

# **11W NET3011**

## **CCNP SWITCH – Chapter 3**

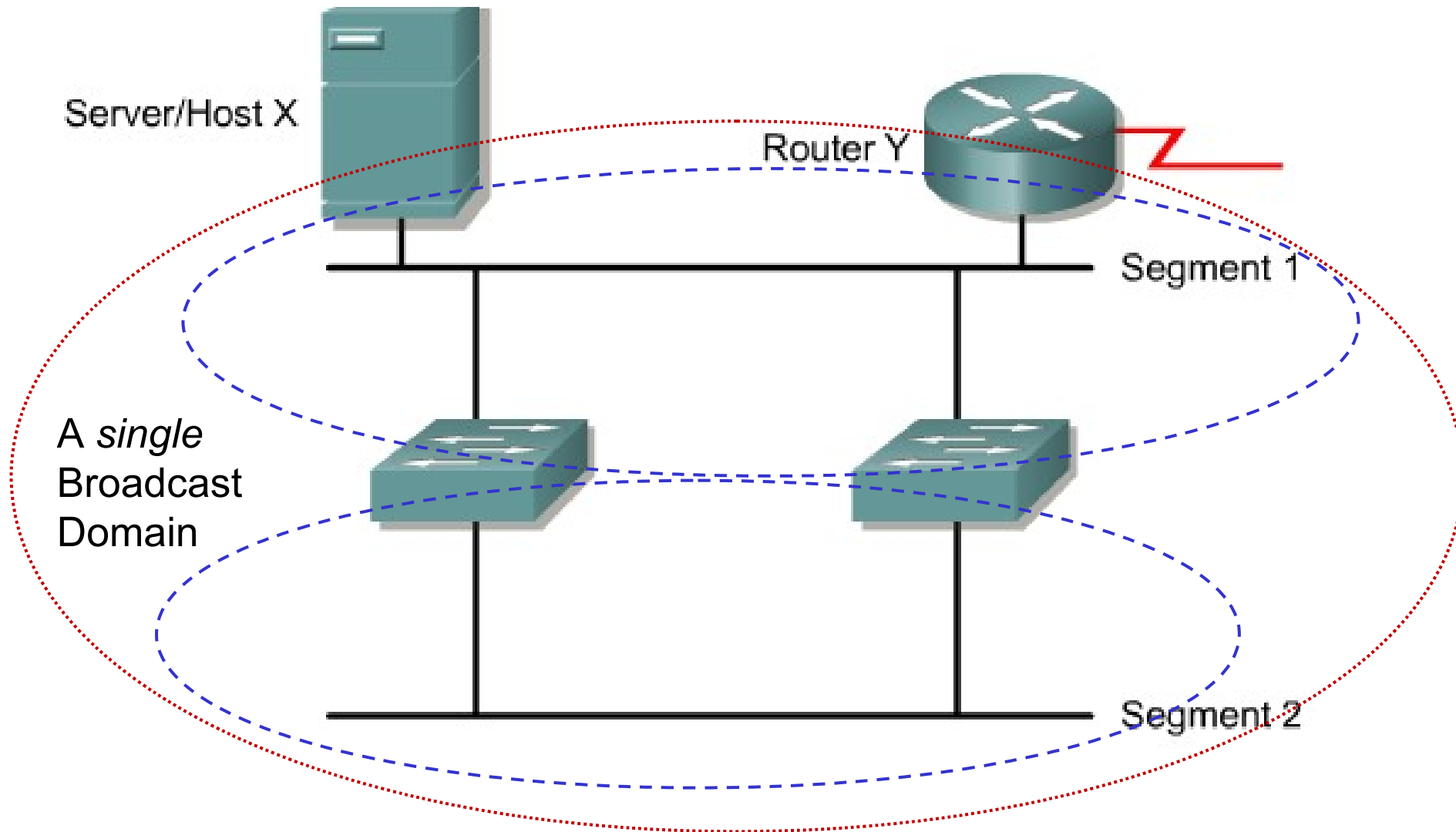
# **Spanning Tree Protocol**

**David Bray**

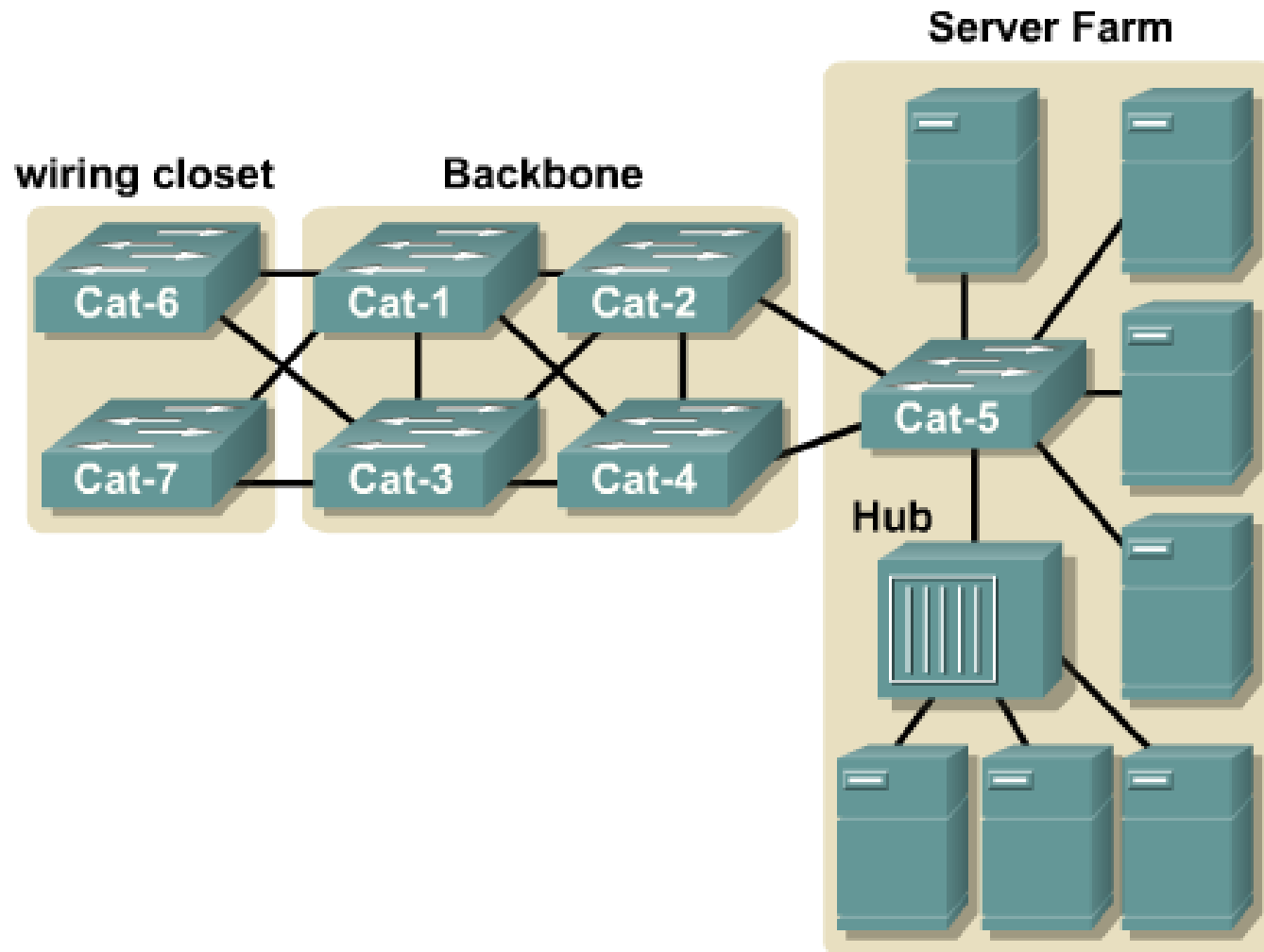
**brayd@algonquincollege.com**

with contributions obtained from Rick Graziani & Cisco

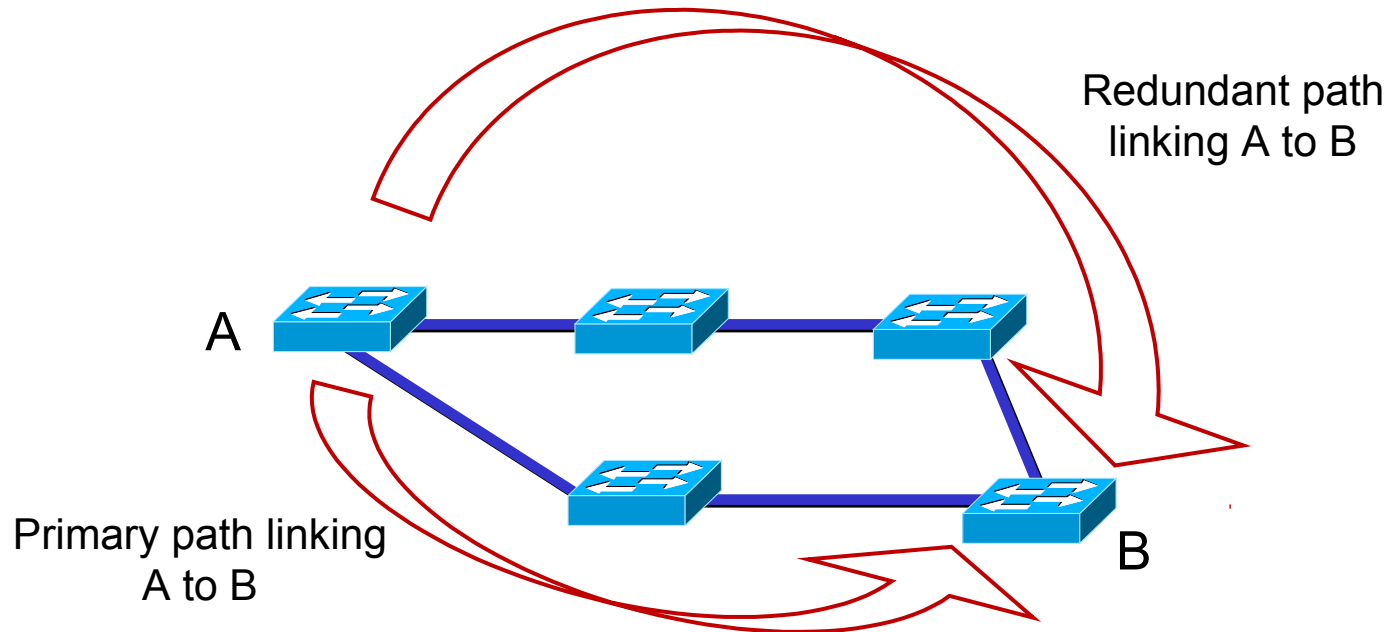
# Simple Redundant Switched Topology



# Using Bridging Loops for Redundancy



# Looped Paths Between Switches



- A switched network is commonly designed with *redundant paths* in order to provide fault-tolerance (i.e. eliminate any single point of failure), which gives rise to one or more *looped paths* for frames.

# Loops by Mistake

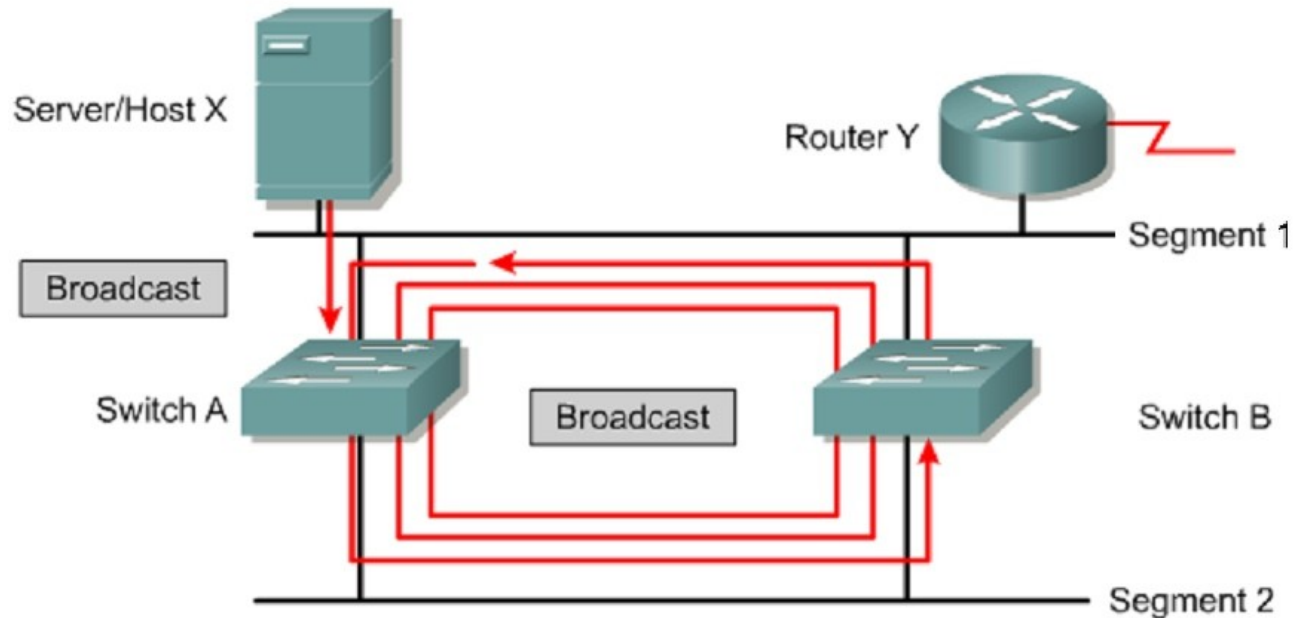


- Even if there are no deliberate loops for redundancy, loops can be mistakenly introduced.

# Layer 2 Loops Can Cause Problems

- Although redundancy at Layer 2 is desirable for fault tolerance, this can give rise to several problems:
  1. Broadcast Storms
  2. Reception of Duplicate Frames
  3. MAC Table Instability/Corruption

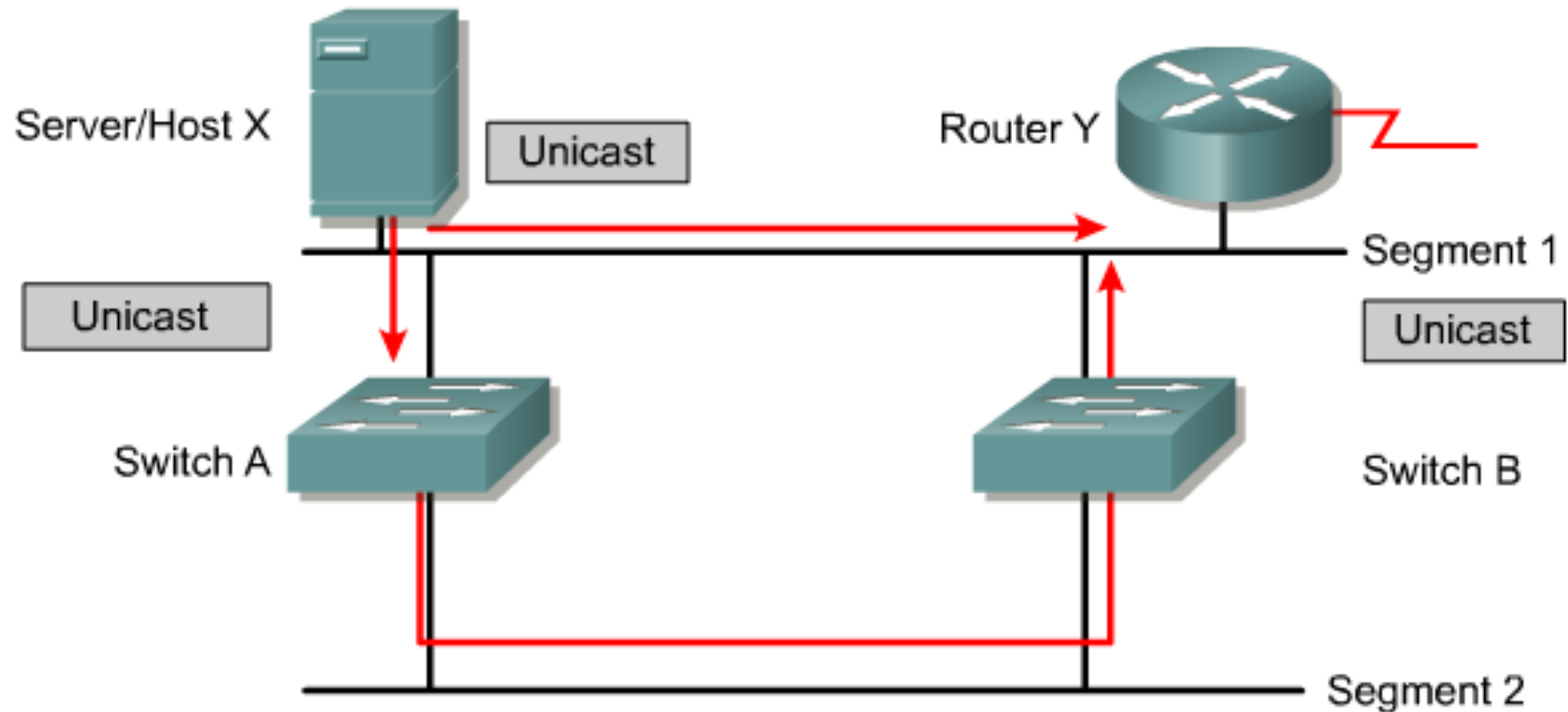
# 1. Broadcast Storm



- Host X sends a broadcast.
- Switches continue to propagate broadcast traffic over and over.

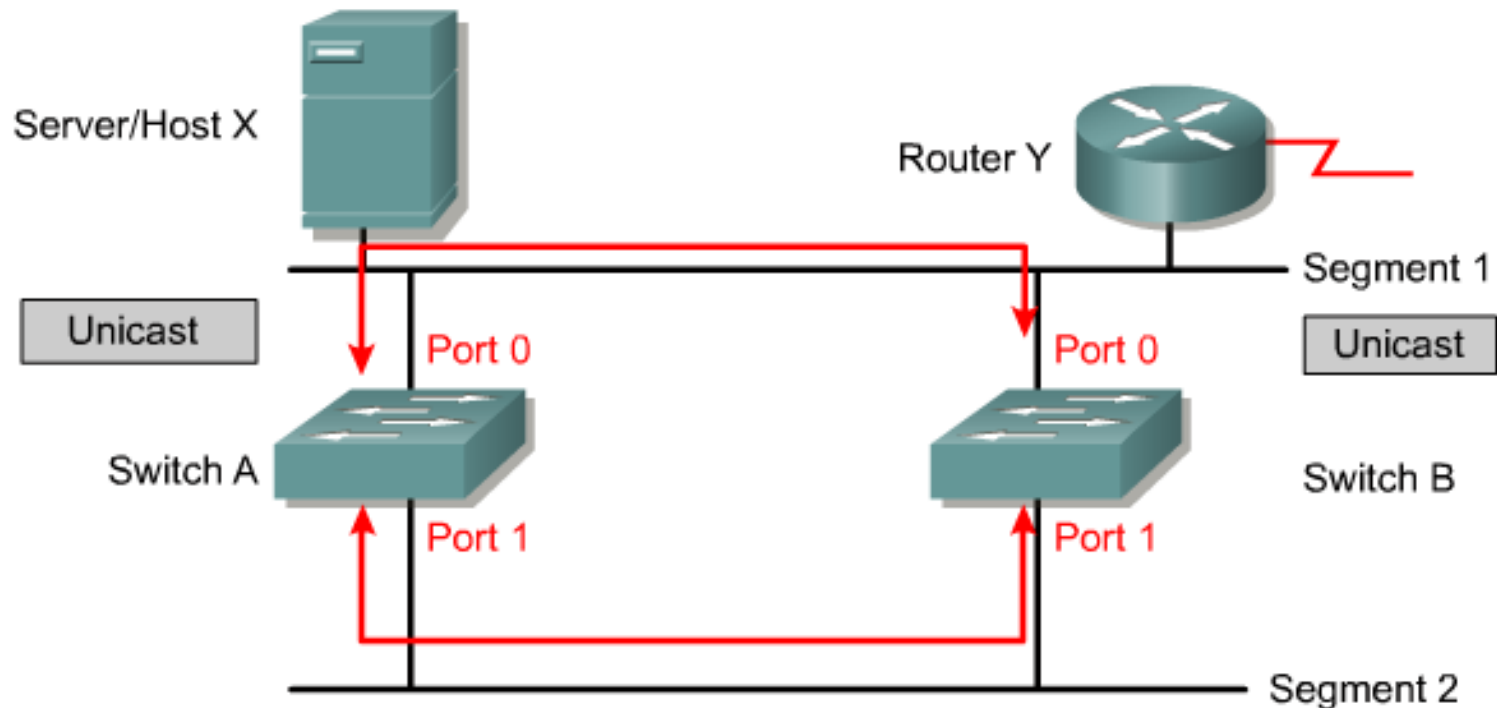
- broadcasts grow exponentially
- more dangerous than a routing loop since Ethernet frame has no TTL

## 2. Reception of Duplicate Frames



- Unicast frame from Host X to Router Y (who receives first frame directly)
- Switch A hasn't learned MAC for Router Y, so floods frame onto Segment 2
- Switch B unknowingly forwards a duplicate frame onto Segment 1

# 3. MAC Table Instability/Corruption

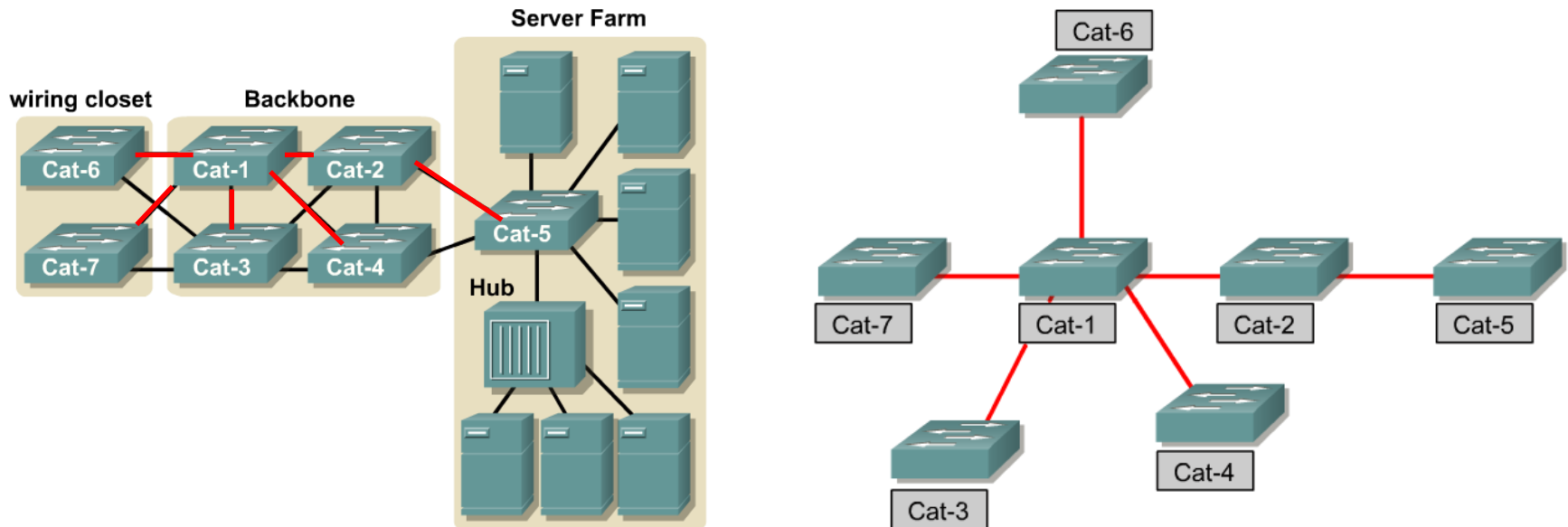


- Unicast frame from Host X to Router Y arrives at Switch A who hasn't yet learned MAC for Router Y; frame flooded onto Segment 2
- Switch B hears frame from Host X and associates Host X with Port 0
- Shortly afterwards, frame from Switch A causes Switch B to re-associate Host X with Port 1 (depending on traffic and aging timers, this could repeat)

# Solution: STP

- Allow physical loops for redundancy, but constrain frame forwarding to a logical tree imposed on the physical links.
- Spanning Tree Protocol (STP) was originally developed by DEC
- extended & revised by IEEE and standardized as 802.1d
- Other related IEEE Standards:
  - 802.1w Rapid STP
  - 802.1s Multiple STP  
(derived from Cisco's Multiple Instance STP)

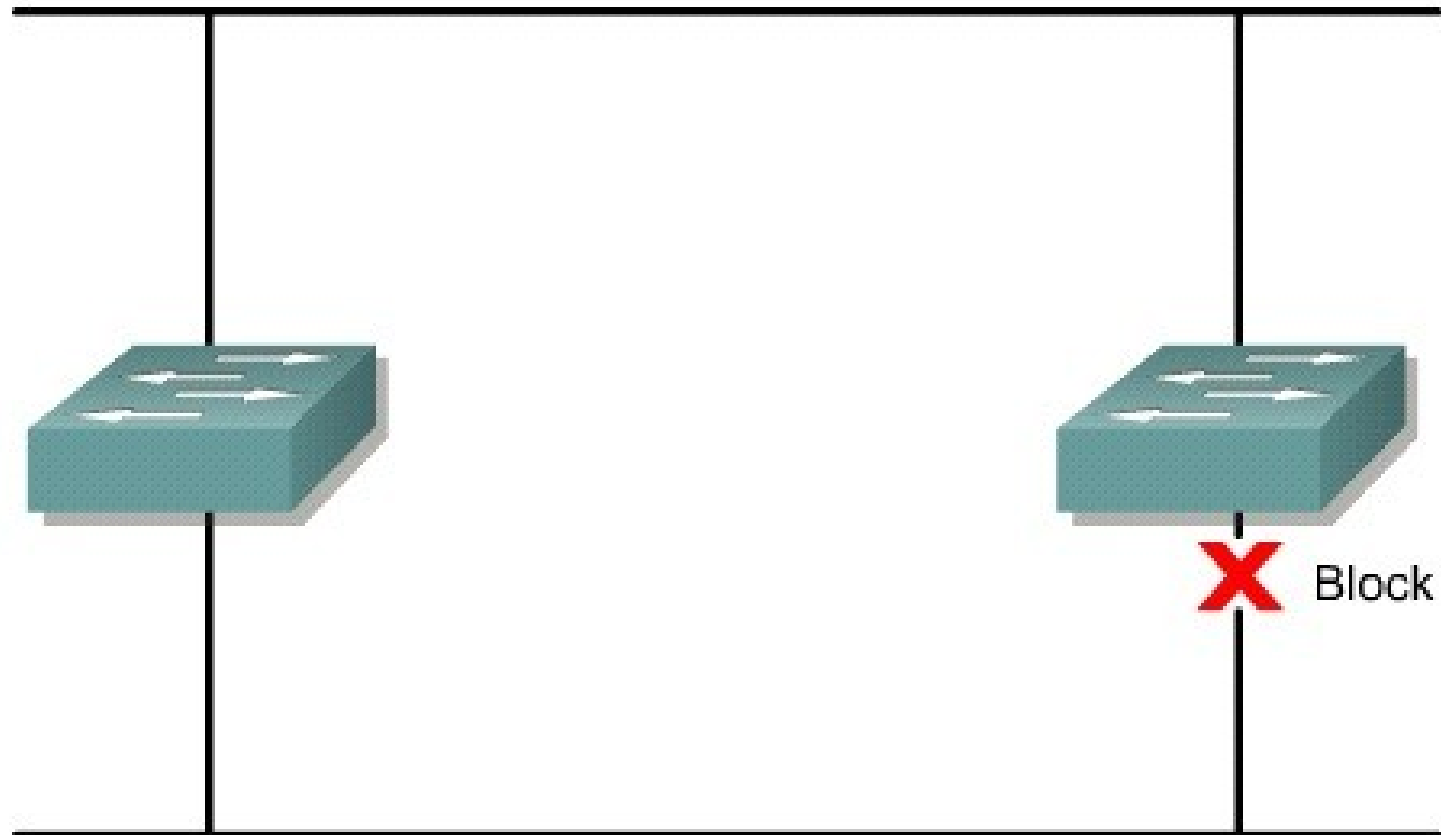
# Redundancy Using Spanning Tree



- It is called *spanning tree* because all devices in the network are reachable or *spanned*.
- STP (802.1d) can take a “relatively” long time to converge (30-50 secs).

*Rapid* STP (802.1w) reduces the time to (re-)compute a loop-free logical topology to 15 secs or less.

# Spanning-Tree Protocol



Provides a loop-free redundant network topology by placing certain ports in the blocking state.

# Concept Behind STP

- 1. Designate a primary switch – Root Bridge**
- 2. From that switch, calculate the single lowest cost path to every other switch in that broadcast domain (i.e. build a *Tree*).**
- 3. Only allow frames to traverse these paths (i.e. block frame forwarding via any other paths)**

- This guarantees that there will no loops in the active topology.
- In this section we will use the terms bridge/switch interchangeably, and as always, every reference to a "switch" should be taken to mean a layer 2 device unless stated otherwise.

# Some Observations

- This strategy doesn't necessarily result in the shortest/quickest path from one switch to the next
  - network design & selection of Root Bridge is critical (*how to force Root later*)
- Removal of loops also removes path redundancy (there's only a single path from the root to every non-root device)
  - need a way to detect link/device failure & recalculate these paths, enabling some of the *previously blocked* redundant paths

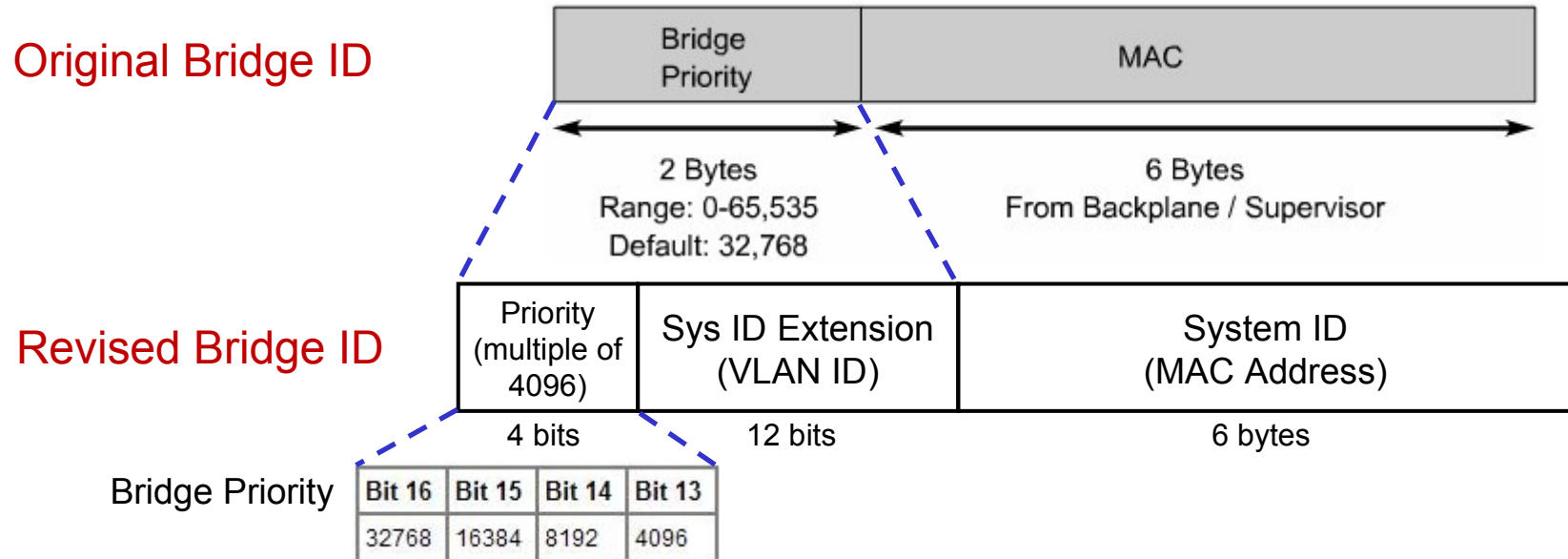
# Requirements to Implement STP

- Devices need to know about each other.
- Devices need to agree on who is the Root Bridge and how it's selected.
- Devices need to know when an active link has failed (consistent timer values).
- Devices need to agree on link cost valuations.
- Devices need a way of blocking non-chosen paths (subject to recalculation).
- Each device or segment (i.e. collision domain) must have a single active path to the Root Bridge.

# The Ideal Root Bridge

- The device *chosen* as Root Bridge should:
  - have a *central* position in the topology (network radius should be 7 or less)
  - be a high-end device
    - lots of processing power
    - ample memory
    - good fault tolerance (e.g. redundant PSUs)
  - have access to high bandwidth links
  - be located where it will be noticed (LEDs visible, should things go badly)

# 8-Byte Bridge ID (BID)



- Each switch/bridge is uniquely identified by a Bridge ID. The updated format provides uniqueness across VLANs.  
SW(config)# spanning-tree extend system-id
- The Bridge ID is also used to elect a *Root Bridge*.
- The default Bridge Priority of 32768 (4-bit value 1000) is often *lessened* by the network administrator in order to force a specific device to be selected as the Root Bridge.

# Quick Peek at STP Information

```
ALSwitch#show spanning-tree
```

```
VLAN0001
```

```
Spanning tree enabled protocol ieee
```

```
Root ID      Priority 32768
Address      0003.e334.6640
Cost         19
Port         23 (FastEthernet0/23)
Hello Time   2 sec Max Age 20 sec Forward Delay 15 sec
```

**BID of Root Bridge**

**Root Port**

**Hello Interval**

```
Bridge ID   Priority 32769 (priority 32768 sys-id-ext 1)
Address     000b.fc28.d400
Hello Time  2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time  300
```

**This device's BID  
(note VLAN 1)**

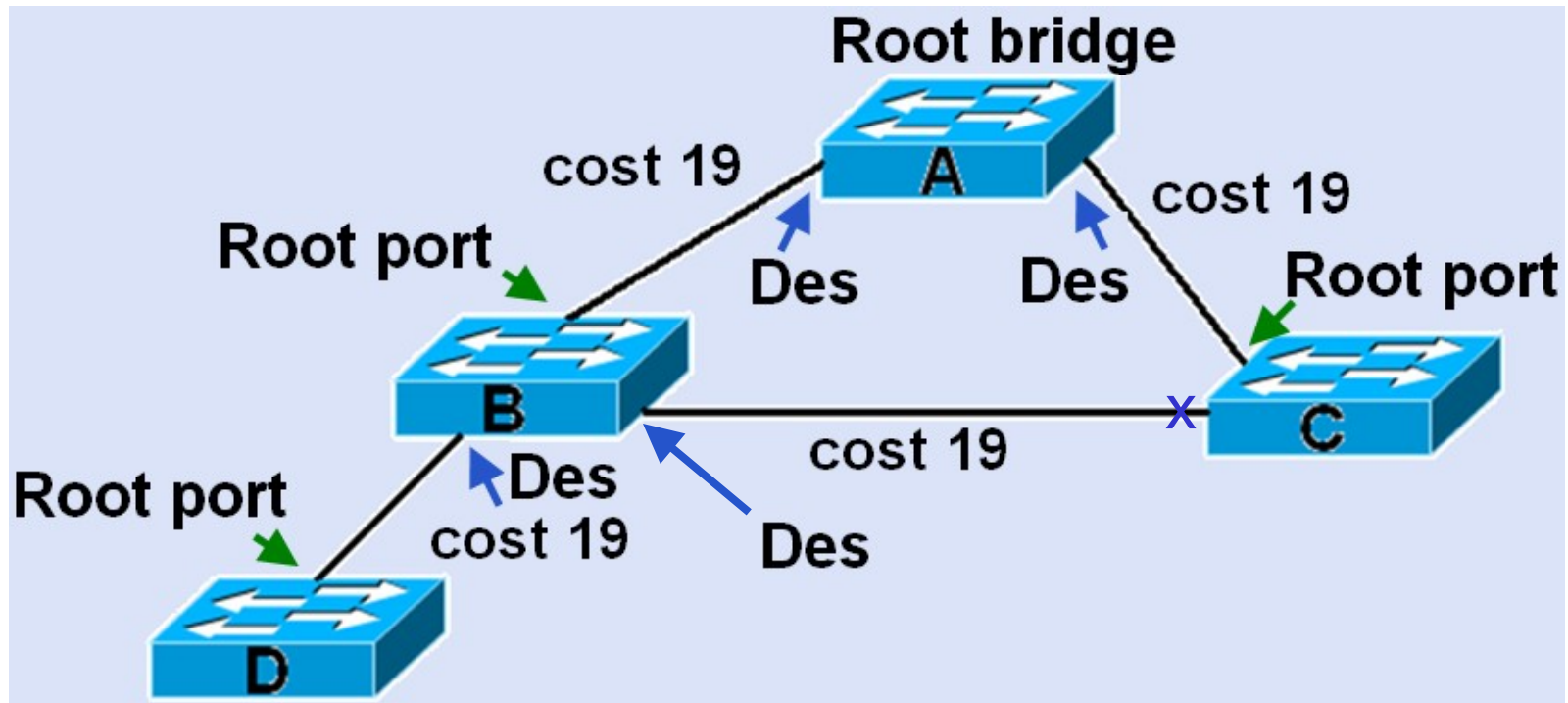
Interface Name	Port ID Prio.Nbr	Designated Cost Sts	Port ID Prio.Nbr
		Cost Bridge ID	
Fa0/23	128.23	19 FWD 0 32768 0003.e334.6640	128.25

ALSwitch#

# Definitions

- ***Root Port***: the port on a non-root bridge that is closest to (and sends frames to), the Root Bridge
  - exactly one per non-root device
  - BPDUs should be received regularly on this port
- ***Designated Port***: a port responsible for propagating frames from the Root Bridge onto an inter-switch segment
  - one per inter-switch segment
- ***Designated Bridge*** (for a segment): the device housing the *Designated Port* for that segment
  - one per inter-switch segment

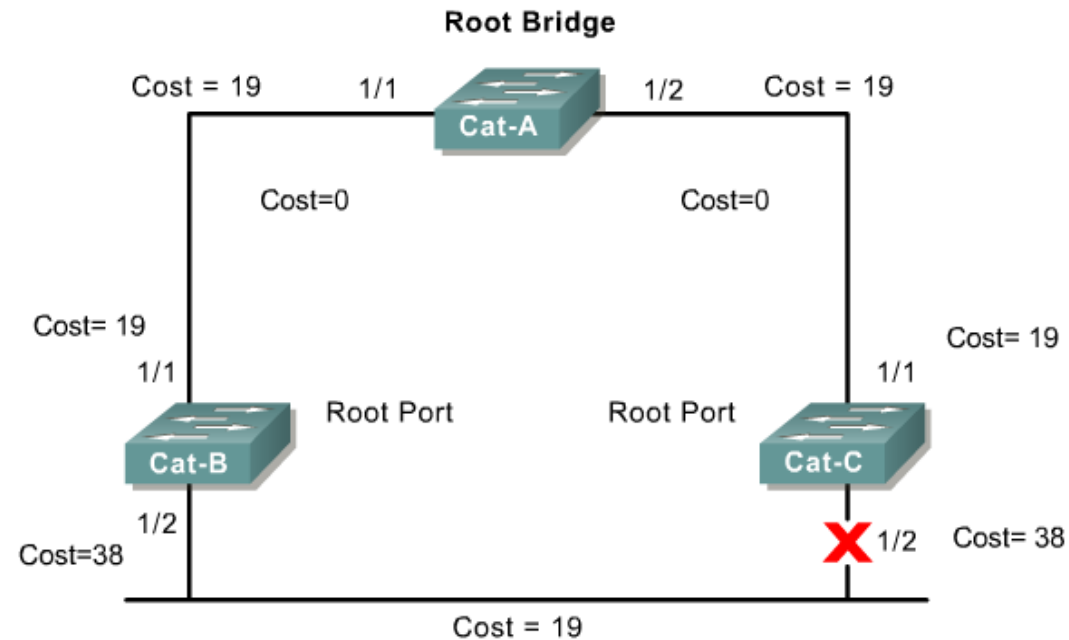
# Quick Example of Port Roles



- Each non-root device has exactly 1 root port
- Each inter-switch segment has 1 designated port

# Spanning Tree Operation

1. One **root bridge** is elected.
2. On each non-root switch, elect one **root port**.
3. On each inter-switch segment, elect a **designated port**.
4. All other inter-switch ports are **blocked**.



# 1. Electing a Root Bridge (and BPDUs)

- Bridges communicate using Layer 2 multicast frames (DA=01.80.C2.00.00.00) called Bridge Protocol Data Units (BPDUs - key fields on next slide).
- Upon initialization, each bridge sources Configuration BPDUs at the Hello interval (every 2 sec by default) with its own BID as the Root BID.
  - As all bridges do this, it is called a "root war".
- Rules for sending BPDUs:
  - If a bridge receives a BPDU with a lower value than its own, it stops sourcing its own BPDUs but instead, adopts and propagates this lower (i.e. *better*) one.
- Eventually, the device with the *lowest* BID wins the war and becomes the Root Bridge.
- To remember that lower values win, picture a switch with a golf club through it!

# Data Fields in a BPDU

- The *best* BPDU is the one with the lowest (in order):  
Root BID, Root Path Cost, Sender BID, Sender Port ID

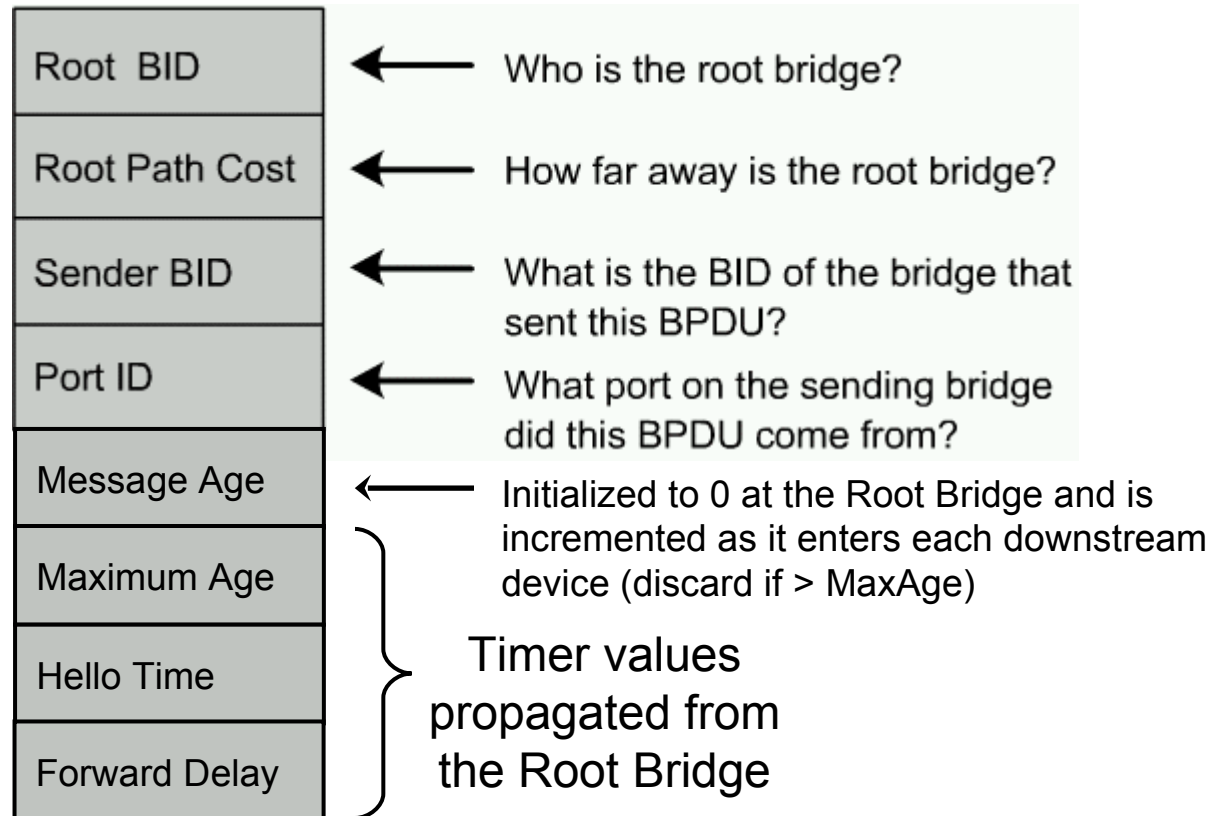
Key Data Fields  
Contained in a BPDU

1-byte Port Priority +  
1-byte Port Number

default = 20 secs

default = 2 secs

default = 15 secs



# Affecting Root Bridge

- Configure bridge priority:

```
SW1 (config) #spanning-tree vlan vlan-id priority  
                                                    { 0-65535}
```

*(if “extended system ID” is enabled, only multiples of 4096 are valid)*

- Configure as primary root bridge:

```
SW1 (config) #spanning-tree vlan vlan-id root primary  
(indirectly sets bridge priority to 24576 or 4096 less than lowest value)
```

- Configure as secondary root bridge:

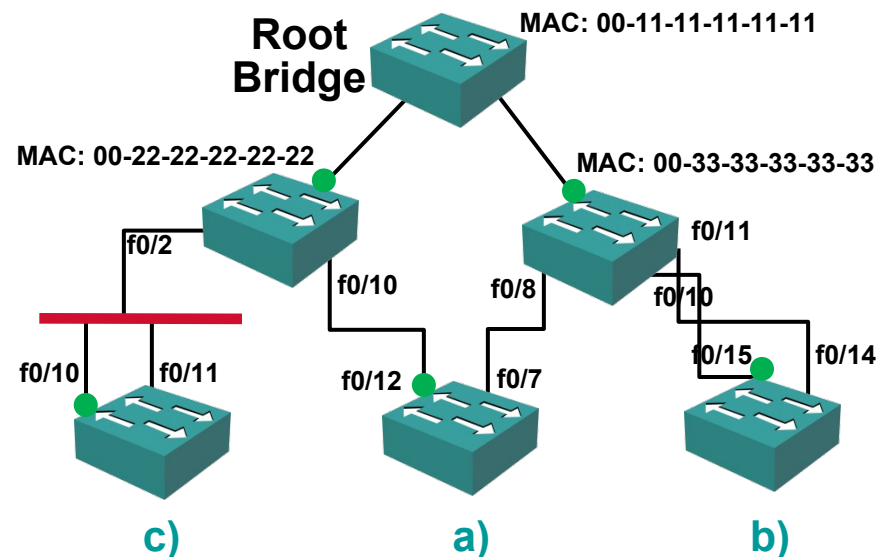
```
SW1 (config) #spanning-tree vlan vlan-id root secondary  
(indirectly sets bridge priority to 28672)
```

## 2. Elect Root Port (one per device)

- As before, the BPDUs received at every non-root device are used for this election.
- Every non-root device elects its *Root Port* as the one with the lowest “root path cost”.
- In the event of multiple least-cost root paths, the following aspects are compared **top-down**, until we break the tie:
  - a) lowest Sender Bridge ID,
  - b) then lowest Sender Port ID,
  - c) then lowest Local Port ID

### Diagram Notes:

- All links are equal cost
- All bridge priorities and port priorities at default
- Chosen *Root Ports* marked as ●



# Spanning Tree Link Costs

Link Speed	Cost(Revised IEEE Spec)	Cost (Previous IEEE Spec)
10 Gbps	2	1
1 Gbps	4	1
100 Mbps	19	10
10 Mbps	100	100

Diagram illustrating the relationship between link speed and cost. A bracket groups the 'Cost (Previous IEEE Spec)' column for 10 Gbps, 1 Gbps, and 100 Mbps, pointing to a box containing the formula  $\frac{1000}{\text{Bandwidth}}$ .

- STP costs are represented as unsigned 4-byte values.
- BPDUs are sourced from the Root Bridge with a Root Path Cost of zero.
- As a BPDU is received, the bridge increments the Root Path Cost field by the cost of the link on which the BPDU arrived. (Refer to above chart.)
- The value of Root Path Cost from within any device, is the accumulated cost to send a frame to the Root Bridge from that device.

# Setting Port (Link) Cost & Priority

- Configure port (i.e. link) cost:

```
SW1(config-if)#spanning-tree cost {1-200000000}
```

- Reverting to the default cost:

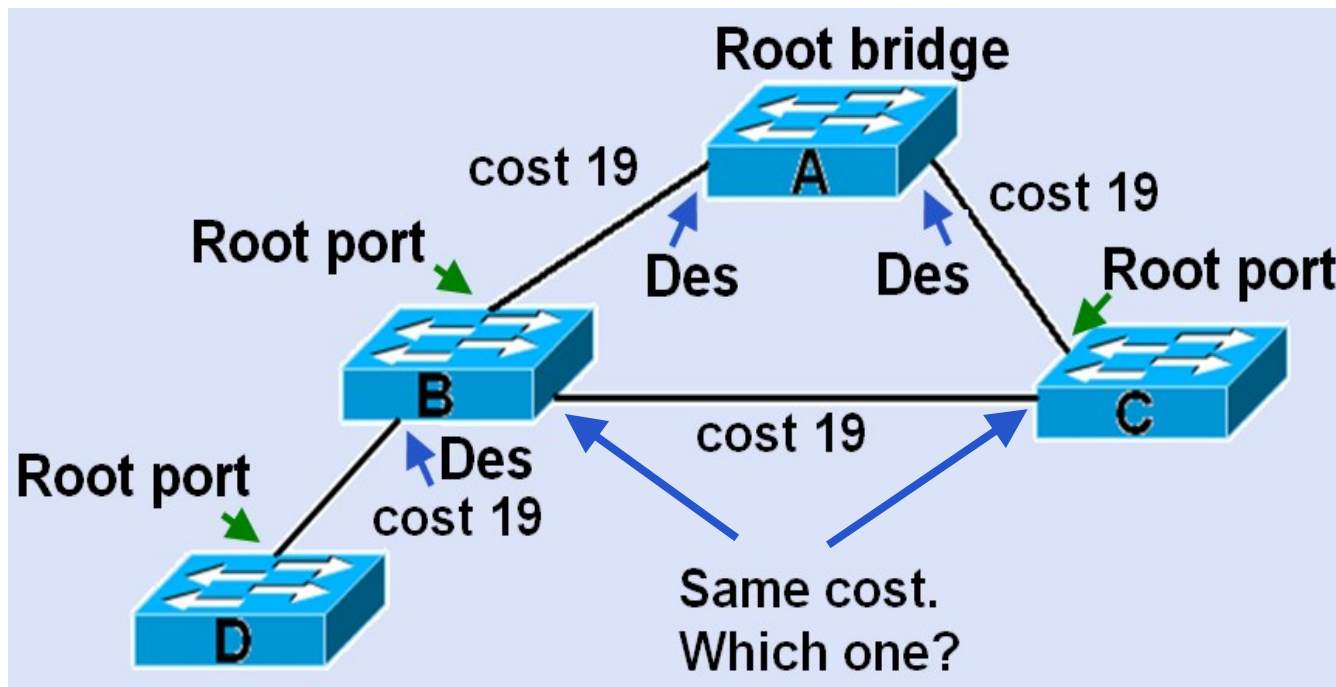
```
SW1(config-if)#no spanning-tree cost
```

- Configure port priority (default is 128):

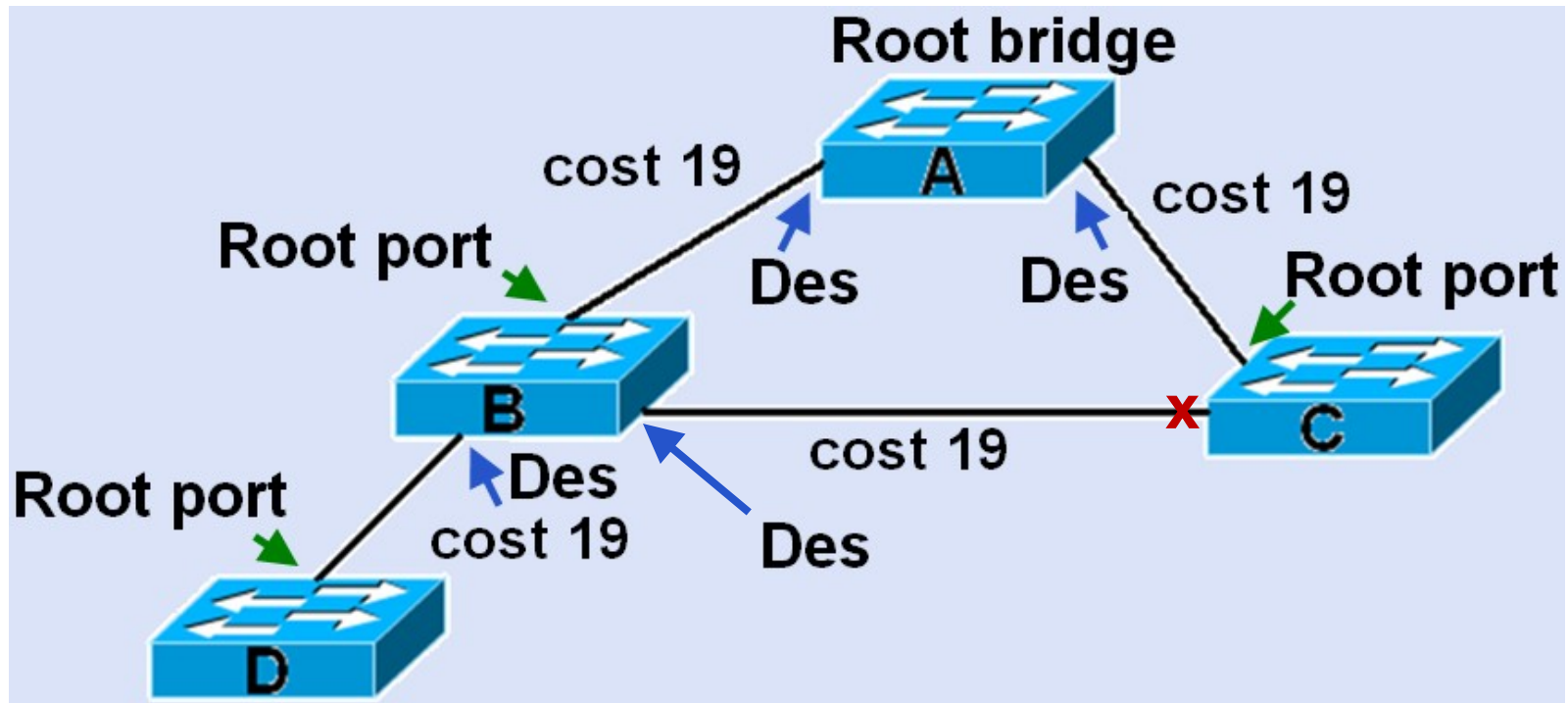
```
SW1(config-if)#spanning-tree port-priority {0-240}  
(in increments of 16)
```

### 3. Electing Designated Ports

- On every inter-switch segment, the port with the lowest *root cost path* becomes the designated port.
- In the event of a tie between ports, the one from the lowest BID (then lowest port ID) wins out.



## 4. Blocking Other Inter-Switch Ports



- All inter-switch ports neither elected *root* nor *designated* are now **blocked**. (Sw# sh span blocked)
- The LED on such switchports will be amber (unless they are trunks, in which case it shows trunk status).

# Timer Values

- Each bridge uses the timer values propagated from the Root Bridge.
  - Although timer values can be set individually at each device, these will not be used unless the device becomes the Root.
- Default MaxAge of 20 secs is based on **7** segment hops (of up to 2 secs each) and 3 lost Hellos.
  - **Root Bridge plus 6 more** devices in any one branch
- Transition from *blocking* to *forwarding* could take up to 50 secs:
  - 20 secs (aging) + 15 secs (listening) + 15 secs (learning)

# STP Information Revisited

```
ALSwitch#show spanning-tree
VLAN0001
```

```
Spanning tree enabled protocol ieee
```

```
Root ID      Priority      32768
             Address      0003.e334.6640
             Cost        19
             Port        23 (FastEthernet0/23)
```

```
Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
```

**Timers**

```
Bridge ID  Priority      32769 (priority 32768 sys-id-ext 1)
           Address      000b.fc28.d400
```

```
Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
```

```
Aging Time 300
```

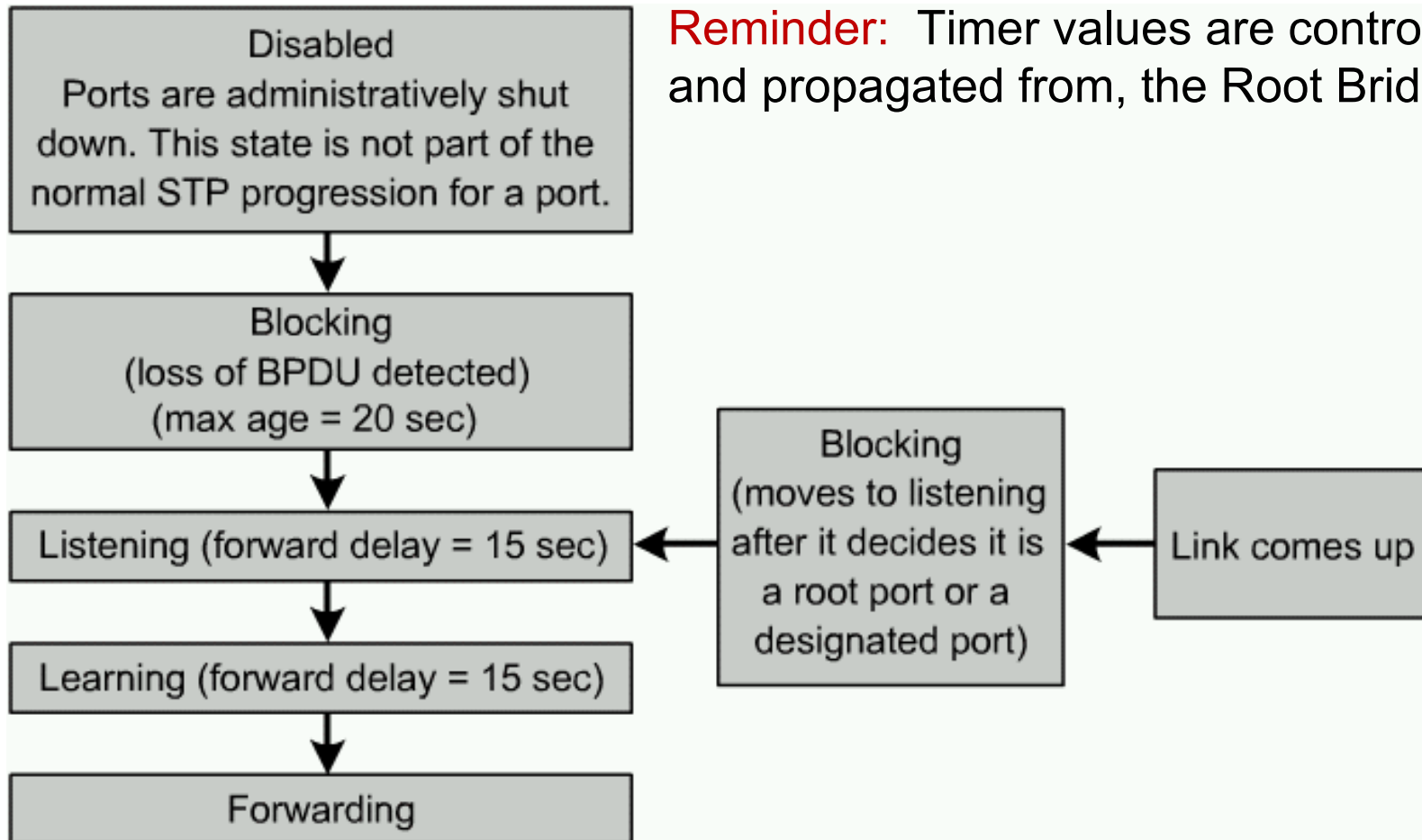
**MAC IDs aged out after 5 mins (the default)**

Interface Name	Port ID Prio.Nbr	Cost	Sts	Designated Cost	Bridge ID	Port ID Prio.Nbr
Fa0/23	128.23	19	FWD	0	32768 0003.e334.6640	128.25

**Root Port**

**Designated Port at other end**

# Spanning Tree Port States

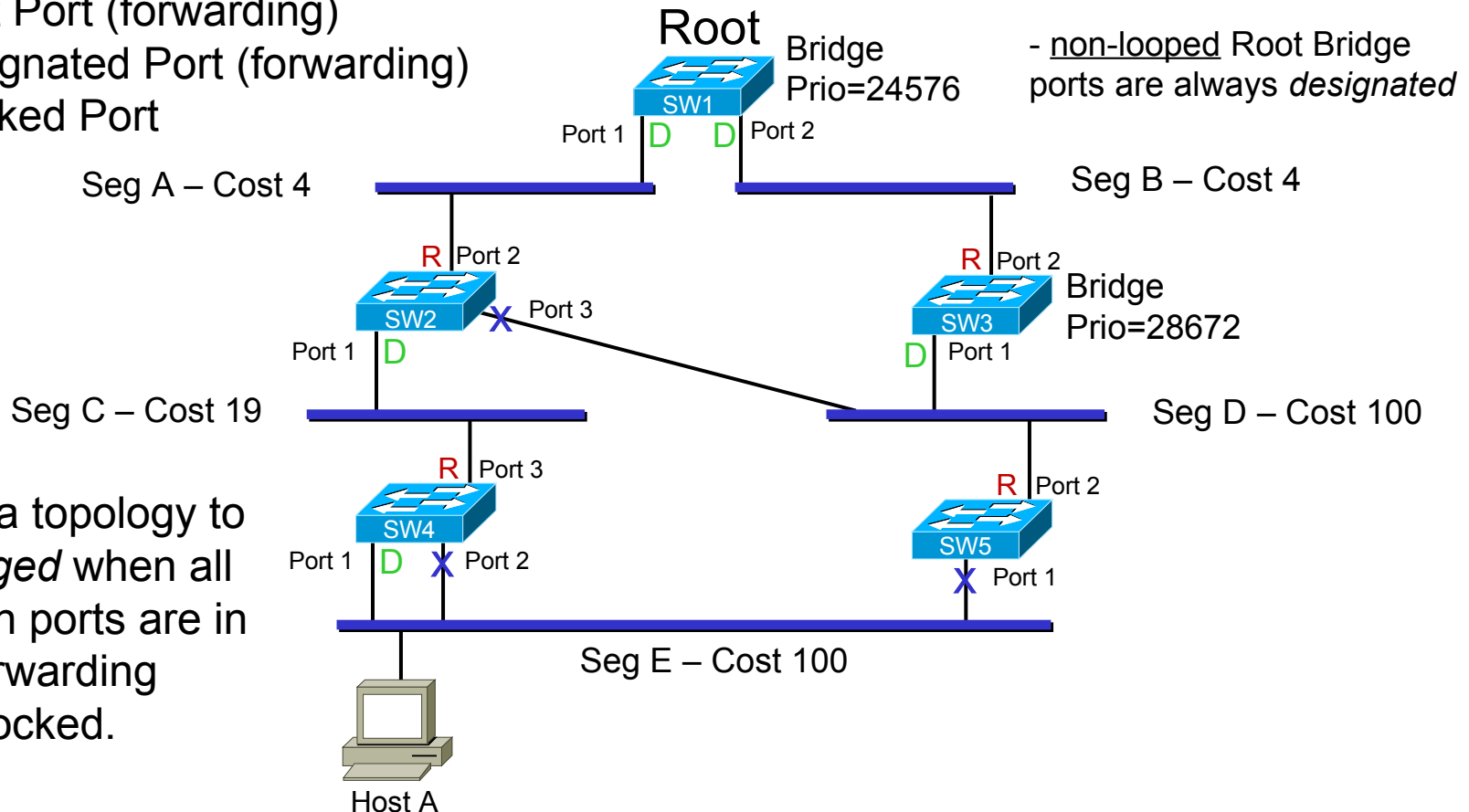


# STP Port State Descriptions

- **Disabled** – inoperative due to error, failure or administrator action
- **Blocking** – only receiving (and processing) BPDUs
- **Listening** – sending/receiving BPDUs, waiting (forward delay) for topology discovery to complete
- **Learning** – waiting (forward delay) and learning MAC addresses in order to minimize impact of flooding (propagating received BPDUs)
- **Forwarding** – normal operation forwarding frames and learning MAC addresses (propagating received BPDUs)

# Converged Topology (Switch to Hub)

- R – Root Port (forwarding)
- D – Designated Port (forwarding)
- X – Blocked Port

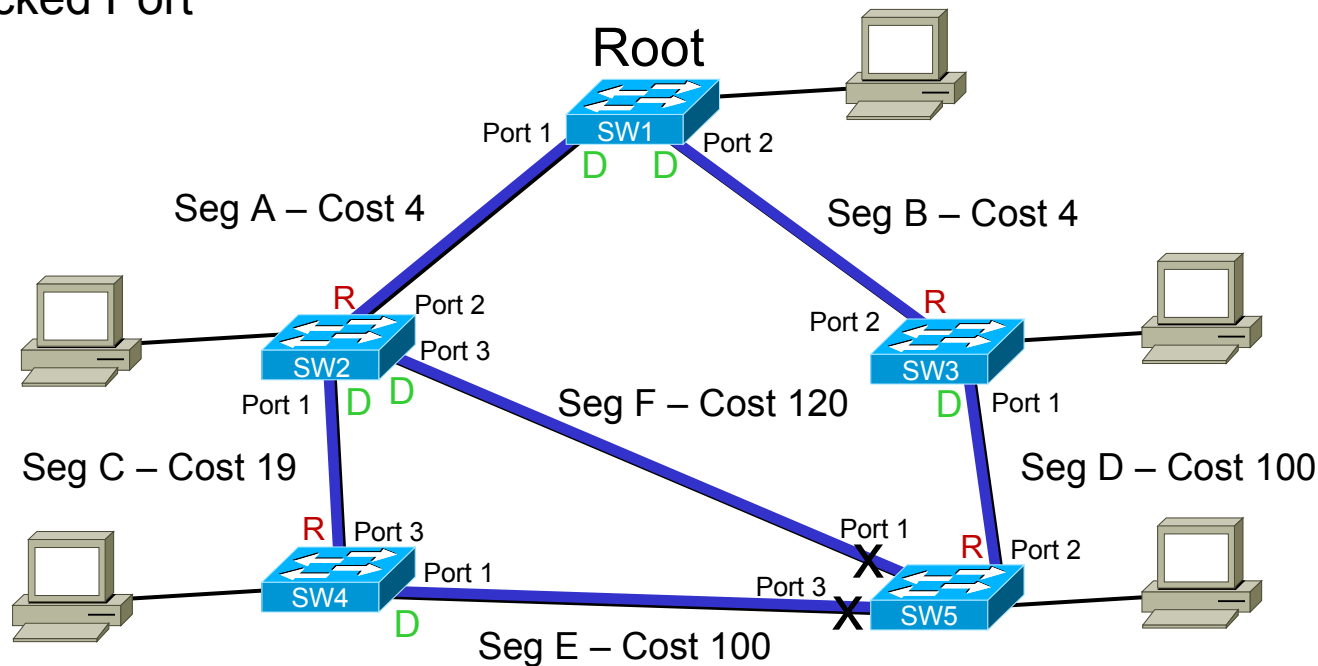


We deem a topology to be *converged* when all inter-switch ports are in either a forwarding state or blocked.

**n.b.** In this example, assume the MAC ID of SW(n) is less than that of SW(n+1) and that all priorities are at their defaults, unless otherwise shown.

# Converged Topology (Fully Switched)

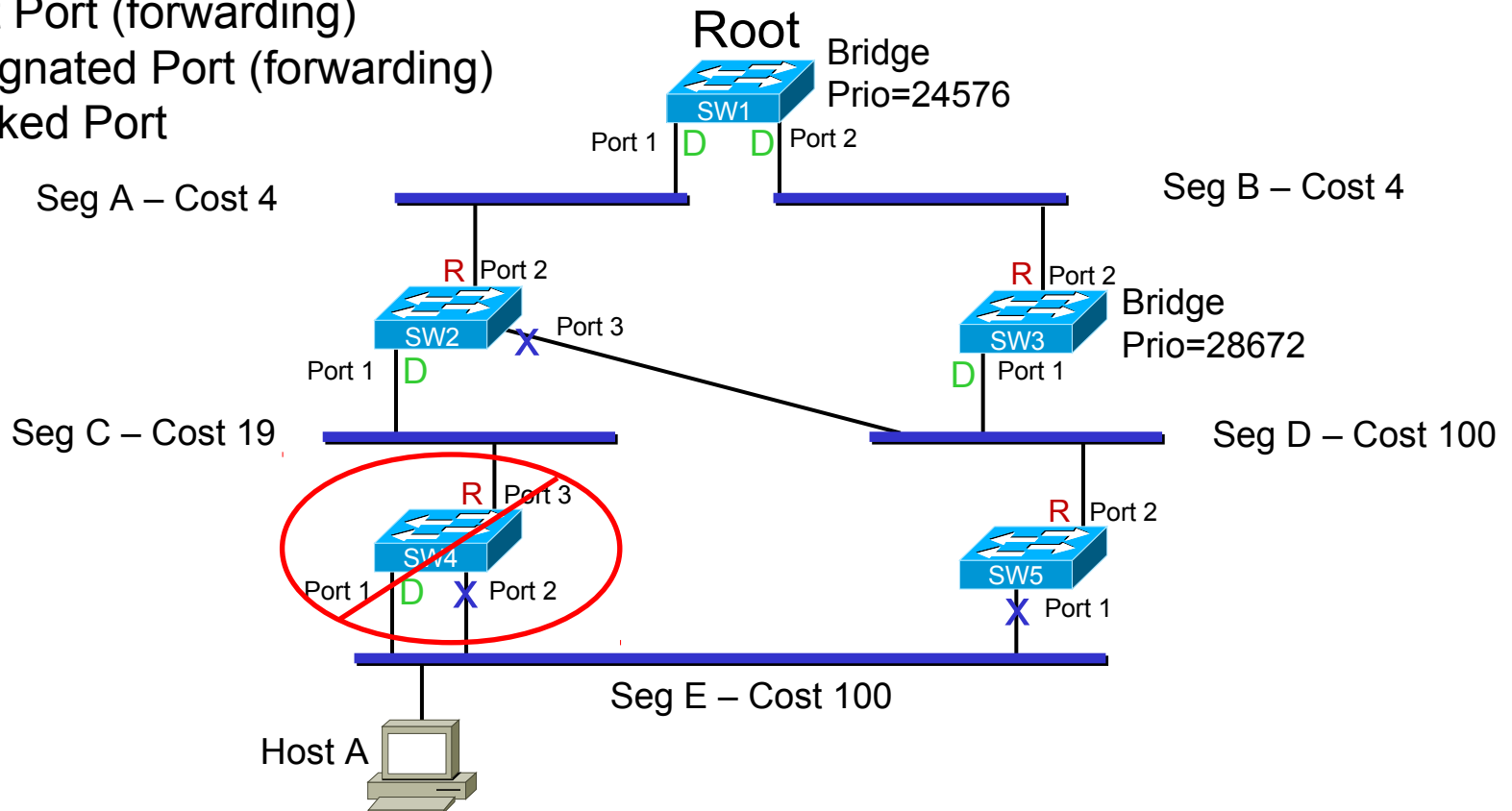
- R – Root Port (forwarding)
- D – Designated Port (forwarding)
- X – Blocked Port



n.b. In this example, assume the MAC ID of SW(n) is less than that of SW(n+1) and that all priorities are at their defaults, unless otherwise shown.

# Device Failure

- R – Root Port (forwarding)
- D – Designated Port (forwarding)
- X – Blocked Port



- Failure of device SW4 triggers a Topology Change Notification at SW5, which stops receiving BPDUs on Port 1 (from SW4 Port 1).
- Seg E suffers a temporary loss of communications while the topology recovers. (20 + 15 + 15 = 50 secs)

# Failure of Designated Port

Upon failure of the Designated Port serving a segment (whether the port, link or Designated Bridge), there are two possibilities:

1. if no other switch ports are eligible to serve that segment, connectivity is lost until the fault is corrected
2. if one or more other service-eligible switch ports exist, they must be Blocking (**why?**) ...
  - a. these blocking ports will no longer be receiving the Config-BPDUs normally propagated by the Designated Bridge, leading to a timeout after MaxAge (20 secs)
  - b. all such ports then compete to serve the segment as they did initially:
    - i. ports move to **Listening** state, each propagating their saved *best* BPDU onto the segment (15 secs)
    - ii. the port having the superior BPDU will become Designated, move to **Learning** state (15 secs) and then **Forwarding**
    - iii. the new Designated Bridge originates a **TCN** to the Root Bridge

# Topology Change Notification (TCN)

- **During normal operations**, the root bridge generates Configuration BPDUs (message type 0x00) every *Hello* interval; all other switches propagate these out their designated ports but do not normally source their own.
- However, if **any switch senses a topology change**, it will source a Topology Change Notification (TCN) BPDU (message type 0x80) towards the root bridge (unless its root path has failed); any switch receiving a TCN (except the root) will propagate it out its root port.
- **Any switch receiving a TCN** (including the root) is required to acknowledge the originator by returning a Configuration BPDU with the Topology Change Acknowledge bit set (flags = 0x80).
  - Any switch sending a TCN will repeat it every *Hello* interval until an acknowledgement is received from its upstream neighbour.
- Upon receiving a TCN, the root bridge will set the Topology Change (TC) bit in its Configuration BPDUs (flags = 0x01) for the next  $\text{FwdDelay} + \text{MaxAge}$  secs (default 15+20), causing all switches to shorten their Bridge Table aging time to  $\text{FwdDelay}$  (from default of 300 sec), thereby flushing any learned MAC addresses not in active use.

# BPDU Packet Decode

Configuration BPDUs sourced from Root Bridge every 2 sec by default.

## 802.3 Header

Destination: 01:80:C2:00:00:00 *Mcast 802.1d Bridge group*  
 Source: 00:D0:C0:F5:18:D1 *MAC ID of source port*  
 LLC Length: 38

## 802.2 Logical Link Control (LLC) Header

Dest. SAP: 0x42 *802.1 Bridge Spanning Tree*  
 Source SAP: 0x42 *802.1 Bridge Spanning Tree*  
 Command: 0x03 *Unnumbered Information*

## 802.1 - Bridge Spanning Tree

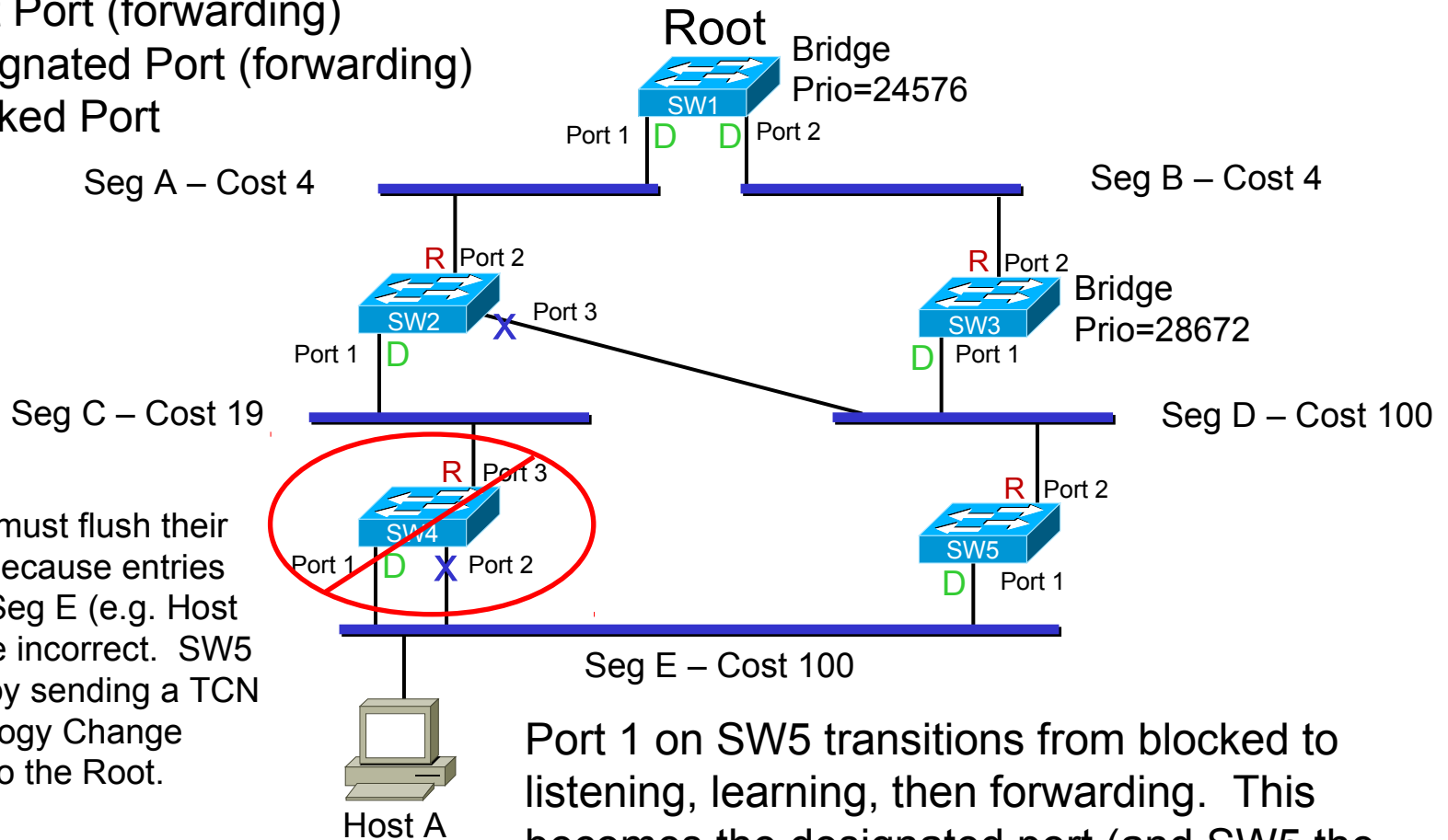
<b>Bytes</b>	2	Protocol Identifier:	0	<i>(always 0)</i>
	1	Protocol Version ID:	0	<i>(0 for STP, 2 for RSTP)</i>
	1	Message Type:	0	<i>Configuration BPDU</i>
	1	Flags:	%00000000	
	2+6	Root Priority/ID:	0x8000/ 00:D0:C0:F5:18:C0	
	4	Cost Of Path To Root:	0x00000000	
	2+6	Bridge Priority/ID:	0x8001/ 00:D0:C0:F5:18:C0	
	1+1	Port Priority/ID:	0x80/ 0x1D	<i>(128.29)</i>
	2	Message Age:	0/256 seconds	<i>(exactly 0 seconds)</i>
	2	Maximum Age:	5120/256 seconds	<i>(exactly 20 seconds)</i>
	2	Hello Time:	512/256 seconds	<i>(exactly 2 seconds)</i>
	2	Forward Delay:	3840/256 seconds	<i>(exactly 15 seconds)</i>

**Message Type 0x80 is a TCN BPDU (Topology Change Notification), in which case no more data fields will follow.**

**Topology Change = bit 0  
TC Acknowledge = bit 7**

# Spanning Tree Recalculation

- R – Root Port (forwarding)
- D – Designated Port (forwarding)
- X – Blocked Port



Port 1 on SW5 transitions from blocked to listening, learning, then forwarding. This becomes the designated port (and SW5 the designated bridge for Seg E), thereby restoring communications.

# Failure of Root Path

Upon failure of a device's Root Path (whether the port, link or upstream device):

1. it will no longer be receiving the Config-BPDUs originated by the Root Bridge, leading to a timeout after MaxAge (20 secs)
  - if the device can directly sense root port failure, it can transition to the next step without waiting for the MaxAge timeout period
2. the device discards its saved *best* BPDU
3. it must now choose another root port, which it does in the same fashion as during initial startup (BPDU with least root path cost)
  - once the new root port is determined, a new *best* BPDU is saved
  - **Note:** If the failure prevents all reachability to the Root Bridge, a new Root Bridge would be elected as a result.
4. as the new root port transitions to Forwarding, the device sources a TCN to the Root Bridge

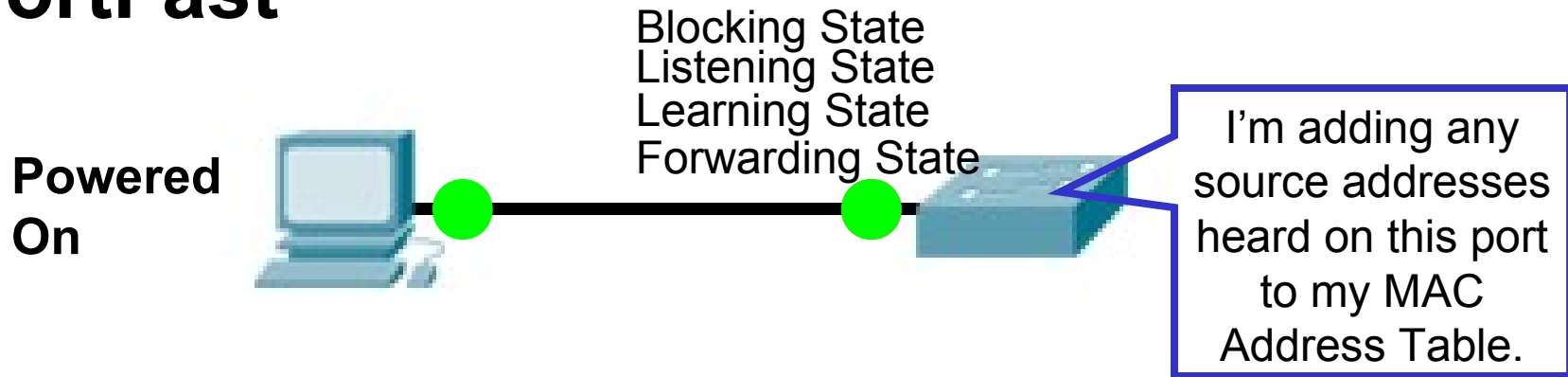
# Failure Scenarios

- In-class exercises to explore the impact of various failure scenarios
  - designated port failures
  - root port/path failures
  - device failures
- Ensure you have a detailed understanding of what happens in each case:
  - the sequence and timing of the events
  - which device issues a TCN and the consequence of that throughout the topology
  - what changes (if any) to the BPDUs propagated in the topology (and for what period of time)
  - how the topology re-converges

# Powercycle a host and watch link lights...

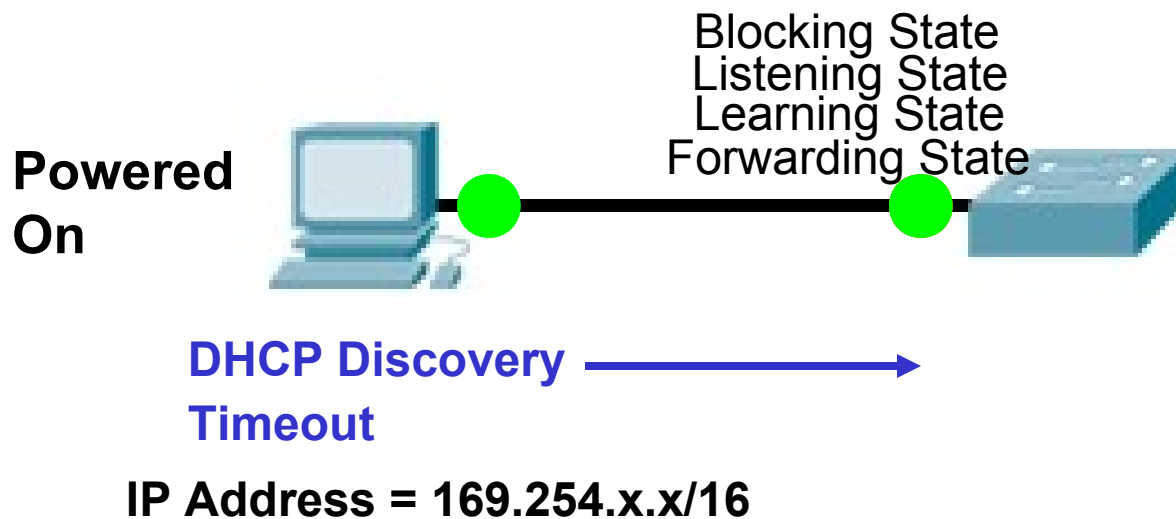
The screenshot displays a network simulation interface. On the left, the 'Physical Device view' shows a server rack with a power button and a 'LINK' indicator. Below it, a control panel includes a dropdown menu and a power button. On the right, a network topology diagram shows two switches: 'Distribution2' and 'Access2'. 'Distribution2' is connected to 'Access2' via a fiber link (G0/1 to Fa0/3). 'Access2' has several ports connected to hosts: Fa0/1 to a host in VLAN 20, Fa0/3 to another host in VLAN 20, Fa0/20 to a host in VLAN 20, and Fa0/10 to a host in VLAN 10. A blue arrow points to the Fa0/10 port on 'Access2', with the text 'How long until switch link light turns green?' next to it. The interface also includes zoom controls (Zoom In, Original Size, Zoom Out) and a scenario dropdown menu (Scenario 0) with 'New' and 'Delete' buttons.

# PortFast



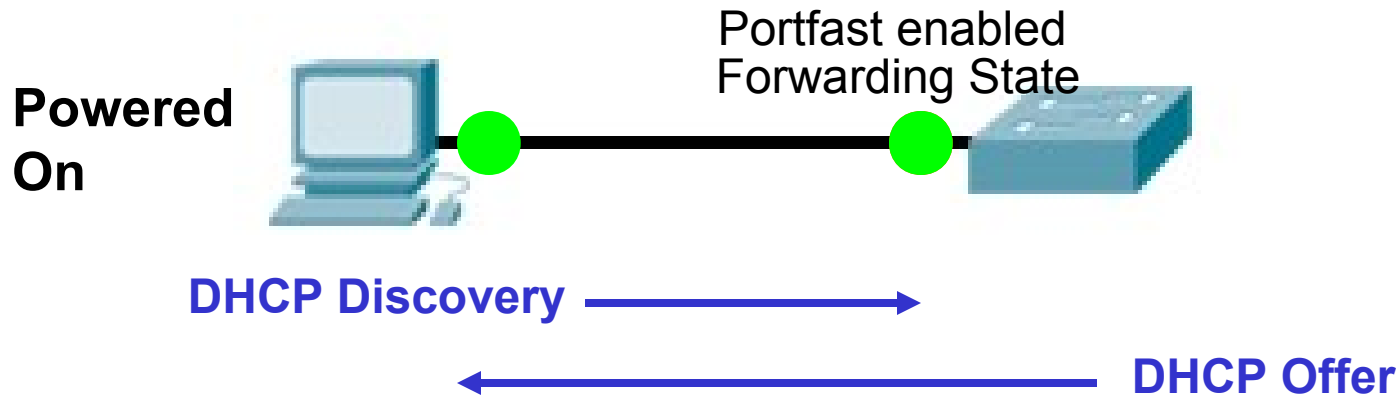
- Host powered on.
- Port moves from **blocking state** immediately to **listening state** (15 seconds).
  - Determines where switch fits into spanning tree topology.
- After 15 seconds port moves to **learning state** (15 seconds).
  - Switch learns MAC addresses on this port.
- After 15 seconds port moves to **forwarding state** (30 seconds total).

# PortFast – Problem DHCP



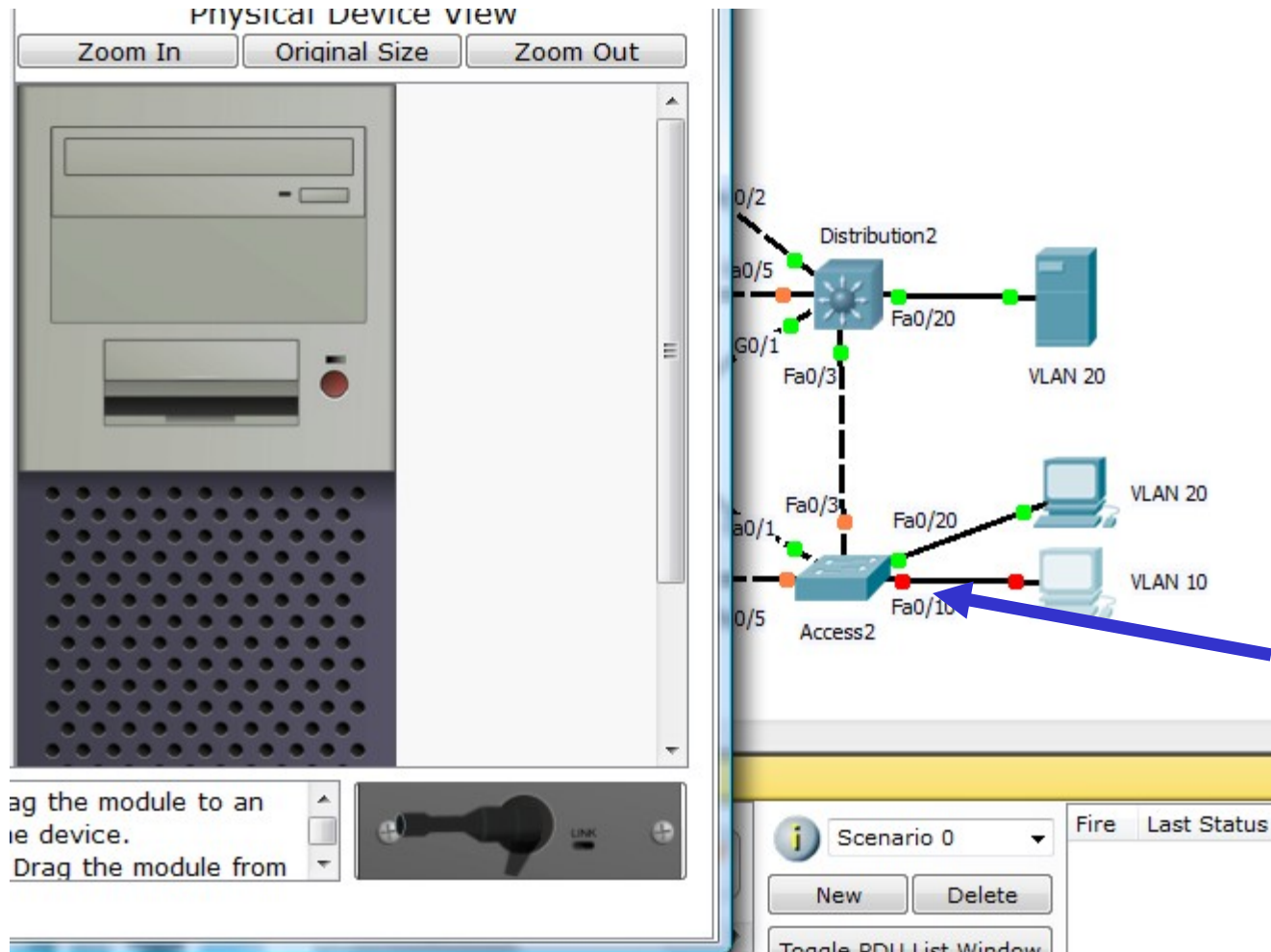
- Host sends DHCP Discovery
- Host never gets IP addressing information
- **Portfast also suppresses TCNs** to enhance stability and efficiency:
  - Through normal booting or shutdown of a user PC, the link goes up or down causing the switch to source a TCN.
  - Individually, there's no significant impact. But given enough hosts, the topology could be subjected to a large number of nuisance TCNs and unnecessary flushing of MAC address tables.
  - This leads to excessive flooding of “unknown” unicast frames.
  - With Portfast enabled, TCNs are NOT sent due to state changes on the port.

# PortFast



- The purpose of **PortFast** is to minimize the time that access ports wait for STP to converge.
- When a port comes up, the port immediately moves into the Forwarding state.
- The biggest advantage of enabling PortFast is to prevent **DHCP** timeouts.
- Host sends DHCP Discovery
- Host can now obtain appropriate IP addressing information.

# Powercycle the host again (portfast enabled)



# Configuring Portfast

```
Access2 (config) #interface range fa 0/10 - 24
Access2 (config-if-range) #switchport mode access
                        <Previously configured>
Access2 (config-if-range) #spanning-tree portfast
```

**OR**

```
Access2 (config) #spanning-tree portfast default
```

- **Warning:** PortFast should only be enabled on ports that are connected to a single host.
- If hubs or switches are connected to the interface when PortFast is enabled, temporary bridging loops can occur.
  - Unless “BPDU filter” is enabled, the port still participates in STP.
- If a BPDU is received on a Portfast configured port, it is moved immediately into **Blocking state**.
  - If “BPDU guard” is enabled, the port is placed into *errdisable* mode.
  - To resume operation, port must be bounced or wait a configured timeout.

# Verifying Portfast (Example)

```
Sw# show spanning-tree interface f0/6 portfast
VLAN0010          enabled
Sw#
```

- This output shows f0/6 only allows VLAN 10 and has Portfast enabled.

## Access Port Macro

```
Sw(config-if)# switchport host
switchport mode will be set to access
spanning-tree portfast will be enabled
channel group will be disabled
Sw(config-if)# end
Sw#
```

# BPDU Guard

- BPDU Guard puts an interface configured for STP PortFast into an *errdisable* state upon receipt of a BPDU. BPDU guard disables interfaces as a preventive step to avoid potential bridging loops.
- BPDU guard shuts down PortFast-configured interfaces that receive BPDUs, rather than putting them into the STP blocking state (the default behaviour). In a valid configuration, PortFast-configured interfaces should not receive BPDUs. Reception of a BPDU by a PortFast-configured interface signals a mis-configuration or the connection of an unauthorized device.
- BPDU guard provides a secure response to invalid configurations, because the administrator must manually re-enable the *errdisabled* interface after fixing the invalid configuration. It is also possible to set up a time-out interval after which the switch automatically tries to re-enable the interface. However, if the invalid configuration still exists, the switch will of course, *errdisable* the interface again.

# BPDU Guard Configuration

- To enable BPDU guard globally, use the command:

```
SW(config)# spanning-tree portfast bpduguard default
```

- To enable BPDU guard on a port, use the command:

```
SW(config-if)# spanning-tree bpduguard enable
```

- BPDU guard logs messages to the console:

```
2009 May 12 15:13:32 %SPANTREE-2-RX_PORTFAST:Received  
BPDU on PortFast enable port.
```

```
Disabling 2/1
```

```
2009 May 12 15:13:32 %PAGP-5-PORTFROMSTP:Port 2/1 left  
bridge port 2/1
```

# BPDU Guard Configuration Example

```
Switch(config)# spanning-tree portfast bpduguard default
Switch(config)# end
Switch# show spanning-tree summary totals
Root bridge for: none.
PortFast BPDU Guard is enabled
Etherchannel misconfiguration guard is enabled
UplinkFast is disabled
BackboneFast is disabled
Default pathcost method used is short
Name          Blocking Listening Learning Forwarding STP Active
-----
34 VLANs      0          0          0          36          36
```

# BPDU Filtering

- BPDU filtering prevents a Cisco switch from sending BPDUs on PortFast-enabled interfaces, preventing unnecessary BPDUs from being transmitted to host devices.
- BPDU guard has no effect on an interface if BPDU filtering is enabled.
- **When enabled on an interface**, BPDU filtering has these attributes:
  - It ignores all BPDUs received
  - It sends no BPDUs
- **When enabled globally**, BPDU filtering has these attributes:
  - It affects all operational PortFast ports on switches that do not have BPDU filtering configured on the individual ports.
  - If BPDUs are seen, the port loses its PortFast status, BPDU filtering is disabled, and STP sends and receives BPDUs on the port as it would with any other STP port on the switch.
  - Upon startup, the port transmits ten BPDUs. If this port receives any BPDUs during that time, PortFast and PortFast BPDU filtering are disabled

# BPDU Filtering Configuration

- To enable BPDU filtering globally, use the command:  
`SW(config)# spanning-tree portfast bpdupfilter default`
- To enable BPDU guard on a port, use the command:  
`SW(config-if)# spanning-tree bpdupfilter enable`

# Verifying BPDU Filtering Configuration (1)

- PortFast BPDU filtering status:

```
Switch# show spanning-tree summary
Switch is in pvst mode
Root bridge for: none
Extended system ID          is enabled
Portfast Default            is disabled
PortFast BPDU Guard Default is disabled
Portfast BPDU Filter Default is disabled
Loopguard Default          is disabled
EtherChannel misconfig guard is enabled
UplinkFast                  is disabled
BackboneFast                is disabled
Configured Pathcost method used is short
```

Name	Blocking	Listening	Learning	Forwarding	STP Active
VLAN0001	2	0	0	6	8
1 vlan	2	0	0	6	8

# Verifying BPDU Filtering Configuration (2)

- Verifying PortFast BPDU filtering on a specific port:

```
Switch# show spanning-tree interface fastEthernet 4/4 detail
```

```
Port 196 (FastEthernet4/4) of VLAN0010 is forwarding  
Port path cost 1000, Port priority 160, Port Identifier 160.196.  
Designated root has priority 32768, address 00d0.00b8.140a  
Designated bridge has priority 32768, address 00d0.00b8.140a  
Designated port id is 160.196, designated path cost 0  
Timers:message age 0, forward delay 0, hold 0  
Number of transitions to forwarding state:1
```

```
The port is in the portfast mode by portfast trunk configuration
```

```
Link type is point-to-point by default
```

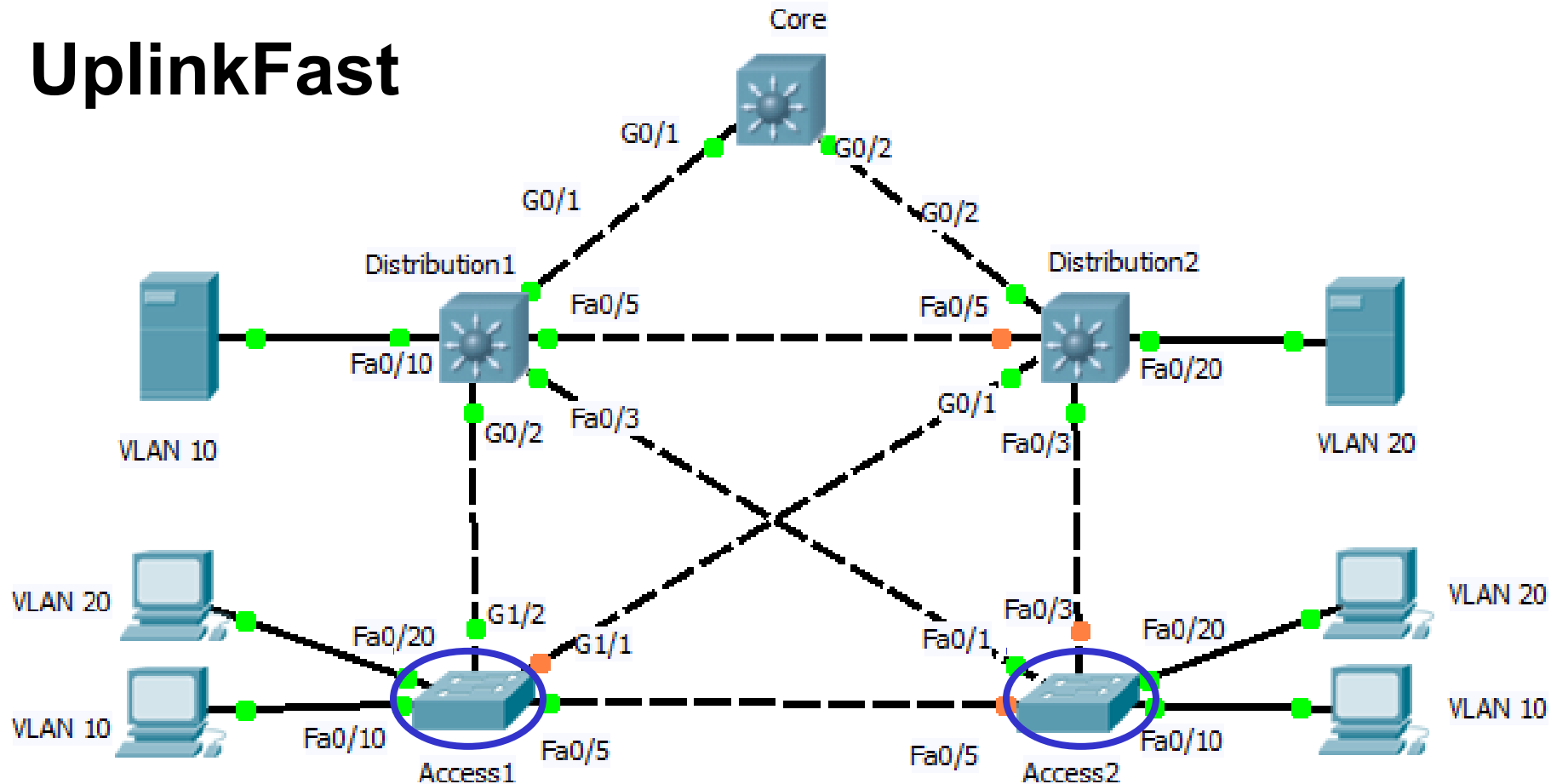
```
Bpdu filter is enabled
```

```
BPDU:sent 0, received 0
```

# Uplink Fast

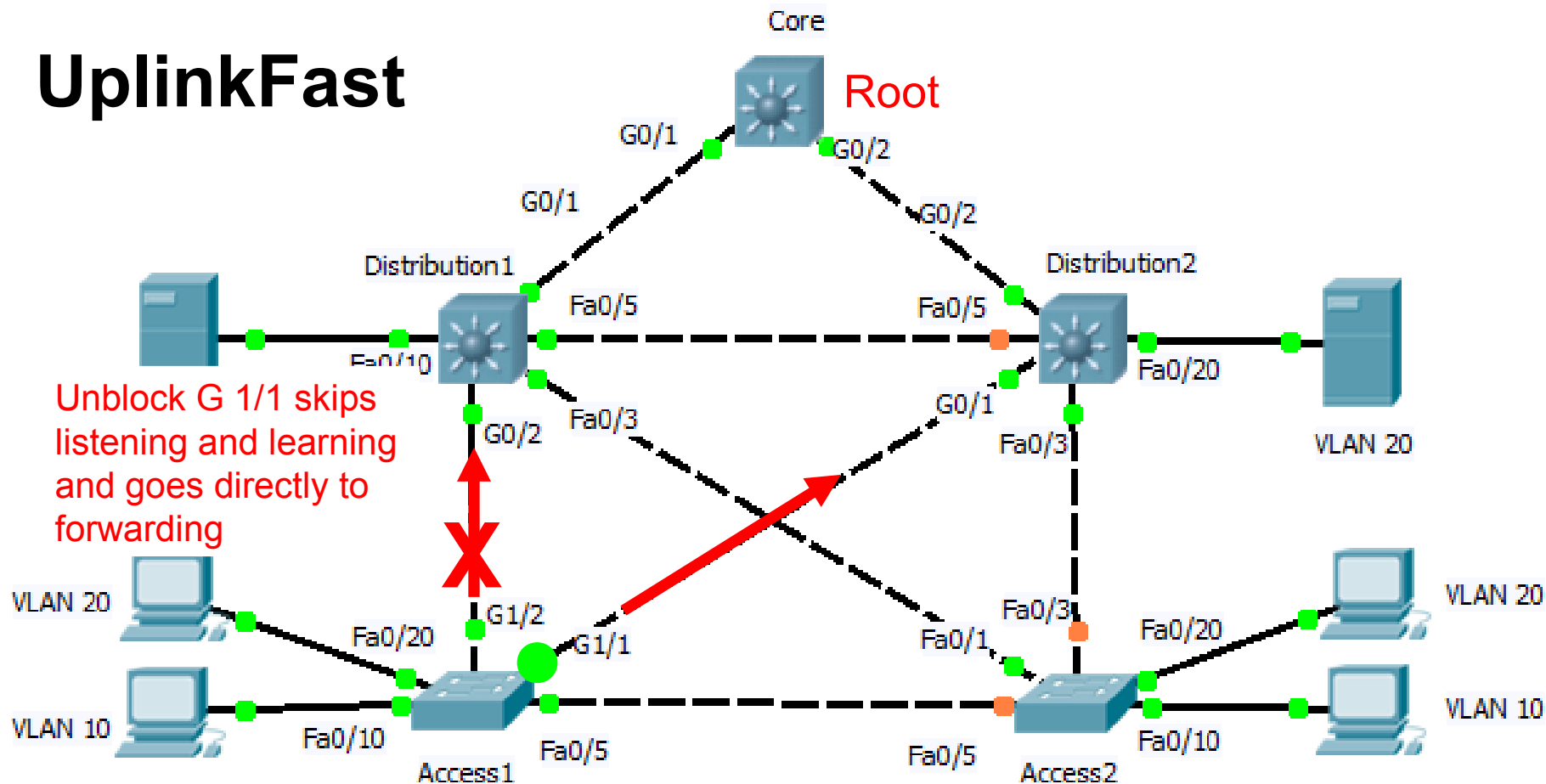
- This optimization reduces the recovery time when a root port (or link) fails on a **non-root** and **non-transit** device (i.e. an Access Layer switch), by:
  - Guarding against this device becoming the Root (through increasing its bridge priority and port costs)
  - Tracking alternate ports where root BPDUs are received
- When a **root port failure** occurs, instead of sourcing a TCN, the device will:
  - Clear its CAM of all addresses reached through the failed port
  - Transition the least cost standby root port immediately from Blocking into the Forwarding state
  - Out of this new root port, send a dummy multicast frame (DA=01.00.0C.CD.CD.CD) for each learned MAC address, with it as the source MAC (to update the CAM of all other switches)

# UplinkFast



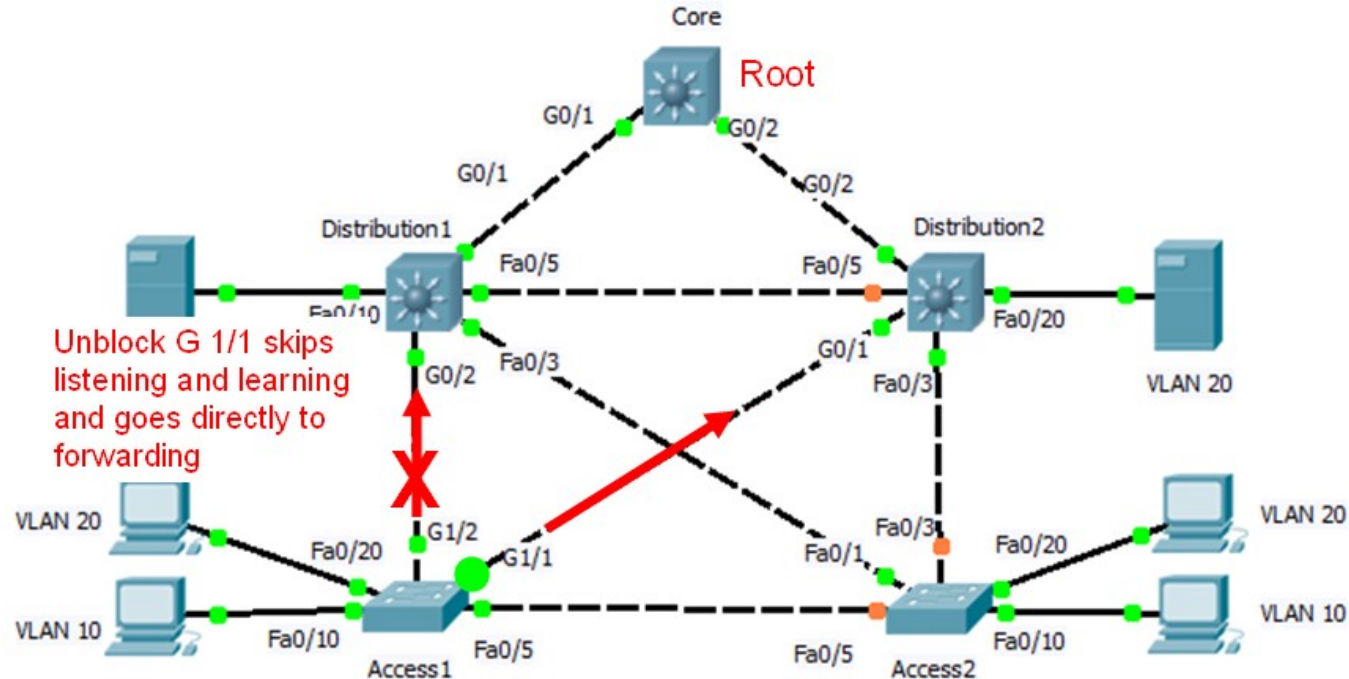
- **Uplinkfast** allows access layer switches having redundant links to multiple distribution switches, the ability to converge quickly when its root path fails.
  - For “Leafs” (end nodes) of the spanning tree.
  - **Not** for use within backbone or distribution switches (use BackboneFast for that - *next*).

# UplinkFast



- UplinkFast must have direct knowledge of the link failure in order to move a blocked port into a forwarding state.
- In addition to its **Root Port** it has **one or more *other potential root ports***.
- If its Root Port fails, the next-lowest cost path is unblocked and used without delay (almost).
  - This switchover occurs within 1 second.

# UplinkFast



*Not available in Packet Tracer*

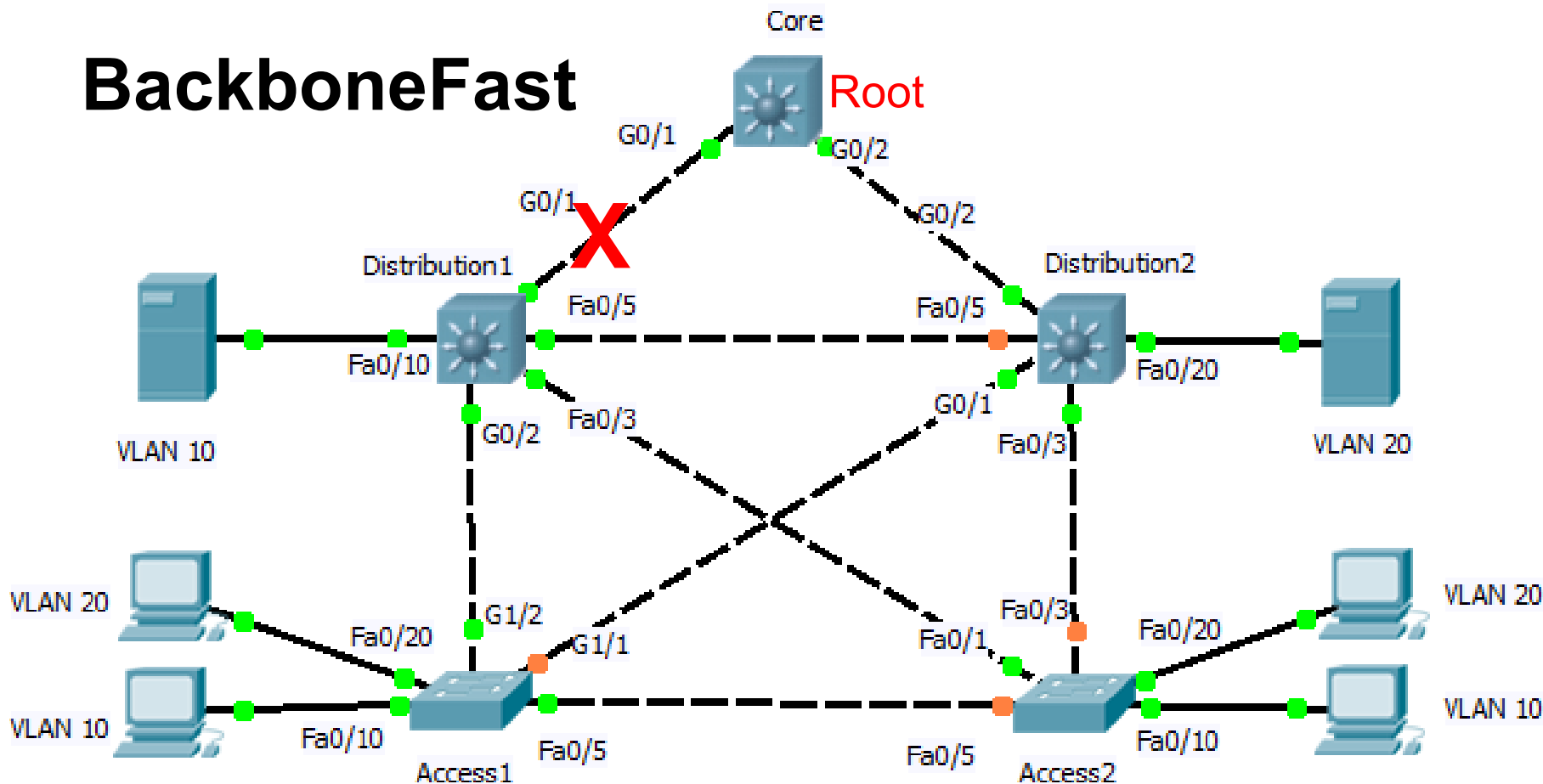
```
Access1 (config) #spanning-tree uplinkfast
```

- Uplinkfast has to be enabled for the entire switch and all VLANs.
  - Not supported on a per-VLAN basis.
- Uplinkfast keeps track of all possible paths to the Root Bridge.
  - So, it's NOT allowed to be the Root Bridge
  - Switch's BID: Raised to 49,152 (0xC000) to make it unlikely to be elected the Root Bridge.

# Backbone Fast

- This reduces the re-convergence time when a device receives an inferior BPDU from a neighbour on a blocked/root port. This means:
  - the root port on that neighbour, the designated bridge, has failed (so it is attempting to become the root), OR
  - the designated bridge has reconfigured with an inferior root path cost (i.e. the root path cost through *this* device is better)
- The device first confirms root reachability by sending a Root Link Query Request (RLQ-REQ) BPDU out the root port, targeting the root BID.
  - other devices with the same root BID forwards it root-wards (any device having a different root BID will return RLQ-NAK)
  - if this reaches the targeted root, it returns a RLQ-ACK BPDU
  - once the ACK is received, the Blocked port moves to Listening and Learning to become the designated port for the segment
- Backbone Fast must be configured on all switches throughout the enterprise.
- **RLQ-REQ**: SNAP frame, Cisco OUI 0x00000c, Prot-ID: 0x0108 (-ACK is 0x0109)

# BackboneFast



Switch(config)# **spanning-tree backbonefast**

*Not available in  
Packet Tracer*

- Backbone fast is a Cisco proprietary feature that, once enabled on all switches can save a switch up to 20 seconds (MaxAge) in situations where it must recover from an indirect link failure.
- Configured in global configuration mode and should be enabled on all switches in the network.
  - Requires the use of special RLQ (Root Link Query) requests and replies.

1. Uh Oh! My RP went down. With no alternate root path, maybe I'm the new root? ...Send out BPDUs to that effect on my DPs.

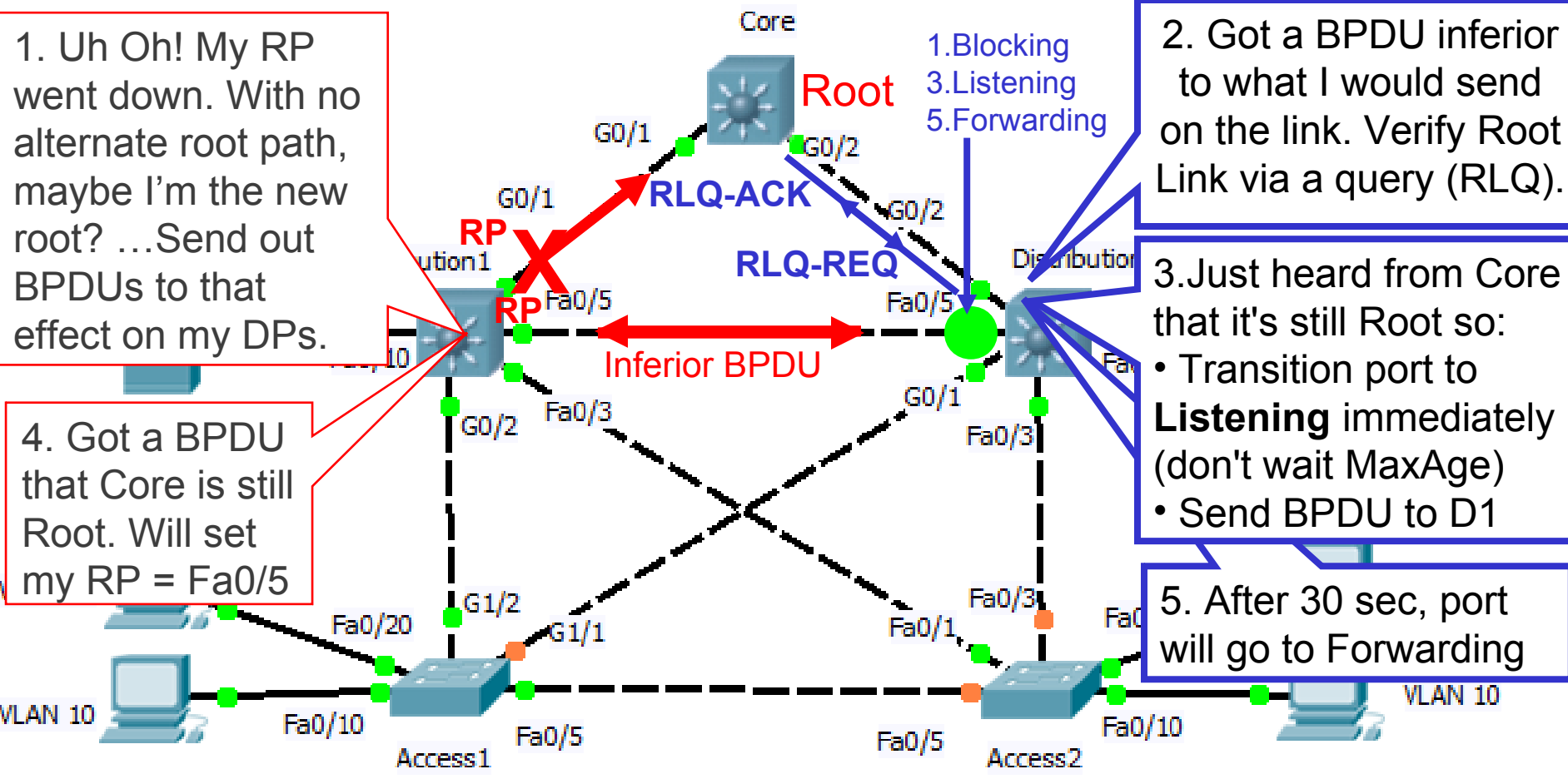
4. Got a BPDU that Core is still Root. Will set my RP = Fa0/5

1. Blocking  
3. Listening  
5. Forwarding

2. Got a BPDU inferior to what I would send on the link. Verify Root Link via a query (RLQ).

3. Just heard from Core that it's still Root so:  
• Transition port to **Listening** immediately (don't wait MaxAge)  
• Send BPDU to D1

5. After 30 sec, port will go to Forwarding



- BackboneFast applies when a root port or blocked port on a switch receives an **inferior BPDU** from the Designated Bridge, signaling an indirect failure.
- Inferior BPDUs are sent from a designated bridge that has lost its connection to the root bridge.
- Normally, a switch must wait for Max Age (20 seconds) to expire before responding to an inferior BPDU.
- With Backbonefast, we *immediately* verify alternate paths to Root.

# Inferior BPDUs

## Normal BPDUs

Bytes	Field
2	Protocol ID
1	Version
1	Message type
1	Flags
8	Root ID = Core
4	Cost of path
8	Bridge ID = Dist1
2	Port ID
2	Message age
2	Max age
2	Hello time
2	Forward delay

310P\_126

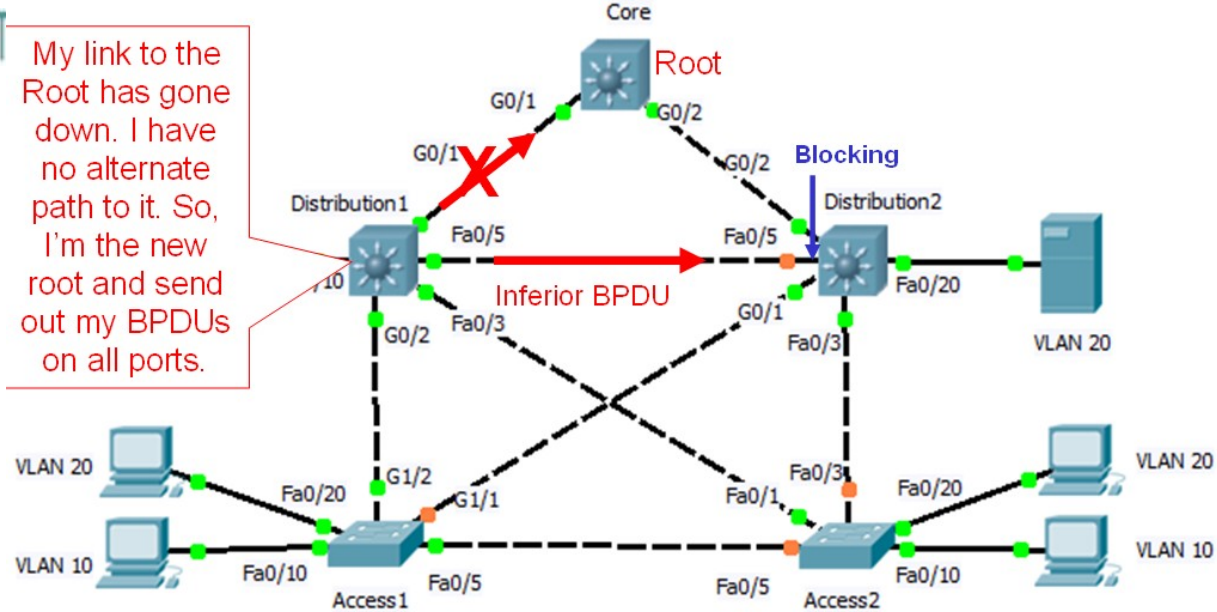
## Inferior BPDUs

2	Protocol ID
1	Version
1	Message type
1	Flags
8	Root ID = Dist1
4	Cost of path
8	Bridge ID = Dist1
2	Port ID
2	Message age
2	Max age
2	Hello time
2	Forward delay

310P\_126

Same  
Switch

My link to the Root has gone down. I have no alternate path to it. So, I'm the new root and send out my BPDUs on all ports.

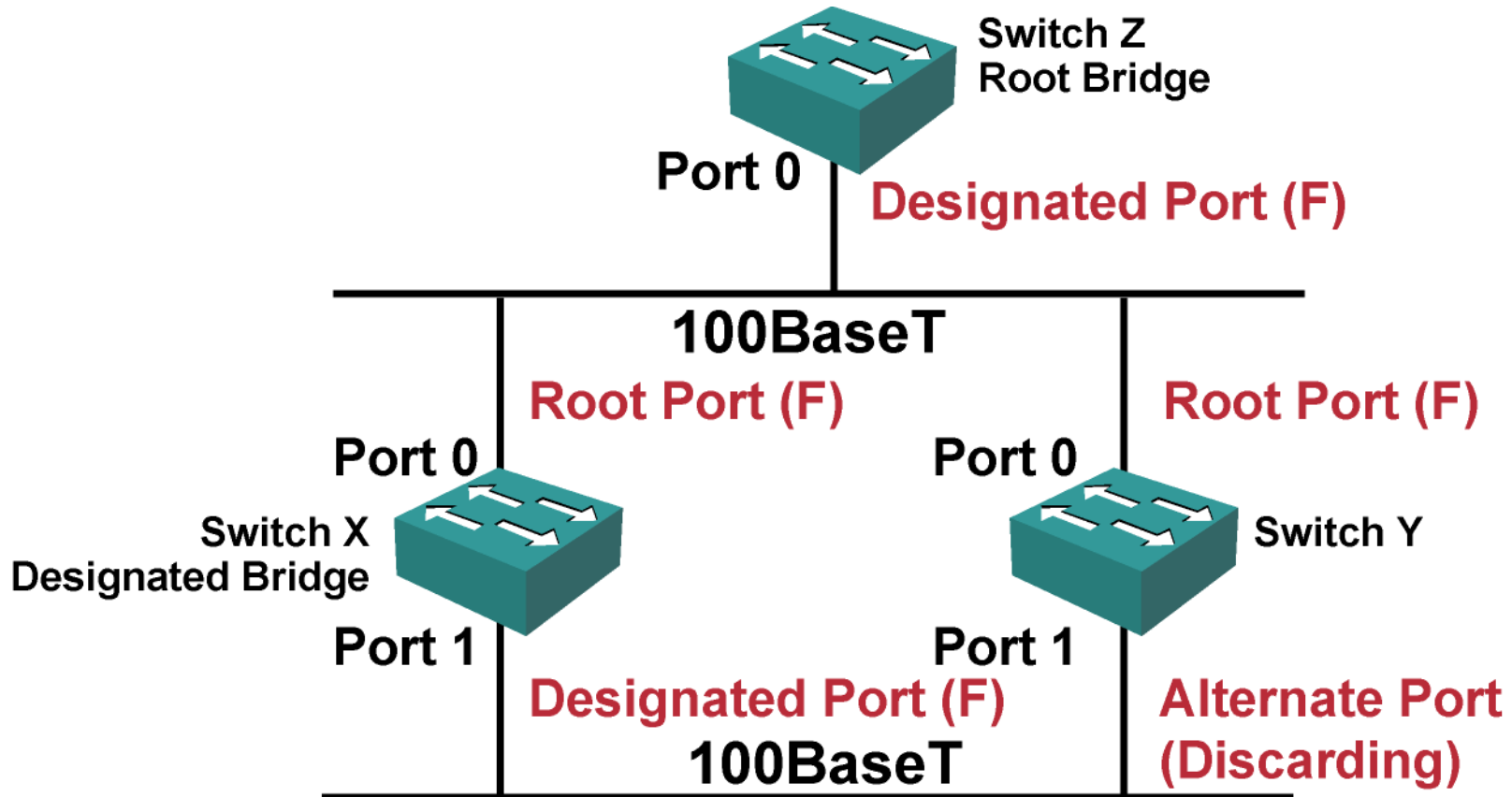


- Dist1 is *designated* for the link to Dist2.
- *Normally*, it sends BPDUs on that link with Core shown as the Root Bridge.
- With the loss of its g0/1 link and no other path to the root bridge, it has no choice but to declare itself as the root.
- As a result, it formulates an **Inferior BPDUs** with itself as both the Root and the Sender (having a root path cost of zero), sending it out all ports on which it would normally propagate Config BPDUs from the Root.

# Rapid STP

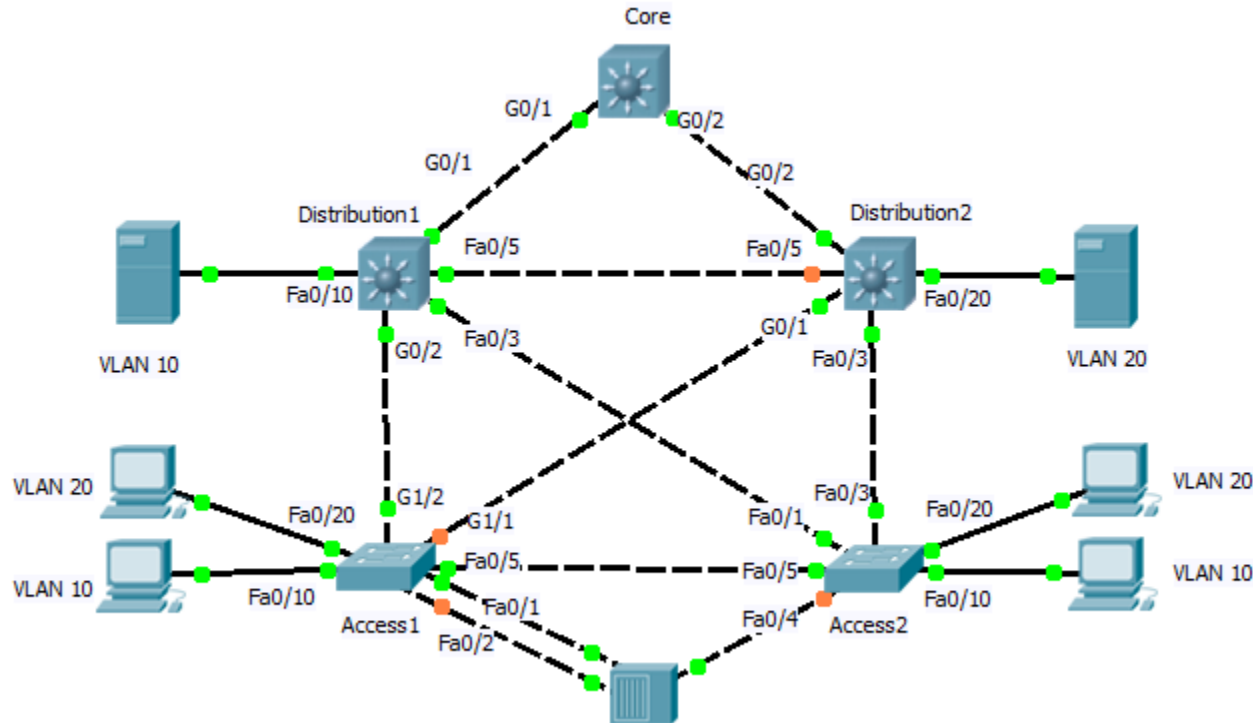


# Rapid Spanning Tree Protocol



310P\_130

# Rapid Spanning Tree Protocol



- The greatest weakness of STP is **convergence**.
- Depending on the type of failure, it takes anywhere from **30 to 50 seconds**, to converge the network.
- **RSTP** helps with the convergence issues that plague legacy STP.

# STP vs RSTP

802.1D



VS



802.1w

- **RSTP** is based on the **IEEE 802.1w** standard.
- IEEE 802.1w **took 802.1D's principle concepts** and ***made convergence faster***.
- **STP** topology change takes **30 seconds** (two intervals of Forward Delay timer).
- **RSTP is proactive** and therefore negates the need for the 802.1D delay timers.
- RSTP (802.1w) supersedes 802.1D, while still remaining **backward compatible**.
- **RSTP BPDU format is the same** as the IEEE 802.1D BPDU format, except:
  - Version field is set to 2 to indicate RSTP
  - BPDU type is also set to 2, reflecting RSTP/MSTP
  - single byte added at the end (Version 1 Length – compatibility with 802.1g)
- ***The RSTP spanning tree algorithm (STA) elects a root bridge in exactly the same way as 802.1D.***

# RSTP

- RSTP can be applied on Cisco switches as:
  - A single instance per VLAN
    - **Rapid PVST+ (RPVST+)**
  - Multiple instances
    - **IEEE 802.1s Multiple Spanning Tree (MST)**

# STP Port Behaviour and States

- 802.1D
  - **Ports**
    - **Root Port**
    - **Designated Port**
    - **Blocking Port**
      - Not Designated Port and Not Root Port
      - Cisco's proprietary UplinkFast has a hidden Alternative Port offering parallel paths, but in Blocking state.
  - **States**
    - **Disabled** (Not 802.1D state)
    - **Blocking**
    - **Listening**
    - **Learning**
    - **Forwarding**
      - Only state that sends/receives data.

## – Ports

- Root Port
- Designated Port
- Blocking Port

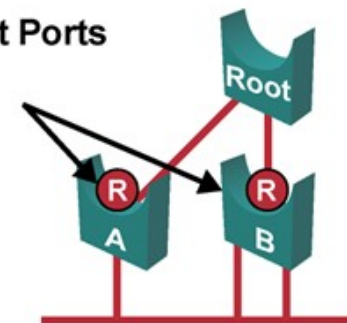
# RSTP

Root Bridge: Same election process as 802.1D (lowest BID)

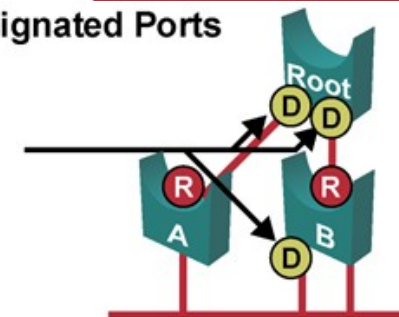
## Ports

- **Root Port (802.1D Root Port)**
  - The one switch port on each switch that has the best root path cost.
- **Designated Port (802.1D Designated Port)**
  - The switch port on a network segment that has the best root path cost to the root.
- **Alternate Port (802.1D Blocking Port)**
  - A port with an alternate path to the root.
  - An alternate port receives superior BPDUs from **another switch** and is a discarding port.
  - Similar to how Cisco UplinkFast works.
- **Backup Port (802.1D Blocking Port)**
  - A port that provides a redundant (but less desirable) connection to a segment where another port on the same switch already connects.
  - A backup port receives superior BPDUs from the **same switch** it is on and is a discarding port.

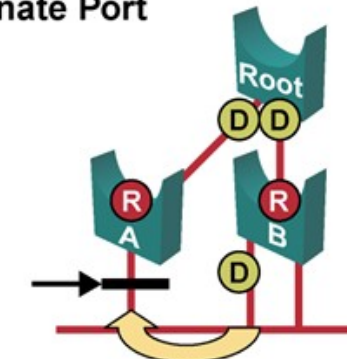
## Root Ports



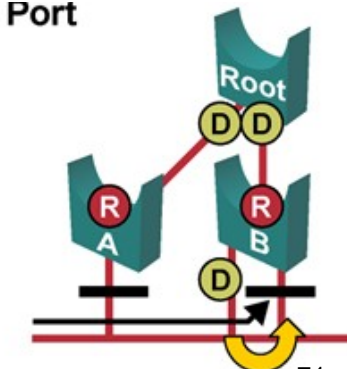
## Designated Ports



## Alternate Port

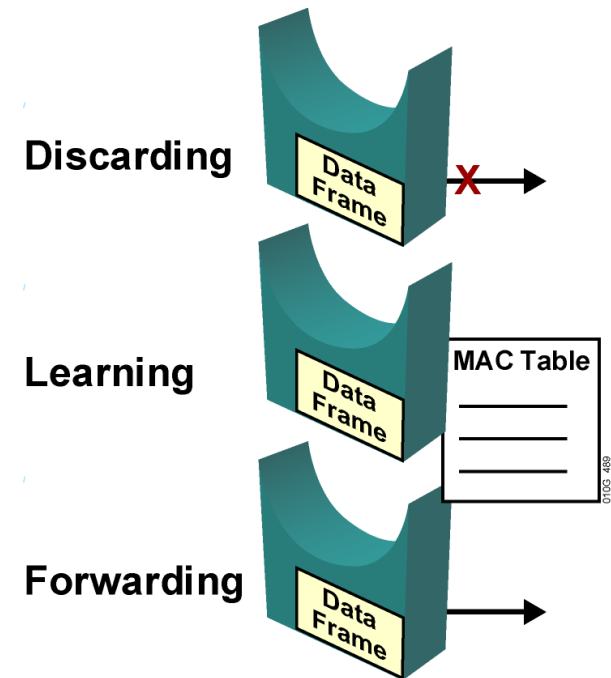


## Backup Port



# RSTP Port States

Operational Port State	STP Port State	RSTP Port State
------------------------	----------------	-----------------



- RSTP defines port states based on how incoming data frames are handled.
- **Discarding**
  - Incoming frames are dropped
  - No MAC Addresses learned
  - Combination of 802.1D (Disabled), Blocking and Listening
- **Learning**
  - Incoming frames are dropped
  - MAC Addresses learned
- **Forwarding**
  - Incoming frames are forwarded.

# RSTP BPDUs

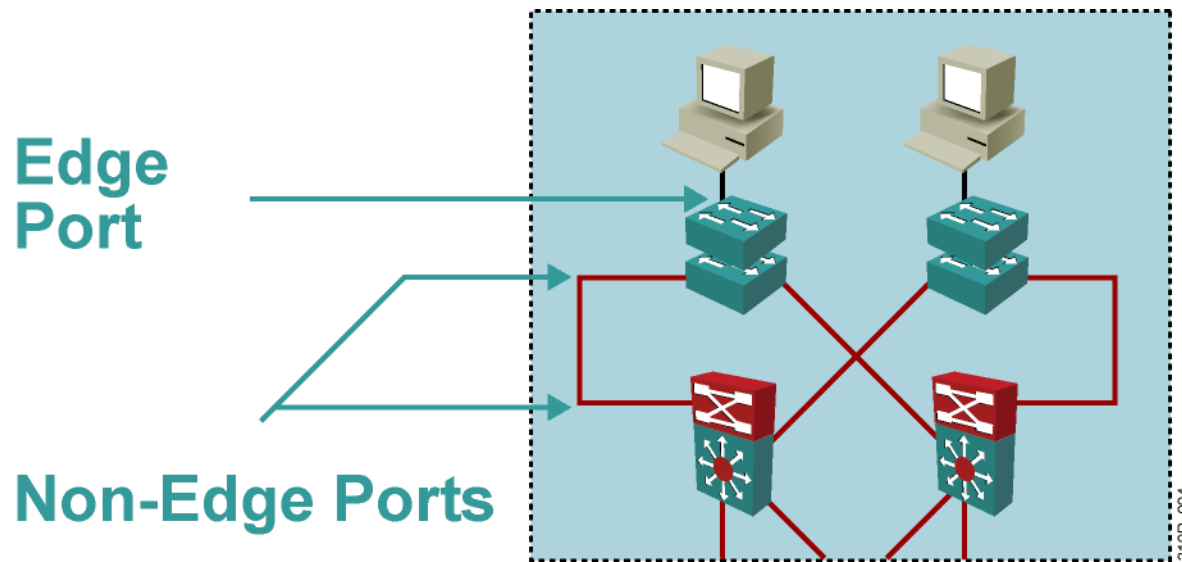
STP Port State	STP BPDUs	RSTP Port State	RSTP BPDUs
Disabled	Not Sent/Received	Discarding	Not Sent/Received
Blocking	Receive only	Discarding	Sent/Received
Listening	Sent/Received	Discarding	Sent/Received
Learning	Sent/Received	Learning	Sent/Received
Forwarding	Sent/Received	Forwarding	Sent/Received

- RSTP uses almost the same 802.1D BPDUs format for backward compatibility.
  - 802.1D and 802.1w switches can coexist.
- BPDUs sent out every non-edge switch port at Hello Time intervals regardless of whether Configuration BPDUs should be sent on the port.
- When three BPDUs in a row (6 seconds) are missed:
  - the neighbor switch is presumed down
  - All MAC address information pointing to that switch (out that port) is immediately aged out (flushed)
  - Switch can detect a neighbor down in **6 seconds instead of MaxAge of 20 seconds.**

# RSTP Convergence

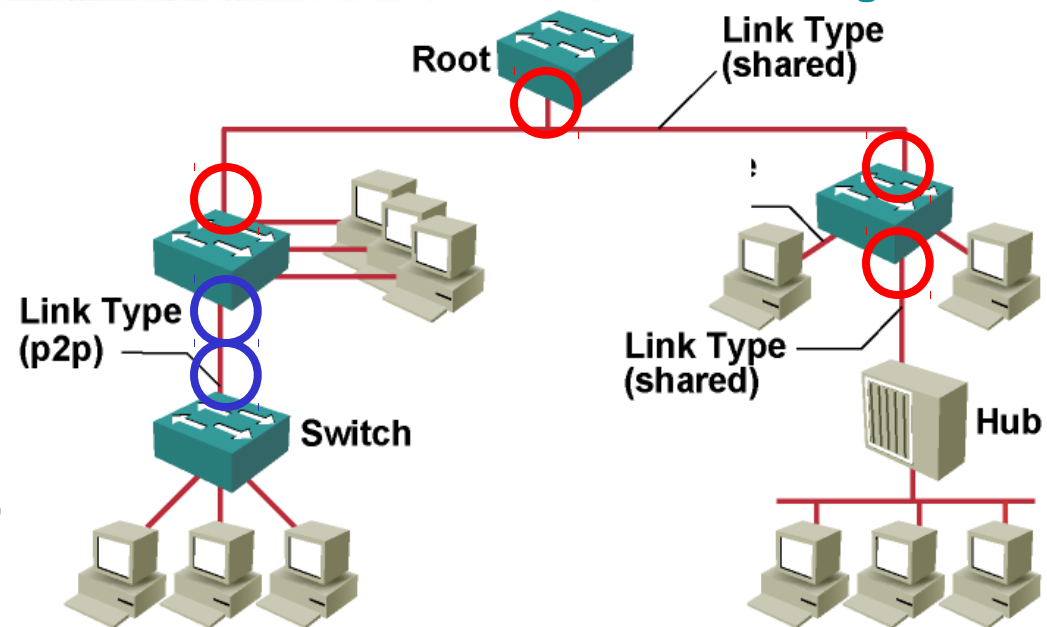
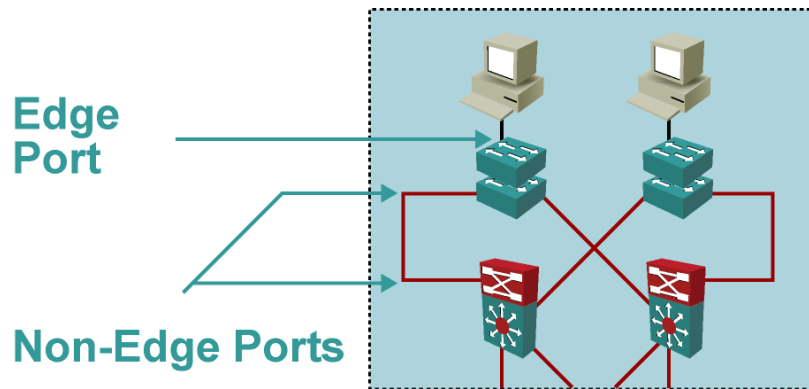
- [http://www.cisco.com/en/US/docs/switches/lan/catalyst2950/software/release/12.1\\_9\\_ea1/configuration/guide/swmstp.html#wp1048403](http://www.cisco.com/en/US/docs/switches/lan/catalyst2950/software/release/12.1_9_ea1/configuration/guide/swmstp.html#wp1048403)
- Convergence is a two step process:
  1. Elect a Root Bridge
  2. Examine all switch ports which by default are in Blocking state and advance to the appropriate state to prevent loops.
- **STP** requires the expiration of several timers before switch ports can be moved to Forwarding state.
- **RSTP** takes a different approach:
  - When a switch joins the topology (powered-up) or reacts to a failure in the existing topology...
  - ***Determines its forwarding decisions based on the type of port.***
    - **Edge Port**
    - **Root Port**
    - **Point-to-Point Port**
    - **Shared Medium Port**

# Edge Ports



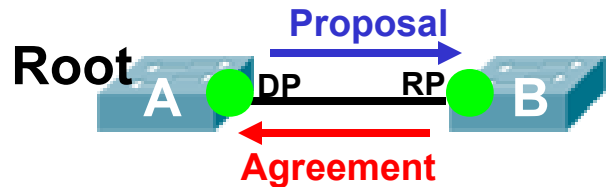
- **Edge port** will never have a switch connected to it so cannot form bridging loops.
- Immediately transitions to **forwarding state**.
- Traditionally identified with **STP PortFast** feature.
- For familiarity, the command is the same: **spanning-tree portfast**
- Never flags topology changes when the port transitions to a disabled or enabled status.
- If an edge port receives a BPDU, it loses its Edge Port status & becomes a normal rapid spanning-tree port.

# Non-Edge Ports



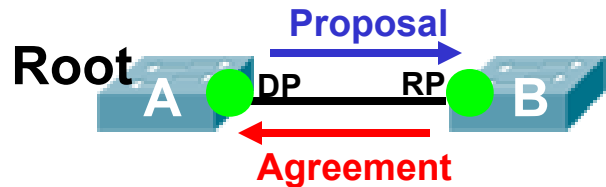
- **Root Port**
  - The one switch port on each switch that has the best root path cost to the root.
- **Point-to-Point Port (Link Type)**
  - Port operating in **full-duplex mode**.
  - **Connects to another switch** and becomes a **Designated Port**.
  - Uses a **quick handshake** with neighboring switch rather than timers to decide port state.
- **Shared Medium Port (Link Type)**
  - Port operating in **half-duplex mode**.
  - ***It is assumed that the port is connected to shared media where multiple switches might exist.***

# Point-to-Point: The Quick Handshake



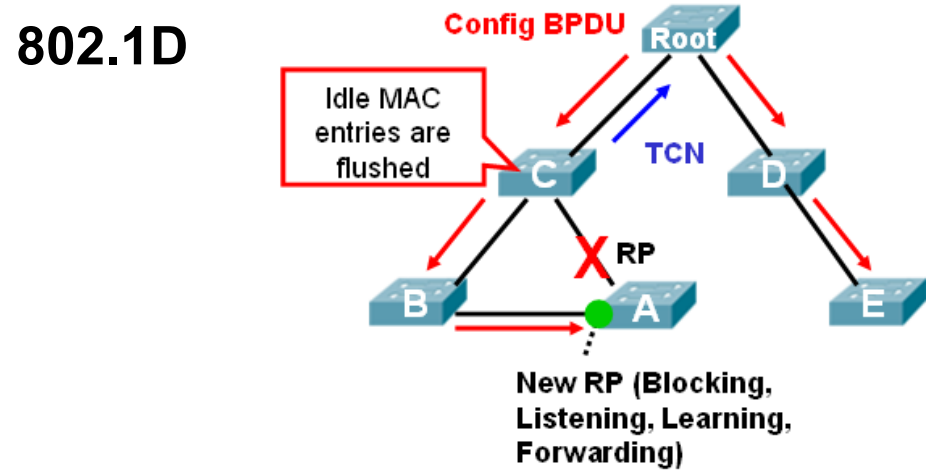
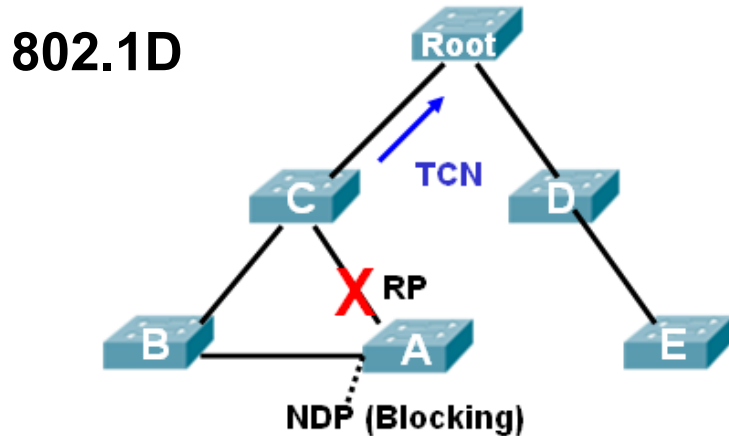
- **Switch A** is connected to Switch B through a point-to-point link,
  - All ports are in the **Discarding** (Blocking) state.
- **Switch A** has a lower BID than Switch B.
- **Switch A** sends a **proposal message** (Configuration BPDU) to Switch B, proposing itself as the **Root Bridge** and the designated switch on the segment.
- **Switch B:**
  - Selects its **new root port** the *port from which the proposal message was received* and *immediately goes into Forwarding State*
  - Forces all **nonedge ports** to the **Discarding (Blocking) state**,
  - **Sends an agreement message.**
- **Switch A:** Immediately transitions its **designated port to the forwarding state.**
- No loops in the network are formed because Switch B blocked all of its nonedge ports and because there is a point-to-point link between Switches A and B.

# Proposal - Agreement



- **Switch C is connected to Switch B:** a similar set of handshaking messages are exchanged.
- **Switch C** selects the port connected to Switch B as its **root port**, and both ends immediately transition to the **forwarding state**.
- Handshaking process continues throughout topology.

# RSTP Topology Change Notifications



- 802.1D

- Switch detects a state change (up or down), it sends the Root Bridge a TCN BPDU.
- The Root Bridge sends out a Configuration BPDU (TC bit set) to all switches to tell them about the change. In all, at least 30 seconds to recover from a failure.

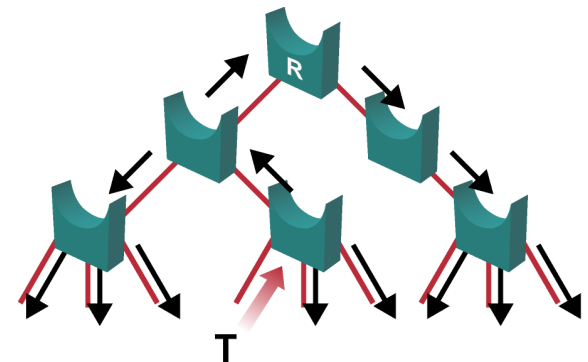
- RSTP

- *Detects a topology change only when a non-edge port transitions to the Forwarding State.*
- RSTP uses its **convergence mechanisms** (Edge Ports, Point-to-Point ports, handshaking, etc.) to prevent bridging loops.
- Therefore, topology changes are **detected only so MAC address tables can be updated and corrected.**
- This means that a **loss of connectivity is not considered as a topology change** any more, contrary to 802.1D (that is, a port that moves to blocking no longer generates a TC event).

# RSTP Topology Change Notifications

## RSTP

- When a topology change occurs:
  - Switch **flushes the MAC addresses** associated with **all non-edge ports**.
  - Switch **sends BPDUs** with TC bit set to **all neighbors** so they can update their MAC Address tables too.
- When a bridge receives a BPDU with the TC bit set from a neighbor:
  - It **clears the MAC addresses** learned on all its ports (**except the port on which it received the topology change**).
  - It **sends BPDUs** with TC set **on all its designated ports and root port** (RSTP no longer uses the specific TCN BPDU, unless a legacy bridge needs to be notified).
- This way, the **TC floods very quickly** across the whole network - now a **one step** process.
- The **initiator of the topology change floods this information** throughout the network, as opposed to 802.1D where it had to come from the Root.
- **Much faster than the 802.1D** equivalent, where we had to wait for the root bridge to be notified, and then flush for MaxAge plus FwdDelay duration.
- In just a **few seconds**, or a small multiple of hello-times, **most of the entries in the CAM tables of the entire network (VLAN) flush**.
- This approach results in **potentially more temporary flooding**, *but* on the other hand it **quickly clears potential stale information** and allows **rapid convergence**.

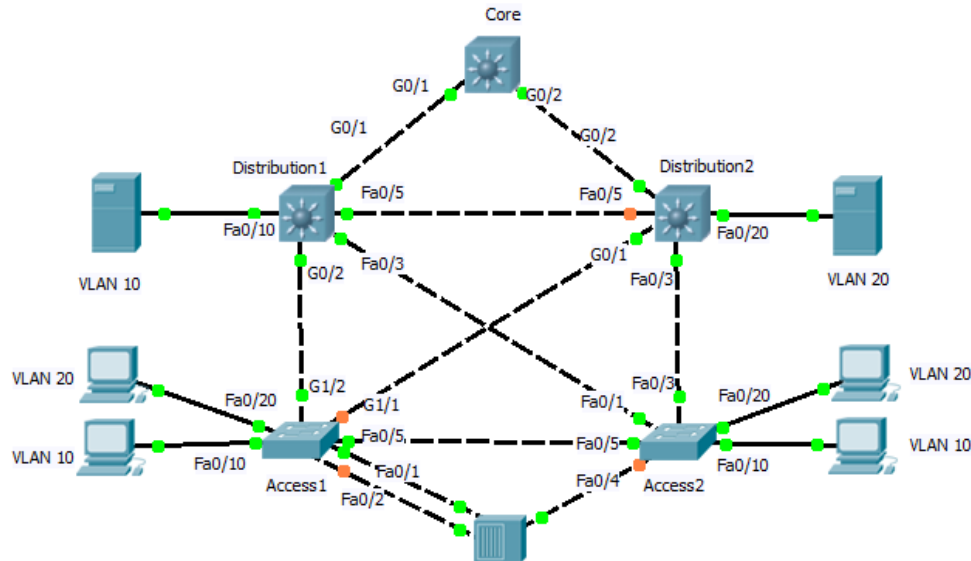


The originator of the TC directly floods this information through the network.

3109\_197

# Rapid PVST Implementation Commands

*Cisco implements RSTP with Rapid PVST+*



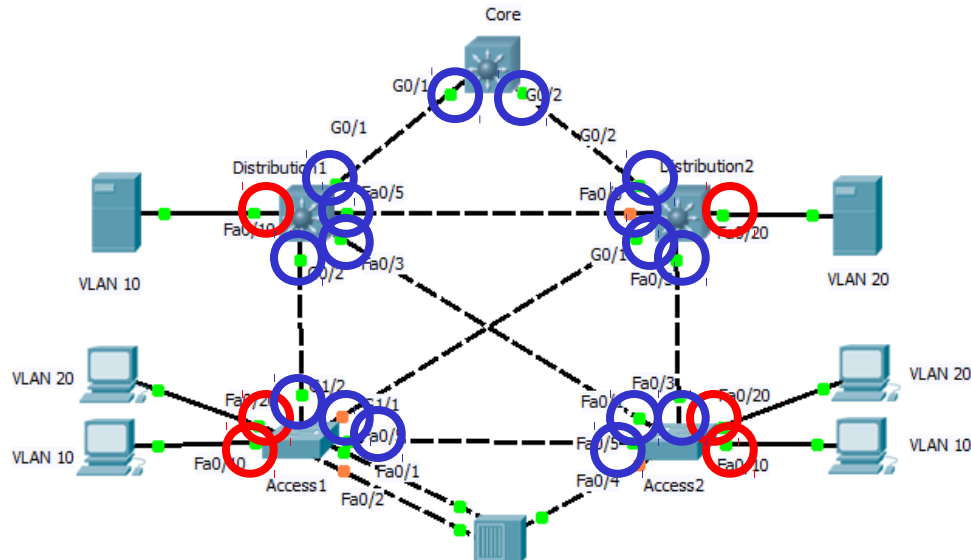
```
Switch(config)# spanning-tree mode rapid-pvst
```

- To revert back to the default PVST+ using traditional 802.1D:

```
Switch(config)# spanning-tree mode pvst
```

# Rapid PVST Implementation Commands

*Cisco implements RSTP with Rapid PVST+*



- To configure an RSTP edge port:

```
Switch(config-if) # spanning-tree portfast
```

- RSTP automatically decides if a port is point-to-point link operating in full duplex or half-duplex.
- If you need to set it manually, other switch is in Half-Duplex but still point-to-point (by the way, both ends must then be Half-Duplex):

```
Switch(config-if) # spanning-tree link-type point-to-point
```

# Rapid PVST Implementation Commands

```
Access1# show spanning-tree
```

```
VLAN0001
```

```
Spanning tree enabled protocol rstp
```

```
Root ID      Priority      24577
             Address      0001.C945.A573
             Cost          4
             Port          26(GigabitEthernet1/2)
             Hello Time    2 sec      Max Age 20 sec      Forward Delay 15 sec
```

```
Bridge ID    Priority      32769 (priority 32768 sys-id-ext 1)
             Address      0003.E461.46EC
             Hello Time    2 sec      Max Age 20 sec      Forward Delay 15 sec
             Aging Time    20
```

# Lifting the Hood on Cisco Protocols

- Cisco switches default to PVST+ (Per VLAN Spanning Tree, Plus), a proprietary version of STP.
- The RSTP version is RPVST+ (sometimes also called Per VLAN Rapid Spanning Tree – PVRST+)
- Whereas 802.1D BPDUs are LLC-encapsulated via DSAP/SSAP=0x42, PVST+ payloads are SNAP-encapsulated (DSAP/SSAP=0xAA) using the Cisco OUI of 0x00000c, followed by a Protocol ID = 0x010b.
- Other Cisco protocol IDs you may encounter in this format, include:
  - CDP: PID=0x2000
  - VTP: PID=0x2003
  - DTP: PID=0x2004
  - PAgP: PID=0x104
  - RLQ-REQ: PID=0x108
  - RLQ-ACK: PID=0x109
  - UDLD: PID=0x111