 15 sec)
 - Learning (learn MAC addresses, no forwarding): wait for addresses to be learnt (**ForwardDelay**, 15 sec)

State	Actions		
	Forward	ST	Learn
Blocking			X
Listening			X
Learning		X	X
Forwarding	X	X	X

40

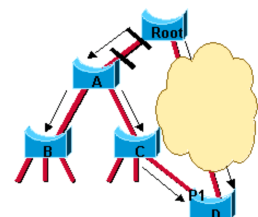
Forwarding Table entry timers

- MaxLife** = duration of an entry in the forwarding table
- Two timer values are used
 - long timer (5mn): normal case
 - short timer = **ForwardDelay** (15 s): after spanning tree updates
- Timer switching mechanism
 - Bridge B detects change in ST -> **MaxLife** = **ForwardDelay**

41

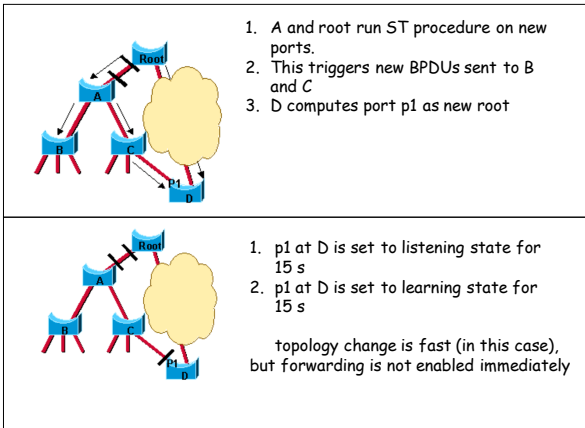
Example

- Bridge A newly connected to root. What happens?



Source: CISCO RSTP White Paper

42



43

RSTP - Rapid ST protocol

- IEEE 802.1W
- Evolution of STP
- Goal: fast reconfiguration
- Improvement of handling topology changes and synchronization with packet forwarding
 - avoids use of timers as much as possible
- Main improvements are
 - fast reconfiguration: use of alternate paths to root or backup path to a LAN
 - fast transition to forwarding state with negotiation protocol instead of relying on timers
 - fast flushing of forwarding tables after topology changes

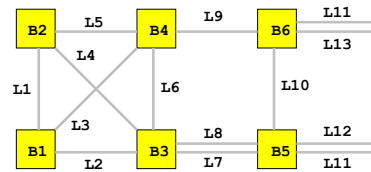
44

Port Roles in RSTP

- A port **role** is one of: root, designated, alternate, backup, blocked
- root** port = port towards root (same as STP)
- designated** port = connects LAN to the spanning tree (same as STP)
- Port that is not root nor designated
 - is **alternate**: connects the bridge to root after topology update (alternate path to root)
 - is **backup**: connects LAN to the spanning tree after topology update (alternate path to root for the LAN)
 - is **blocked**: not in the spanning tree

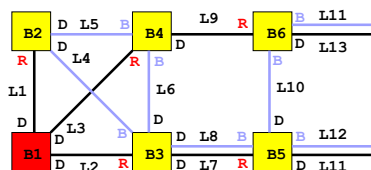
45

Another example of STP



46

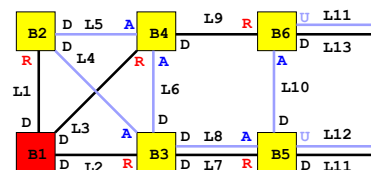
Constructed ST



- B1 - root
- R - root ports, D - designated ports, B - blocked ports

47

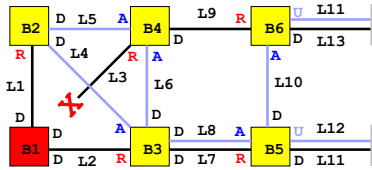
ST constructed by RSTP



- B1 - root
- R - root ports, D - designated ports, B - blocked ports
- A - alternative ports, U - backup ports

48

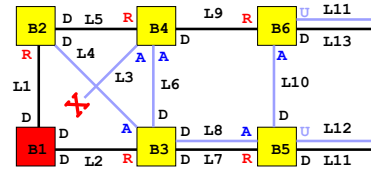
L3 fails



- B1 - root
- R - root ports, D - designated ports, B - blocked ports
- A - alternative ports, U - backup ports

49

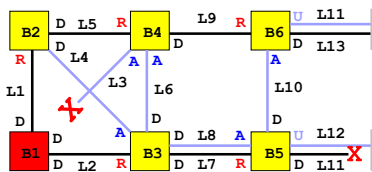
L3 fails



- On B4
 - port on L3 becomes A and state Discarding
 - port on L5 becomes R and state Forwarding

50

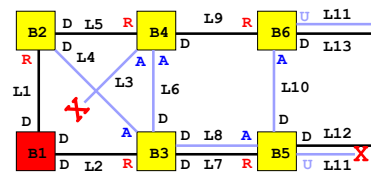
L11 fails



- B1 - root
- R - root ports, D - designated ports, B - blocked ports
- A - alternative ports, U - backup ports

51

L11 fails



- On B5
 - port on L11 becomes U and state Discarding
 - port on L12 becomes D and state Forwarding

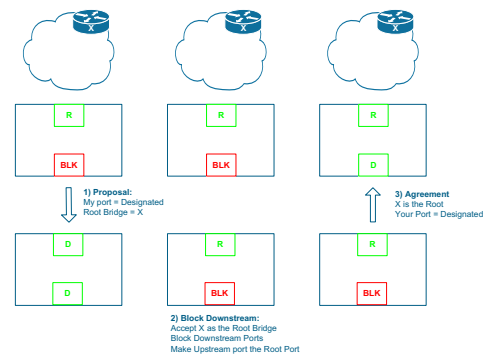
52

RSTP - Rapid ST protocol

- If topology change
 - same reconstruction protocol as STP
 - topology change notification flooded across ST
- Rapid recovery
 - Proposal/Agreement sequence between bridges that change state of a port: immediate transition to Forwarding state
 - link failure detection by MAC layer
 - change R to A and D to U (order of 10 ms)
 - but similar delay to STP, if topology update

53

RSTP - Rapid ST protocol

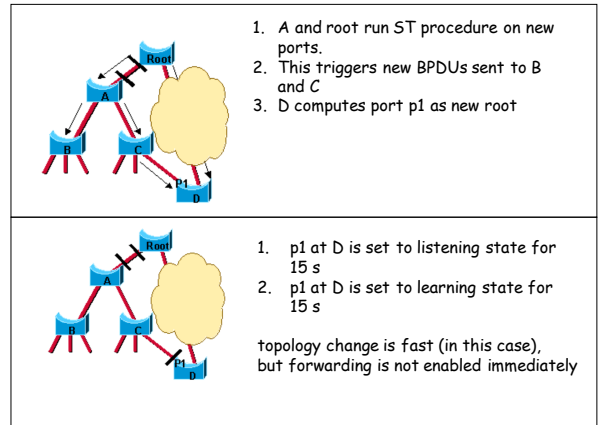


54

RSTP - Rapid ST protocol

- Better root information received:
 - block all designated ports
 - send Proposal BPDUs on all potential designated ports
- Downstream bridges:
 - compare Proposal with the current root information
 - if Proposal is better
 - elect root port
 - block all downstream ports
 - send Agreement BPDU upstream
 - send Proposal BPDUs on all potential designated ports
 - Otherwise
 - bridge that rejects Proposal – it has better root information
 - blocks the port on which it received Proposal
 - end of sync

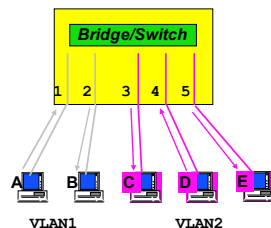
55



56

VLAN - Virtual LAN

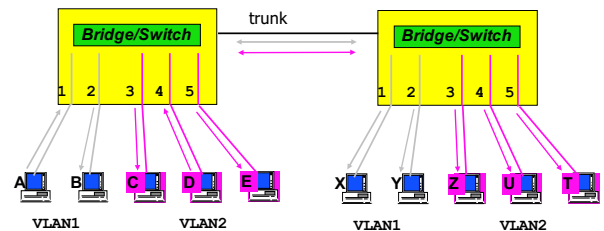
- Keep the advantages of Layer 2 interconnection
 - auto-configuration (addresses, topology - Spanning Tree)
 - performance of switching
- Enhance with functionalities of Layer 3
 - extensibility
 - spanning large distances
 - traffic filtering
- Limit broadcast domains
- Security
 - separate subnetworks



57

Virtual LANs

- No traffic between different VLANs
- VLANs build on bridges or switches



58

VLANs

- How to define which port belongs to a VLAN?
 - per port
 - simple, secure, not flexible for moving hosts (one host per port)
 - per MAC address
 - several hosts per port, flexible for moving hosts, not secure, difficult to manage, problems with protocols Layer 3 (should be coupled with dynamic address negotiation - DHCP)
 - per Layer 3 protocol
 - allows to limit frame broadcast (VLAN1: IP, VLAN2: IPX)
 - per Layer 3 address
 - one VLAN per IP subnetwork
 - flexible for moving hosts
 - less efficient (requires inspecting packets)
 - per IP Multicast group
 - hosts that join an IP multicast group can be seen as members of the same virtual LAN

59

VLANs

- How to extend VLAN to several bridges/switches?
 - needs frame identification - tagging
- Frame tagging
 - explicit tagging
 - add VLAN identifier to MAC frames
 - implicit tagging
 - VLAN Id deduced from port number, MAC address, layer 3 protocol, layer 3 address, IP Multicast group
 - implicit tagging makes use of filtering database
 - mapping between VLAN Id and the appropriate field (e.g. layer 3 address)

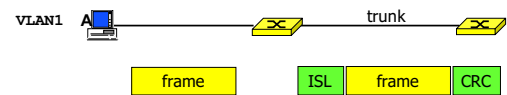
60

Frame tagging

- VLAN identifier in frames
 - usually done by the first switch/bridge
- Standards
 - CISCO: ISL (Inter-Switch Link)
 - IEEE 802.1Q/802.1P

61

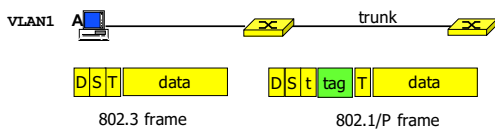
ISL (Inter-Switch Link)



- CISCO: ISL (Inter-Switch Link)
 - proprietary solution: encapsulates a frame in an ISL frame (26 bytes header, 4 bytes CRC)
 - incompatible with other vendors - increased maximal length of 802.3 frame

62

802.1Q



- Frame encapsulation
 - extension for assigning frame priority and VLAN tag
 - t - 2 bytes of TPI (Tag Protocol Identifier): 0x8100
 - tag - 2 bytes of TCI (Tag Control Information): priority (3 bits), VLAN Id (12 bits) (VID 0x001 often reserved for management)
 - max length = 1522 bytes

63

802.1Q

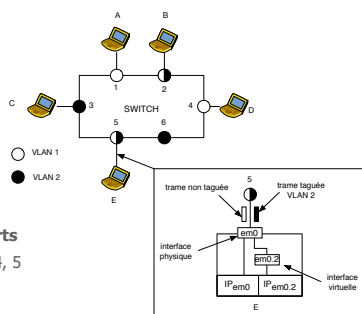
- Bridge/switch keeps track of VLAN members based on dynamic filtering entries
 - Dynamic Registration: specify ports registered for a specific VLAN
 - added and deleted using GARP VLAN Registration Protocol (GVRP), GARP is the Generic Attribute Registration Protocol
 - Group Registration: indicate frames to a group MAC address
 - added and deleted using Group Multicast Registration Protocol (GMRP)
 - multicasts sent on a single VLAN without affecting other VLANs

64

Tagged VLANs

- VLAN table

VLAN	tag	ports
1	untagged	1, 4, 5
1	tagged	2
2	untagged	2,3
2	tagged	5



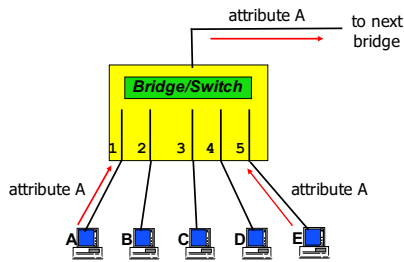
65

GARP

- Defines
 - method to declare attributes to other GARP participants
 - frame type to use (GPDU)
 - rules for registering/deleting attributes
- How does it work?
 - bridge wants to declare an attribute
 - send a declaration
 - other bridges propagate the declaration

66

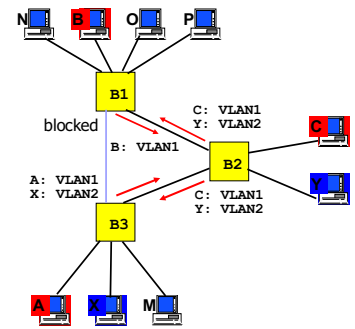
GARP



- Attribute propagation
 - stored at bridge
 - multiple attributes are filtered out - only one declaration is propagated
 - propagation follows the spanning tree of bridges

67

GARP

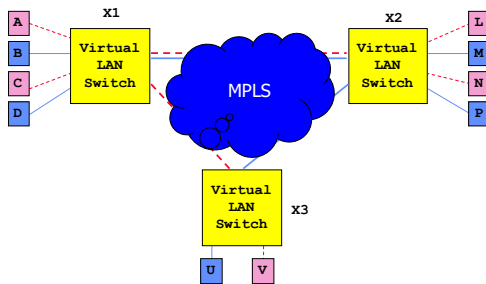


- Attributes on VLAN membership are sent over the spanning tree
- Frames are forwarded according to this information

68

Remote VLANs – VPLS service

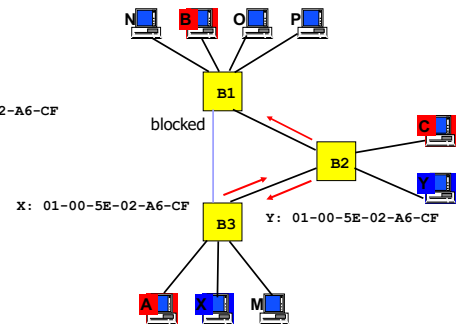
- Interconnect remote switches/bridges
- Uses ATM or other proprietary methods as a physical layer



69

GMRP

IP: 224.2.166.207
Ether: 01-00-5E-02-A6-CF



- Register to a group address - it uses GARP
- Multicast frames are forwarded according to this information

70

Conclusion

- Ethernet switches/bridges dominate
- 1Gb/s for hosts, 10-40-100 Gb/s in the backbone
- Complex interconnection
 - parallel paths may exist for reliability
 - SPT or RSTP guarantees loop-free interconnection
 - VLANs help to structure the interconnection
 - separate broadcast domains
 - limited scalability
- Products
 - Switch/Router - integrated ports: Layer 2 and Layer 3
 - administrator chooses the right level for each port

71