Day 21: Spanning Tree Protocol pt.2:-

→ Spanning Tree Port States:-

- There are two other port states, **Listening**, and **Learning**.
- **Blocking** and **Forwarding** are the two *stable* states.

- Root/Designated remain *stable* in a **Forwarding** state.
- Non-Designated ports remain *stable* in a **Blocking** state.

NOTE: They only remain stable as long as there are no changes in the network topology. If a new device is added, an interface is shutdown, or a hardware failure occurs somewhere, They may have to change states. But, as long as the network is stable, each **Spanning Tree** interface will be stable in one of these states.

- **Listening** and **Learning** are *transitional states*, which are passed through when an interface is activated, or when a **Blocking** port must transition to a **Forwarding** state due to a change in the Network Topology.

- Actually, there is one more state you might hear of, this is the **Disabled** state.
- This simply refers to an interface that is *Administratively disabled*, meaning Shutdown.
- It doesn't play any role in **Spanning Tree**, the interface is Shutdown!

STP Port State	Stable/Transitional
Blocking	Stable
Listening	Transitional
Learning	Transitional
Forwarding	Stable
(Disabled)	

→ Blocking State:-

- Non-Designated ports are in a **Blocking** state.

- Interfaces in a **Blocking** state are effectively *disabled* to prevent loops.
  This is what makes **Spanning Tree** work, disabling redundant interfaces to avoid loops.

- Interfaces in a **Blocking** state do not send/receive regular network traffic.
  Any regular traffic that arrives on an interface in a **Blocking** state will simply be dropped.

- Interfaces in a **Blocking** state **DO** receive **STP BPDU**s.
  They need to RECEIVE and PROCESS **BPDU**s to be aware of the **Spanning Tree Topology**, and be ready to transition toward a **Forwarding** state if they need to.

- Interfaces in a **Blocking** state do NOT forward **STP BPDU**s.

- Interfaces in a **Blocking** state do NOT learn MAC Addresses.
  If regular traffic arrives on the interface it is dropped without adding the MAC Address to the MAC Address Table.

→ Listening State:-

- After the **Blocking** state, interfaces with the Designated or Root role enter the **Listening** state.
- Only **Designated** or **Root** ports enter the **Listening** state (Non-Designated are always **Blocking**).
    That's because Listening is a *transitional state* that eventually leads to the Forwarding state.
    So, there is no need for a Non-Designated port to enter this state.

- The **Listening** state is **15** seconds long by default. This is determined by a timer called the '**Forward Delay**' Timer.
    We'll soon see that this time isn't used only for the Listening state.

- An interface in the **Listening** state ONLY forwards/receives **STP BPDU**s.
- An interface in the **Listening** state does NOT send/receive regular traffic.
    If a regular uni-cast frame is received on a port in the Listening state, it will be discarded.
- An interface in the **Listening** state also does NOT learn MAC addresses from regular traffic that arrives on the interface.
    We said the same thing about the **Blocking** state, but let's explain!:
    When a frame arrives on a Switch interface, the Switch uses the source MAC address field to ***learn*** that MAC address.
    And it updates the MAC Address Table with the MAC address, interface and VLAN information.
    However, if an interface is in the **Spanning Tree Listening** state, it will not do this. The traffic is simply dropped,
    and the MAC address learning process does not occur!

→ Learning State:-

- After the **Listening** state, a Designated or Root port will enter the **Learning** state.

- The **Learning** state is **15** seconds long by default. This is determined by the '**Forward Delay**' Timer (the same timer used for both).
    By default it takes a total of **30** seconds to move through both states and enter a **Forwarding** state.

- An interface in the **Learning** state ONLY sends/receives **STP BPDU**s.
- An interface in the **Learning** state does NOT send/receive regular traffic.
    Same as in the Listening state.

- An interface in the **Learning** state ***learns*** MAC addresses from regular traffic that arrives on the interface.
    Different than Listening state.
    An interface in the Learning state is preparing to forward traffic by building up some of its MAC Address Table beforehand.

→ Forwarding State:-

- Root and Designated ports are in a **Forwarding** state when they are stable.

- A port in the **Forwarding** state operates as normal.

- A port in the **Forwarding** state sends/receives **STP BPDU**s.

- A port in the **Forwarding** state sends/receives normal traffic.

- A port in the **Forwarding** state ***learns*** MAC addresses from the frames that arrive on it and adds them to the MAC Address Table.

- It's a Switchport operating as normal!

- For review, here's a summary of each **Spanning Tree** port state.

STP Port State	Send/Receive BPDUs	Frame forwarding (regular traffic)	MAC address learning	Stable/ Transitional
Blocking	NO/YES	NO	NO	Stable
Listening	YES/YES	NO	NO	Transitional
Learning	YES/YES	NO	YES	Transitional
Forwarding	YES/YES	YES	YES	Stable
Disabled	NO/NO	NO	NO	Stable

→ Spanning Tree Timers:-

- We've already mentioned the **Hello** and **Forward Delay** timers, but we haven't mentioned **Max Age**.

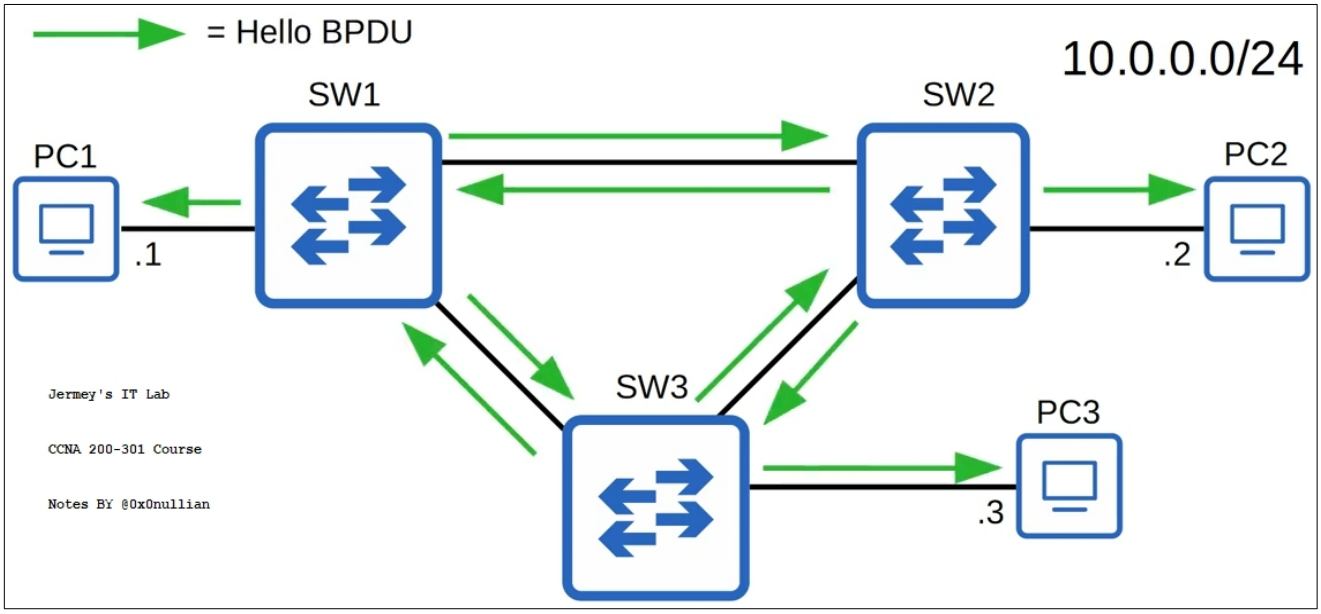
→ Hello Timer:

- It determines how often the Root Bridge sends **Hello BPDU**s. It will send them every **2** seconds by default.

- Other Switches in the Network do not originate their own **BPDU**s, but they will forward **BPDU**s they receive.

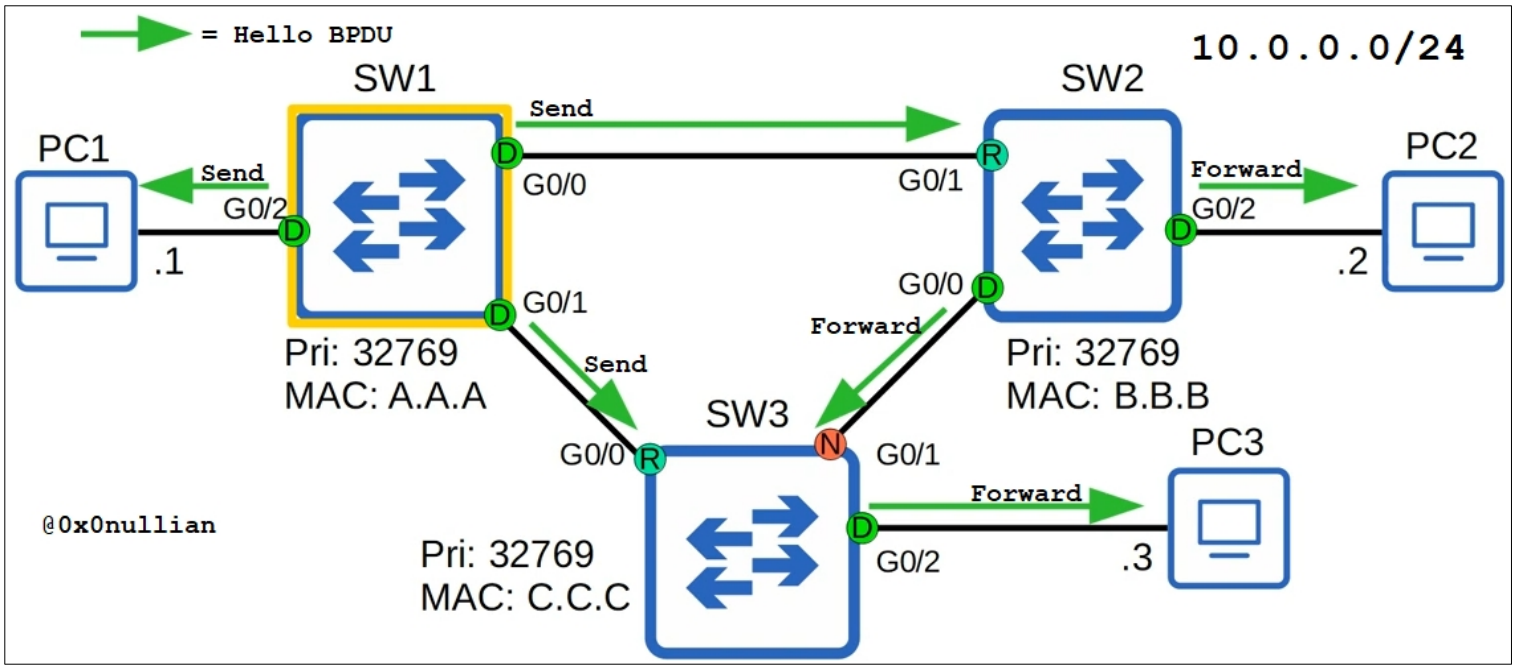
- The Switches will only forward **BPDU**s on their DESIGNATED PORTS. Let's see how that works:-

- Assuming these Switches all come online at the same time, each assumes they are the Root Bridge, and each will send **BPDU**s out of all interface.



- However, once the Network has converged and all Switches and ports are stabilized in their roles, ONLY the Root Bridge sends **BPDUs**.

- Then the other Switches will forward these **BPDU**s on their designated ports, updating information like the *Bridge Root Cost*, sending *Bridge ID*, sending *Port ID*, etc.
- Then **2** seconds later, The Root Bridge will send **BPDU**s again, and the other Switches will again forward these **BPDU**s on their designated ports.



NOTE: Switches do not forward the **BPDU**s out of their **Root** ports and **non-Designated** ports, only their **Designated** ports!

→ Forward Delay Timer:-

- This is the length of Listening and Learning transitional states that a port goes through when it moves to Forwarding.
- This is the length of each of states, not the total length of both combined.
- So, with the default Forward Delay Timer of **15** seconds, it takes a total of **30** seconds for the Switchport to move through both states and forward traffic.

→ Max Age Timer:-

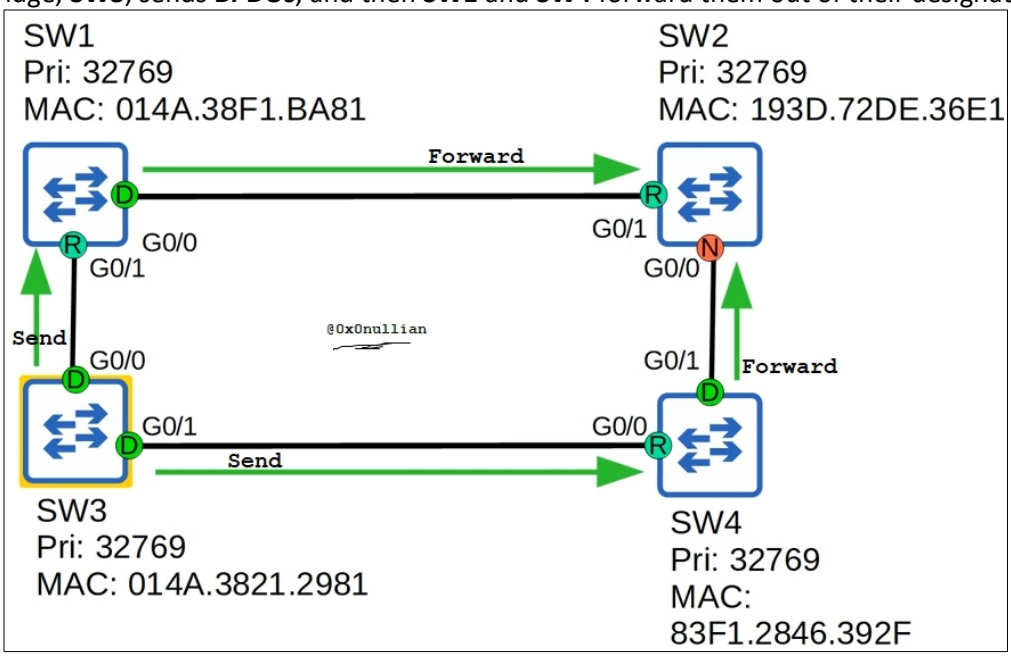
- Indicates how long an interface will wait to change the **Spanning Tree Topology** after ceasing to receive **Hello BPDU**s.
- Let's take a look:

- Remember that each collision domain has one designated port, and **BPDU**s are forwarded out of designated ports.
- So, all Root Ports and non-Designated ports expect to receive **BPDU**s.

** **STP** Timers Summary ** →

STP Timer	Purpose	Duration
Hello	How often the root bridge sends hello BPDUs	2sec
Forward delay	How long the switch will stay in the Listening and Learning states (each state is 15 seconds = total 30 seconds)	15sec
Max Age	How long an interface will wait after ceasing to receive Hello BDPUs to change the STP topology.	20sec (10* hello)

- The Root Bridge, **SW3**, sends **BPDUs**, and then **SW1** and **SW4** forward them out of their designated ports.



→ To demonstrate the Max Age Timer, Let's focus on **SW2**'s **G0/1** interface:-
- It just received a **BPDU**, so the Max Age Timer is rest to **20**. It counts down to **19**... **18**...
- And then the Root Bridge sends **BPDU**s, because of the **2** seconds Hello Timer!
- These **BPDU**s are forwarded by the other Switches, and **SW2** resets its Max Age Timer to **20**... **19**... **18**...
- But, What if a failure occurs on the connection between **SW1** and **SW2**??

- The Root Bridge will send **BPDU**s, and other Switches will forward the **BPDU**s, but **SW1**'s **G0/0** interface is down,
  So **SW2** no longer receives a **BPDU** on its **G0/1** interface.
- The Max Age Timer continues counting down.. **17**... **16**... **15**... and if the failure doesn't recover and **SW2** doesn't receive any more **BPDU**s on its **G0/1** interface, **SW2**'s Max Age Timer will count all the way down to **0**.

→ What happens then??
1. First of all, If another **BPDU**is received before the Max Age Timer counts down to **0**,
The time will reset to **20** seconds and no changes will occur.

2. If another **BPDU** is not received, the Max Age Timer counts down to **0**, and the Switch will re-evaluate its **STP** choices, including Root Bridge, and Local Root, Designated and non-Designated ports.

3. After these decisions, If a non-Designated port is selected to become a Designated or Root port, it will transition from the Blocking state to the Listening state (**15** seconds), and then Learning state (**15** seconds), and then finally the Forwarding state. So, it can take a total of **<u>50 seconds</u>** for a blocking interface to transition to forwarding.

→ Why does it take so long??
- These timers and transitional states are to make sure that loops aren't accidentally created by an interface moving to Forwarding state too soon.
- We've seen how DANGEROUS Layer2 loops can be.
  That's why **Spanning Tree Protocol** is very careful about moving an interface to a forwarding state.

- However, ...

> A forwarding interface can move directly to a blocking state (there is no worry about creating a loop by blocking an interface).
>
> A blocking interface cannot move directly to forwarding state. It must go through the listening and learning states.

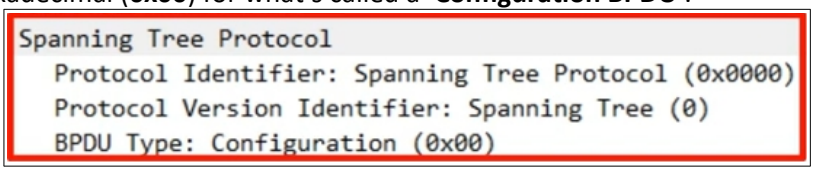
→ STP BPDU, Bridge Protocol Data Unit:-

1- First off, in the Ethernet Header section, notice the destination:

- Cisco's **PVST+** uses the destination MAC address of **0100.0CCC.CCCD** for its **BPDU**s
- It is recommended remembering this, it's a little bit of trivia you might need to know for the CCNA exam test.
- **PVST** = Only supports ISL trunk encapsulation
- **PVST+** = Supports **802.1Q** encapsulation too.
- Since we mentioned the MAC address, regular **Spanning Tree** (meaning not Cisco's **PVST** or **PVST+**), uses a destination MAC address of **0180.C200.0000**.

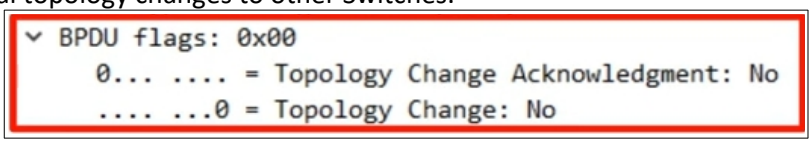
2- **Spanning Tree BPDU** itself:

- The first three fields are the Protocol Identifier, which is always hexadecimal (**0x0000**) for **Spanning Tree**.
- The Protocol Version Identifier is set to **0** for **Classic Spanning Tree**.
  We'll see a different value when we look at **Rapid Spanning Tree Protocol** in Day22.
- Finally, the **BPDU** type is hexadecimal (**0x00**) for what's called a '**Configuration BPDU**'.



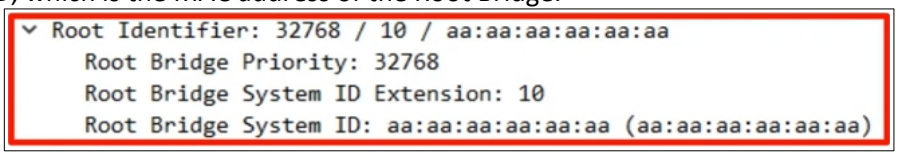
3- **BPDU Flags:**

- These Flags are used to signal topology changes to other Switches.



4- **Root Identifier:-**

- Root Identifier is which give the Bridge Priority, Extended-System ID, which is the VLAN ID, **10** in this case, and the Bridge System ID, which is the MAC address of the Root Bridge.

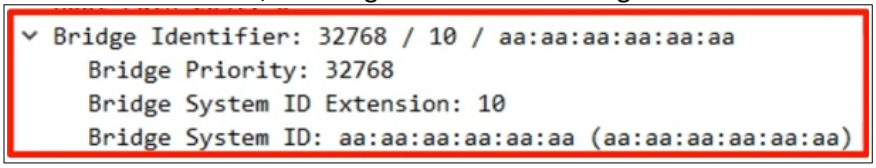


5- **Root Path Cost:-**

- It is **0** in this case, so we know that this is the Root Bridge.

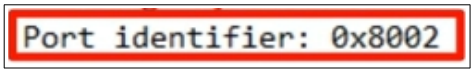
6- **Bridge Identifier:-**

- We can also know this is the Root Bridge by looking at this field.
- Same information as in the **Root Identifier** field, meaning this is the Root Bridge.



7- **Port Identifier:-**

- The interface which sent the **BPDU**. It is hexadecimal (**0x8002**).
- **80** in hexadecimal is equivalent to **128**, which if you remember from Day20, is the default Port Priority.
- **02** is the number of the port itself.



8- The Timers:-

- **Message Age**, it starts at **0** at the Root Bridge and is increased by **1** each time it is forwarded by another Switch.
  It is subtracted from the Max Age when a Switch receives the **BPDU**.
  So for example if the **BPDU** is passed through **5** Switches, when it reaches the **6th** Bridge it will immediately reduce its Max Age Timer to **15**.
  Meaning each time it receives a **BPDU**, its Max Age will reset to **15** instead of **20**, Even though the Max Age timer is **20**.
- After that we have the Three Timers we talked about Max Age, Hello Time, and Forward Delay.
- By the Way, the **STP Timer** on the Root Bridge determine the **STP Timers** used for the rest of the Switches in the Network.
  Even if they are configured differently.

```
Message Age: 0
Max Age: 20
Hello Time: 2
Forward Delay: 15
```

→ STP Toolkit:-

→ These are features that can be enabled to improve the functionality of **Spanning Tree Protocol** in some way.

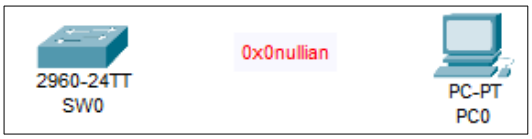
1- Portfast:-

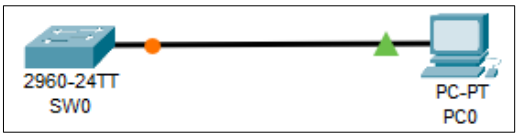
- It solves one problem of **Spanning Tree**.
- ***Portfast*** can be enabled on interfaces which are connected to end hosts, like the **G0/2** interface on each of these Switches.
- These are designated ports in a Forwarding state. However, when they are first turned on or first connected to the PCs,

They must go through the Listening and Learning states first before they can start forwarding traffic. This takes **30** seconds in total.

→ Let's see on the Packet Tracer:-

- Open Packet Tracer, and then add a Switch and a PC.
- Just make sure *Show Link Lights is enabled*. It must be enabled by default, but just to be assured, Go to **Options**, **Preferences**, and then make the *Show Link Light* ***enabled***.
- Place a Switch and the PC like this, and get your timer ready and then connect them together:





- At first we should see that the link light on the Switch is Orange.
- This is the same if we connect a real physical PC to a Physical Switch, the link light will be Orange.
- It's because the port is not in a Forwarding yet, it is going through the Listening and Learning states.
- However, **30** seconds later we should finally see the link light turn Green. The port is finally forwarding.

→ Only Interfaces connected to another Switch can form a Layer2 loop. There is no risk of forming a loop with an end host.

- So, wouldn't it be nice if theses ports connected to end hosts could start forwarding right away,
  without having to wait **30** seconds to go from Listening to Learning to Forwarding??
- Well, that what ***portfast*** does!

> Portfast allows a port to move immediately to the **Forwarding** state, bypassing **Listening** and **Learning**.
> If used, it must be enabled only on ports connected to end hosts.
> If enabled on a port connected to another switch it could cause a Layer 2 loop.

- The purpose of the Listening and Learning states is to avoid causing a loop,
  so bypassing them is risky when connected to another Switch!!!

→ We haven't looked at any other **Spanning Tree** configurations yet because **Spanning Tree** operates by default even without.
We will look at general **STP** configuration, but first let's look at ***portfast***!

→ **Portfast Configurations:-**

- ***Portfast*** is enabled at the interface level with the command (**`spanning-tree portfast`**).
- Then we get a Warning about enabling ***portfast***, as it should be enabled only on ports connected to an end host.
- There is also a message saying that, even though ***portfast*** was configured, it will only take effect
  if the interface is in a non-trunking mode, so if it is an access port!
  That's because trunk ports are typically connected to other Switches.
- We can still configure *portfast* on a trunk port, it just won't take effect.

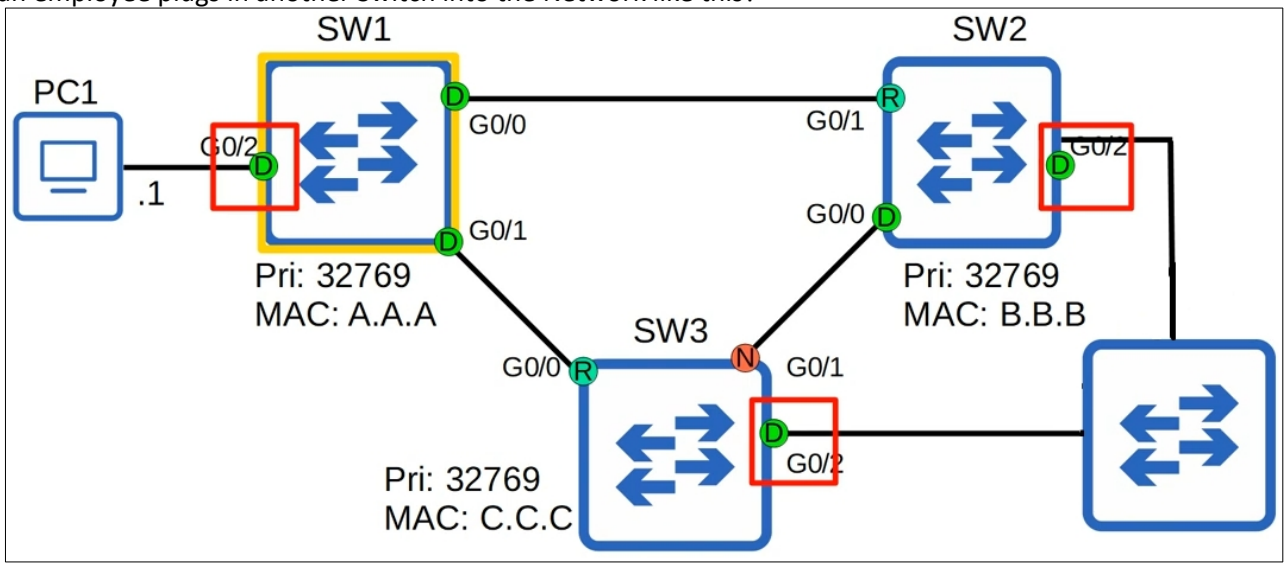
```
SW1(config)#interface g0/2
SW1(config-if)#spanning-tree portfast
%Warning: portfast should only be enabled on ports connected to a single
 host. Connecting hubs, concentrators, switches, bridges, etc... to this
 interface  when portfast is enabled, can cause temporary bridging loops.
 Use with CAUTION

%Portfast has been configured on GigabitEthernet0/2 but will only
 have effect when the interface is in a non-trunking mode.
SW1(config-if)#
```

→ We can also enable ***portfast*** with this command in Global Configuration Mode:
(**`spanning-tree portfast default`**)

- This command enables ***portfast*** on all access ports, not trunk ports.

→ Portfast is a great feature for getting a switchport connected to an end host running quickly without having to wait **30** seconds.

- However, it can still be a risk!
- What if an employee plugs in another Switch into the Network like this?



- This employee doesn't necessarily have malicious intent, they could just e unaware of exactly what they are doing.
- Because Portfast is putting these interfaces into a Forwarding state, a Layer2 loop is formed!
- Portfast can also cause loops if the network cabling is changed without proper caution,
  perhaps a host is moved to a different switchport and a Switch is connected to its old port.

→ The point is that there is a risk to using portfast!

→ However, there is an additional **Spanning Tree** optional feature that we can enable to protect against such loops.
It is called **BPDU Guard**!

2- BPDU Guard:-

- If an interface with **BPDU Guard** enabled receives a **BPDU** from another Switch, the interface will be shutdown to prevent a loop from forming.
- It is very simple to configure: from interface configuration mode, we use the command: (**`spanning-tree bpduguard enable`**).

```
SW1(config)#interface g0/2
SW1(config-if)#spanning-tree bpduguard enable
SW1(config-if)#
```

- Similar to portfast, there is also an option to enable it by default. With this command from the Global Configuration Mode: (**`spanning-tree portfast bpduguard default`**)
- This command enables **BPDU Guard** on all **Portfast** enabled interfaces!
- Notice that the commands are a little different, to enable it directly on the interface its **spanning-tree bpduguard enable**, with no mention of **portfast**.
- However to enable it Globally, we have to include **portfast** in the command, **spanning-tree portfast bpduguard default**.

→ We took this screenshot in Packet Tracer, so the CLI color scheme is a little different than the previous one, but we connected a Switch to a **BPDU-Guard** enabled interface, and now we can see what happens when a **BPDU** arrives on a **BPDU Guard** enabled port:

```
%SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port FastEthernet0/1 with BPDU Guard
enabled. Disabling port.

%PM-4-ERR_DISABLE: bpduguard error detected on 0/1, putting 0/1 in err-disable state


%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to administratively down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to down
```

- The port is disabled, it is effectively shut down.

→ What if you want to enable the port again??
- To enable a port that was disabled by **BPDU Guard**, simply (**`shutdown`**), and then (**`no shutdown`**) the interface.

```
Switch(config-if)#shutdown

%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to administratively down
Switch(config-if)#no shutdown

Switch(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up
```

- We can see that the interface comes up!

- However, if we didn't actually solve the problem and it's still connected to a Switch, We can see here that the interface will immediately be disabled again when the next **BPDU** arrives
- Make sure you actually solve the problem before trying to enable the interface again!

```
%SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port FastEthernet0/1 with BPDU Guard enabled. Disabling port.

%PM-4-ERR_DISABLE: bpduguard error detected on 0/1, putting 0/1 in err-disable state


%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to administratively down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to down
```

→ In terms of **Spanning Tree** optional features, the **200-301** exam topics list only mentions **portfast**.
→ We also have shown **BPDU Guard**, because it is connected to **Portfast**, so it might be included!

→ There are many other optional features that can be enabled, but there's no need to know about them for the CCNA.
→ But let's just quickly introduce two others we should at least know the name and basic purpose of.
Just in case they are mentioned in the Exam.
→ They are **Root Guard** and **Loop Guard**.

4- Root Guard:-

Root Guard	If you enable **root guard** on an interface, even if it receives a superior BPDU (lower bridge ID) on that interface, the switch will not accept the new switch as the root bridge. The interface will be disabled.

- This helps maintain the **Spanning Tree** Topology if someone plugs another Switch into the Network,
either with bad intent, or perhaps without knowing the impact of their action.

5- Loop Guard:-

Loop Guard	If you enable **loop guard** on an interface, even if the interface stops receiving BPDUs, it will not start forwarding. The interface will be disabled.

- This prevents loops that can happen if an interface fails only in one direction, causing what is called a '**uni-directional link'**
that can't receive data, but is still able to forward it, or the opposite.

You probably don't have to know these STP optional features (or others such as UplinkFast, Backbone Fast, etc) for the CCNA. But make sure you know **Portfast** and **BPDU Guard**. If you want to read more about the others just in case, do a Google search.

→ Spanning Tree Configurations:-

→ Let's look at some basic **Spanning Tree Configurations**.

→ Spanning Tree Mode Configuration:-

- We can configure the **Spanning Mode** the Switch uses with the command (**`spanning-tree mode ?`**), there are **3** options:

```
SW1(config)#spanning-tree mode ?
  mst         Multiple spanning tree mode
  pvst        Per-Vlan spanning tree mode
  rapid-pvst  Per-Vlan rapid spanning tree mode

SW1(config)#spanning-tree mode pvst
```

- **MST**, **Multiple Spanning Tree**, not a topic we need to know for the CCNA.
- **PVST**, is the **Classic Spanning Tree**, but with Cisco's Per-VLAN addition.
- **Rapid-PVST**, Improved version of the **PVST**.

- Modern Cisco Switches run **Rapid-PVST** by default, and usually there is no reason to change it
- However, if you want to tey out the **Classic Spanning Tree** for your lab, like we will do for these demonstrations,
We can enable it with this command (**`spanning-tree mode pvst`**).

→ Configure the Primary Root Bridge:-

We can also manually configure the Root Bridge by manipulating the Bridge Priority of a Switch.

- With these MAC addresses and the default priority values, **SW1** is the Root Bridge!
- However, we could configure **SW3** to be the Root Bridge!!
- We could also configure something called a '**Secondary Root Bridge**', which will be next in line to become the Root Bridge, if the current Root Bridge fails!!

→ Let's see how to configure it:-

- This is how to configure the Root Bridge, called the **Primary Root Bridge**.
- With the command (**`spanning-tree vlan n root primary`**), while **`n`** is the VLAN number.
- Now if we (**`do show spanning-tree`**), we will see that this bridge has become the Root Bridge.

```
SW3(config)#spanning-tree vlan 1 root primary
SW3(config)#do show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    24577
             Address     cccc.cccc.cccc
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    24577  (priority 24576 sys-id-ext 1)
             Address     cccc.cccc.cccc
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time  15  sec
```

The **`spanning-tree vlan`** *`vlan-number`* **`root primary`** command sets the STP priority to 24576. If another switch already has a priority lower than 24576, it sets this switch's priority to 4096 less than the other switch's priority.

- So, this command makes this Switch have the lowest priority, making it the Root Bridge.

- If we then check the **`running-config`**, we can see that the command that is actually applied in this case is just:
  (**`spanning-tree vlan 1 priority 24576`**)

```
!
spanning-tree mode pvst
spanning-tree extend system-id
spanning-tree vlan 1 priority 24576
!
```

- So, this command tells the Switch to apply the **Spanning Tree** priority command, either with the Priority **14576**, or **4096** less than the current lowest priority.

→ The command to set the **Secondary Root Bridge**, The Bridge with the second-lowest priority, is the same command like before, just change the word '**`primary`**' to '**`secondary`**'.

- This time the command, (**`spanning-tree vlan n root secondary`**), sets the **STP** Priority for this VLAN to **28672**.

The **`spanning-tree vlan`** *`vlan-number`* **`root secondary`** command sets the STP priority to 28672.

```
SW2(config)#spanning-tree vlan 1 root secondary
SW2(config)#do show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    24577
              Address     cccc.cccc.cccc
              Cost        4
              Port        1 (GigabitEthernet0/0)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    28673  (priority 28672 sys-id-ext 1)
              Address     bbbb.bbbb.bbbb
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec
```

- However, like the Root Primary command,
  the actual command that is applied is the (**`spanning-tree vlan 1 priority 28672`**).

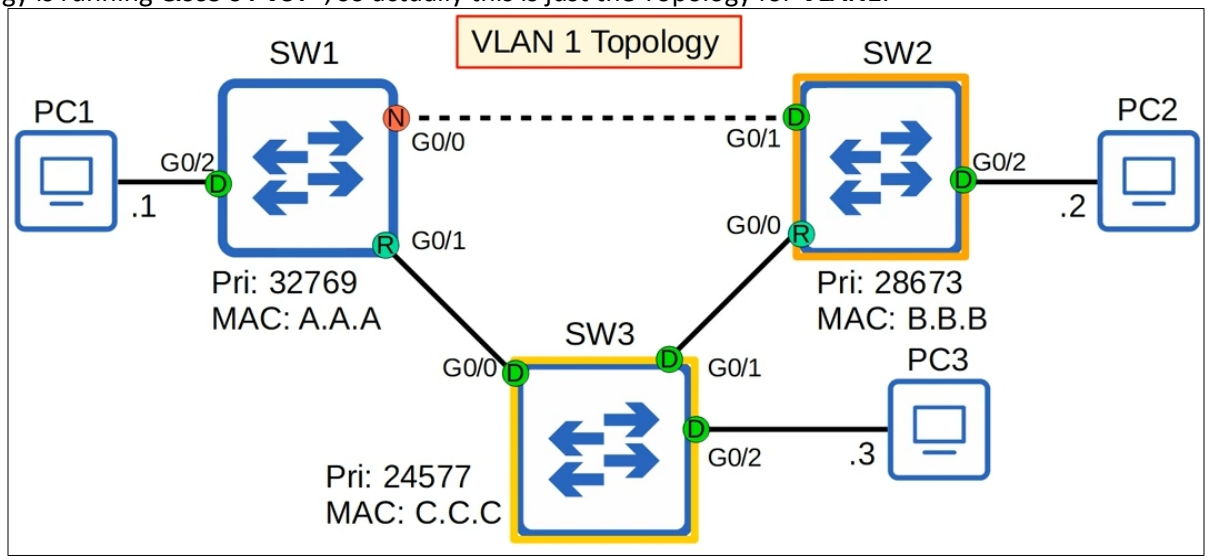
- So, for both of these two commands, we could actually use the (**`spanning-tree priority`**) command as we see here to configure the Root Bridge, the **Spanning-Tree** Root command is just a simple way to do it without remembering the different increments of **4096**.

```
!
spanning-tree mode pvst
spanning-tree extend system-id
spanning-tree vlan 1 priority 28672
!
```

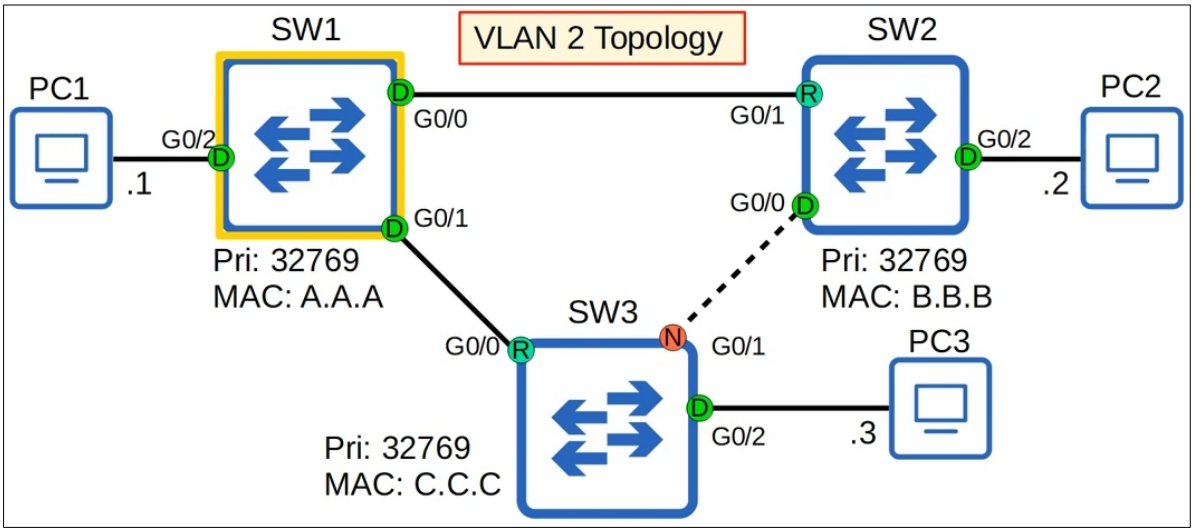
Remember: The Bridge Priority must be set in increments of **4096**, so the Root command is easier to use.

→ So, this is our Topology Now:-

- The interface between **SW1** and **SW2** is disabled because **SW1** is blocking its **G0/0** interface.
- This Topology is running **Cisco's PVST+**, so actually this is just the Topology for **VLAN1**.



- Perhaps there is another VLAN, **VLAN2**, in this Topology, what will the Topology look like for it?
- It will look like this in the next page, the default Topology, because the Root bridge settings we configured only apply to **VLAN1**.



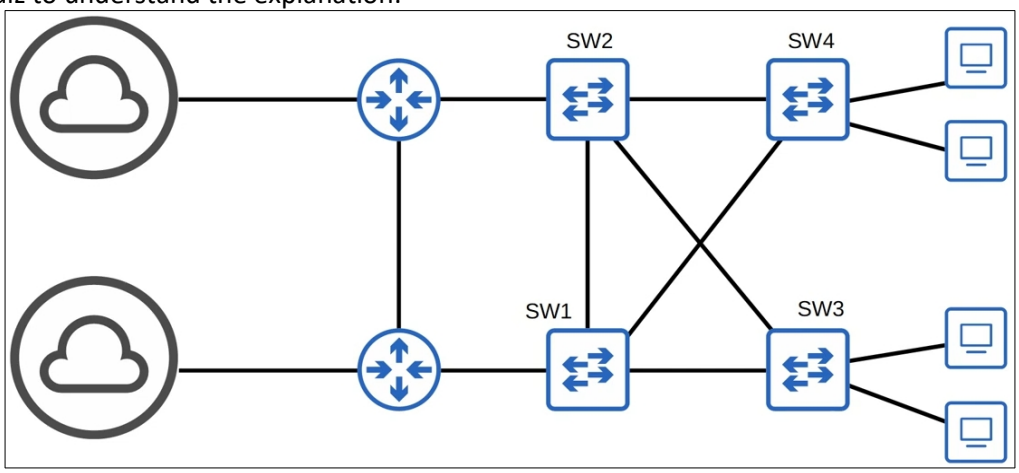
- In **VLAN2**, the connection between **SW1** and **SW2** WON'T be disabled, instead the connection between **Sw2** and **SW3** will be!!
- This allows for what's called **Spanning Tree Load-Balancing**.

→ STP Load-Balancing:-

- If we have multiple VLANs in our network, blocking the same interface in each VLAN is a waste of interface bandwidth.
- That connection will be doing nothing, just waiting for another connection to fail so it can start forwarding.
- However, If we configure a different Root Bridge for different VLANs, different VLANs will disable different interfaces.
- Different connections are used in different VLANs. The Load is balanced across different interfaces.

  This is called **Load-Balancing**.

→ Let's do a quick quiz to understand the explanation:-



Two VLANs are active in this network, 10 and 20. By default, SW3 is the root bridge for both VLANs. Configure SW1 as the primary root for VLAN10 and the secondary root for VLAN20. Configure SW2 as the primary root for VLAN20 and the secondary root for VLAN10.
Which two commands should you issue on SW1, and which two commands should you issue on SW2?

- So, as we guessed: the commands used in **SW1** is

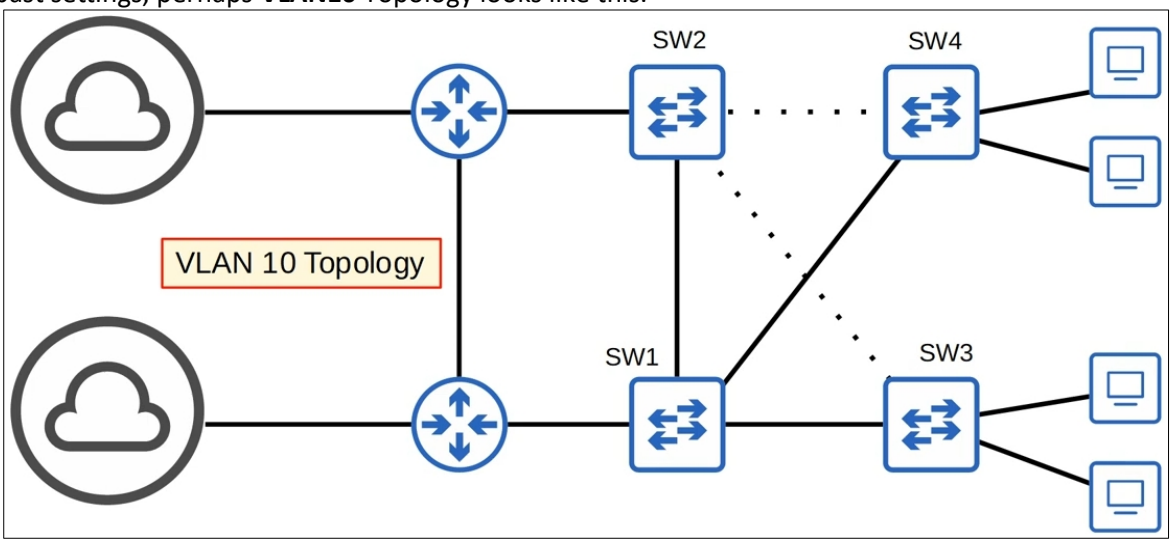
  (**`spanning-tree vlan 10 root primary`**), and (**`spanning-tree vlan 20 root secondary`**)
- And in **SW2** is:

  (**`spanning-tree vlan 20 root primary`**), and (**`spanning-tree vlan 10 root secondary`**)

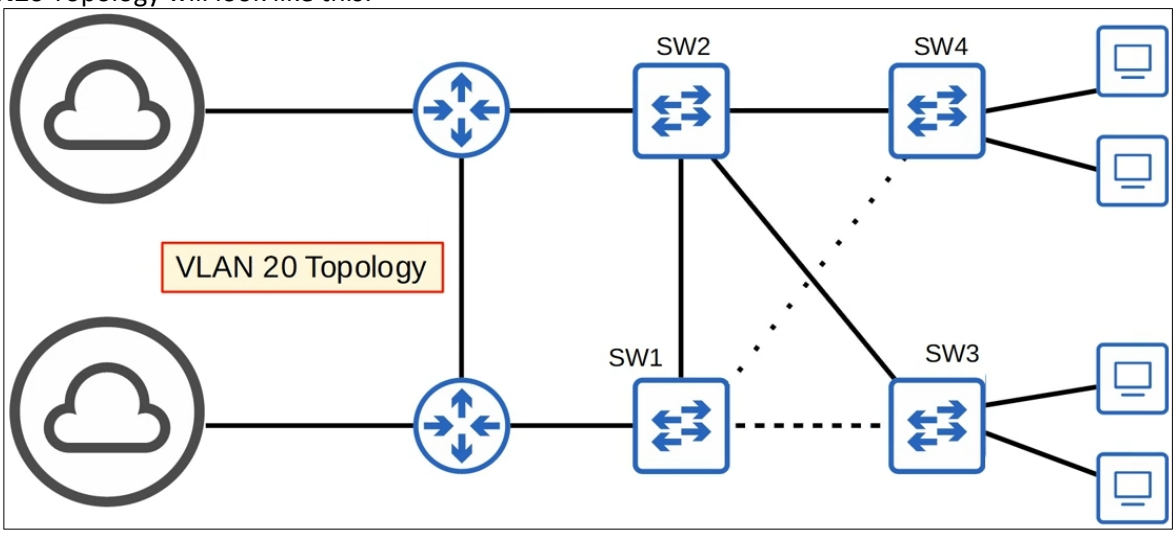
```
SW1(config)# spanning-tree vlan 10 root primary
SW1(config)# spanning-tree vlan 20 root secondary
```

```
SW2(config)# spanning-tree vlan 20 root primary
SW2(config)# spanning-tree vlan 10 root secondary
```

- So, with the past settings, perhaps **VLAN10** Topology looks like this.



- And the **VLAN20** Topology will look like this:-



→ Configure STP Port Settings:-

→ There are two main settings we can configure on a **Spanning Tree Port**: **<u>The cost</u>** and **<u>The Port Priority</u>**.
- As we can see, they are both configured on a **per-VLAN** basis like the Bridge Priority.

1- **cost**, is the Root Cost, remember the Chart in Day20.
- It is used primarily to determine the Root Port, and is also used as a tie-breaker in selecting Designated and non-Designated ports.

2- **The Priority**: The first half of the Port ID, which is the final tie-breaker in determining the Root Port.

→ Why would we change either of these values??
- To change the result of the Root Port or Designated Port selection process.

```
SW2(config-if)#spanning-tree vlan 1 ?
  cost           Change an interface's per VLAN spanning tree path cost
  port-priority  Change an interface's spanning tree port priority

SW2(config-if)#spanning-tree vlan 1
```

- First, we configured the **cost** of this interface, as we can see the range is from **1** to **200** Million.

```
SW2(config-if)#spanning-tree vlan 1 cost ?
  <1-200000000>  Change an interface's per VLAN spanning tree path cost

SW2(config-if)#spanning-tree vlan 1 cost 200
```

- Then we set the **port-priority**, which is configured in increments of **32**, from **0** to **224**.

```
SW2(config-if)#spanning-tree vlan 1 port-priority ?
  <0-224>  port priority in increments of 32

SW2(config-if)#spanning-tree vlan 1 port-priority 32
SW2(config-if)#
```

→ Those are the only **Spanning Tree** interface settings we need to know for the CCNA.

@0x0nullian

Day 22: Rapid Spanning Tree Protocol:-