Spanning Tree Protocol (STP) allows Ethernet LANs to have the added benefits of installing redundant links in a LAN, while overcoming the known problems that occur when adding those extra links. Using redundant links in a LAN design allows the LAN to keep working even when some links fail or even when some entire switches fail. Proper LAN design should add enough redundancy so that no single point of failure crashes the LAN; STP allows the design to use redundancy without causing some other problems.

STP affects many aspects of how switch forwarding logic works. Because Cisco puts the STP exam topics into the ICND2 half of the CCNA Routing and Switching exam, all the detailed examples in the ICND1 Cert Guide avoid showing redundant links in the LANs. For this ICND2 book, most of the LAN examples include redundancy. Therefore, you need to be prepared to rethink what you learned about LANs from reading the ICND1 book while thinking about LANs that have redundant links, and how STP and related features make those LANs work.

This chapter organizes the material into three sections. The first section presents core STP concepts that apply to most types of STP. STP has been improved and changed over the years, with Rapid STP (RSTP) being one major improvement. The first section looks at STP concepts without the RSTP logic added, while the second major section details RSTP concepts. The final major section discusses a small number of features that optimize and secure STP: PortFast, BPDU Guard, and EtherChannels.

As for the exam topics for this chapter, note that they all use the same three verbs: configure, verify, and troubleshoot. This chapter does not get into that level of depth on any of the specific topics, but instead lays the foundation to understand these features so that you
are prepared to delve into the configuration, verification, and troubleshooting details in Chapters 3 and 4.

“Do I Know This Already?” Quiz

Take the quiz (either here, or use the PCPT software) if you want to use the score to help you decide how much time to spend on this chapter. The answers are at the bottom of the page following the quiz, and the explanations are in DVD Appendix C and in the PCPT software.

Table 2-1 “Do I Know This Already?” Foundation Topics Section-to-Question Mapping

<table>
<thead>
<tr>
<th>Foundation Topics Section</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanning Tree Protocol (IEEE 802.1D)</td>
<td>1–4</td>
</tr>
<tr>
<td>Rapid STP (IEEE 802.1w) Concepts</td>
<td>5, 6</td>
</tr>
<tr>
<td>Optional STP Features</td>
<td>7</td>
</tr>
</tbody>
</table>

1. Which of the following IEEE 802.1D port states are stable states used when STP has completed convergence? (Choose two answers.)
   a. Blocking
   b. Forwarding
   c. Listening
   d. Learning
   e. Discarding

2. Which of the following are transitory IEEE 802.1D port states used only during the process of STP convergence? (Choose two answers.)
   a. Blocking
   b. Forwarding
   c. Listening
   d. Learning
   e. Discarding

3. Which of the following bridge IDs wins election as root, assuming that the switches with these bridge IDs are in the same network?
   a. 32769:0200.1111.1111
   b. 32769:0200.2222.2222
   c. 4097:0200.1111.1111
   d. 4097:0200.2222.2222
   e. 40961:0200.1111.1111

4. Which of the following facts determines how often a nonroot bridge or switch sends an 802.1D STP Hello BPDU message?
   a. The Hello timer as configured on that switch.
   b. The Hello timer as configured on the root switch.
   c. It is always every 2 seconds.
   d. The switch reacts to BPDUs received from the root switch by sending another BPDU 2 seconds after receiving the root BPDU.
5. Which of the following RSTP port states have the same name and purpose as a port state in traditional 802.1D STP? (Choose two answers.)
   a. Blocking
   b. Forwarding
   c. Listening
   d. Learning
   e. Discarding

6. RSTP adds some concepts to STP that enable ports to be used for a role if another port on the same switch fails. Which of the following statements correctly describe a port role that is waiting to take over for another port role? (Choose two answers.)
   a. An alternate port waits to become a root port.
   b. A backup port waits to become a root port.
   c. An alternate port waits to become a designated port.
   d. A backup port waits to become a designated port.

7. What STP feature causes an interface to be placed in the forwarding state as soon as the interface is physically active?
   a. STP
   b. EtherChannel
   c. Root Guard
   d. PortFast

**Foundation Topics**

**Spanning Tree Protocol (IEEE 802.1D)**

Without Spanning Tree Protocol (STP), a LAN with redundant links would cause Ethernet frames to loop for an indefinite period of time. With STP enabled, some switches block ports so that these ports do not forward frames. STP intelligently chooses which ports block, with two goals in mind:

- All devices in a VLAN can send frames to all other devices. In other words, STP does not block too many ports, cutting off some parts of the LAN from other parts.
- Frames have a short life and do not loop around the network indefinitely.

STP strikes a balance, allowing frames to be delivered to each device, without causing the problems that occur when frames loop through the network over and over again.

STP prevents looping frames by adding an additional check on each interface before a switch uses it to send or receive user traffic. That check: If the port is in STP forwarding state in that VLAN, use it as normal; if it is in STP blocking state, however, block all user traffic and do not send or receive user traffic on that interface in that VLAN.

Note that these STP states do not change the other information you already know about switch interfaces. The interface’s state of connected/notconnect does not change. The interface’s operational state as either an access or trunk port does not change. STP adds this additional STP state, with the blocking state basically disabling the interface.
In many ways, those last two paragraphs sum up what STP does. However, the details of how STP does its work can take a fair amount of study and practice. This first major section of the chapter begins by explaining the need for STP and the basic ideas of what STP does to solve the problem of looping frames. The majority of this section then looks at how STP goes about choosing which switch ports to block to accomplish STP’s goals.

The Need for Spanning Tree

STP prevents three common problems in Ethernet LANs. All three problems occur as a side effect of one fact: without STP, some Ethernet frames would loop around the network for a long time (hours, days, literally forever if the LAN devices and links never failed). By default, Cisco switches run STP, but you can disable STP. Do not disable it unless you know exactly what you are doing!

Just one looping frame causes what is called a broadcast storm. Broadcast storms happen when any kind of Ethernet frames—broadcast frames, multicast frames, or unknown-destination unicast frames—loop around a LAN indefinitely. Broadcast storms can saturate all the links with copies of that one single frame, crowding out good frames, as well as significantly impacting end-user device performance by making the PCs process too many broadcast frames.

To help you understand how this occurs, Figure 2-1 shows a sample network in which Bob sends a broadcast frame. The dashed lines show how the switches forward the frame when STP does not exist.

![Figure 2-1 Broadcast Storm](image-url)

Answers to the “Do I Know This Already?” quiz:

1 A, B 2 C, D 3 C 4 B 5 B, D 6 A, D 7 D
NOTE Bob’s original broadcast would also be forwarded around the other direction as well, with SW3 sending a copy of the original frame out its Gi0/1 port. To reduce clutter, Figure 2-1 does not show that frame.

Remember that LAN switch? That logic tells switches to flood broadcasts out all interfaces in the same VLAN except the interface in which the frame arrived. In Figure 2-1, that means SW3 forwards Bob’s frame to SW2, SW2 forwards the frame to SW1, SW1 forwards the frame back to SW3, and SW3 forwards it back to SW2 again.

When broadcast storms happen, frames like the one in Figure 2-1 keep looping until something changes—someone shuts down an interface, reloads a switch, or does something else to break the loop. Also note that the same event happens in the opposite direction. When Bob sends the original frame, SW3 also forwards a copy to SW1, SW1 forwards it to SW2, and so on.

The storm also causes a much more subtle problem called **MAC table instability**. MAC table instability means that the switches’ MAC address tables keep changing, because frames with the same source MAC arrive on different ports. To see why, follow this example, in which SW3 begins Figure 2-1 with a MAC table entry for Bob, at the bottom of the figure, associated with port Fa0/13:

\[
\begin{array}{ccc}
0200.3333.3333 & Fa0/13 & VLAN 1
\end{array}
\]

However, now think about the switch-learning process that occurs when the looping frame goes to SW2, then SW1, and then back into SW3’s Gi0/1 interface. SW3 thinks, “Hmm… the source MAC address is 0200.3333.3333, and it came in my Gi0/1 interface. Update my MAC table!” This results in the following entry on SW3, with interface Gi0/1 instead of Fa0/13:

\[
\begin{array}{ccc}
0200.3333.3333 & Gi0/1 & VLAN 1
\end{array}
\]

At this point, SW3 itself cannot correctly deliver frames to Bob’s MAC address. At that instant, if a frame arrives at SW3 destined for Bob—a different frame than the looping frame that causes the problems—SW3 incorrectly forwards the frame out Gi0/1 to SW1, creating even more congestion.

The looping frames in a broadcast storm also cause a third problem: multiple copies of the frame arrive at the destination. Consider a case in which Bob sends a frame to Larry but none of the switches know Larry’s MAC address. Switches flood frames sent to unknown destination unicast MAC addresses. When Bob sends the frame destined for Larry’s MAC address, SW3 sends a copy to both SW1 and SW2. SW1 and SW2 also flood the frame, causing copies of the frame to loop. SW1 also sends a copy of each frame out Fa0/11 to Larry. As a result, Larry gets multiple copies of the frame, which may result in an application failure, if not more pervasive networking problems.

Table 2-2 summarizes the main three classes of problems that occur when STP is not used in a LAN that has redundancy.
**Table 2-2** Three Classes of Problems Caused by Not Using STP in Redundant LANs

<table>
<thead>
<tr>
<th>Problem</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broadcast storms</td>
<td>The forwarding of a frame repeatedly on the same links, consuming significant parts of the links’ capacities</td>
</tr>
<tr>
<td>MAC table instability</td>
<td>The continual updating of a switch’s MAC address table with incorrect entries, in reaction to looping frames, resulting in frames being sent to the wrong locations</td>
</tr>
<tr>
<td>Multiple frame transmission</td>
<td>A side effect of looping frames in which multiple copies of one frame are delivered to the intended host, confusing the host</td>
</tr>
</tbody>
</table>

**What IEEE 802.1D Spanning Tree Does**

STP prevents loops by placing each switch port in either a forwarding state or a blocking state. Interfaces in the forwarding state act as normal, forwarding and receiving frames. However, interfaces in a blocking state do not process any frames except STP messages (and some other overhead messages). Interfaces that block do not forward user frames, do not learn MAC addresses of received frames, and do not process received user frames.

Figure 2-2 shows a simple STP tree that solves the problem shown in Figure 2-1 by placing one port on SW3 in the blocking state.

![Figure 2-2](image_url)

**Figure 2-2** What STP Does: Blocks a Port to Break the Loop

Now when Bob sends a broadcast frame, the frame does not loop. As shown in the steps in the figure:

**Step 1.** Bob sends the frame to SW3.
**Step 2.** SW3 forwards the frame only to SW1, but not out Gi0/2 to SW2, because SW3’s Gi0/2 interface is in a blocking state.
**Step 3.** SW1 floods the frame out both Fa0/11 and Gi0/1.
**Step 4.** SW2 floods the frame out Fa0/12 and Gi0/1.
**Step 5.** SW3 physically receives the frame, but it ignores the frame received from SW2 because SW3’s Gi0/2 interface is in a blocking state.

With the STP topology in Figure 2-2, the switches simply do not use the link between SW2 and SW3 for traffic in this VLAN, which is the minor negative side effect of STP. However, if either of the other two links fails, STP converges so that SW3 forwards instead of blocks on its Gi0/2 interface.

**NOTE** The term *STP convergence* refers to the process by which the switches collectively realize that something has changed in the LAN topology and determine whether they need to change which ports block and which ports forward.

That completes the description of what STP does, placing each port into either a forwarding or blocking state. The more interesting question, and the one that takes a lot more work to understand, is the question of how and why STP makes its choices. How does STP manage to make switches block or forward on each interface? And how does it converge to change state from blocking to forwarding to take advantage of redundant links in response to network outages? The following sections answer these questions.

**How Spanning Tree Works**

The STP algorithm creates a spanning tree of interfaces that forward frames. The tree structure of forwarding interfaces creates a single path to and from each Ethernet link, just like you can trace a single path in a living, growing tree from the base of the tree to each leaf.

**NOTE** STP was created before LAN switches even existed. In those days, Ethernet bridges used STP. Today, switches play the same role as bridges, implementing STP. However, many STP terms still refer to bridge. For the purposes of STP and this chapter, consider the terms *bridge* and *switch* synonymous.

The process used by STP, sometimes called the *spanning-tree algorithm* (STA), chooses the interfaces that should be placed into a forwarding state. For any interfaces not chosen to be in a forwarding state, STP places the interfaces in blocking state. In other words, STP simply picks which interfaces should forward, and any interfaces left over go to a blocking state.

STP uses three criteria to choose whether to put an interface in forwarding state:

- **STP elects a root switch.** STP puts all working interfaces on the root switch in forwarding state.

- **Each nonroot switch considers one of its ports to have the least administrative cost between itself and the root switch.** The cost is called that switch’s *root cost*. STP places its port that is part of the least root cost path, called that switch’s *root port* (RP), in forwarding state.

- **Many switches can attach to the same Ethernet segment, but in modern networks, normally two switches connect to each link.** The switch with the lowest root cost, as compared with the other switches attached to the same link, is placed in forwarding state.
That switch is the designated switch, and that switch’s interface, attached to that segment, is called the designated port (DP).

NOTE  The real reason the root switches place all working interfaces in a forwarding state is that all its interfaces will become DPs, but it is easier to just remember that all the root switches’ working interfaces will forward frames.

All other interfaces are placed in blocking state. Table 2-3 summarizes the reasons STP places a port in forwarding or blocking state.

**Table 2-3  STP: Reasons for Forwarding or Blocking**

<table>
<thead>
<tr>
<th>Characterization of Port</th>
<th>STP State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All the root switch’s ports</td>
<td>Forwarding</td>
<td>The root switch is always the designated switch on all connected segments.</td>
</tr>
<tr>
<td>Each nonroot switch’s root</td>
<td>Forwarding</td>
<td>The port through which the switch has the least cost to reach the root switch (lowest root cost).</td>
</tr>
<tr>
<td>port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Each LAN’s designated port</td>
<td>Forwarding</td>
<td>The switch forwarding the Hello on to the segment, with the lowest root cost, is the designated switch for that segment.</td>
</tr>
<tr>
<td>All other working ports</td>
<td>Blocking</td>
<td>The port is not used for forwarding user frames, nor are any frames received on these interfaces considered for forwarding.</td>
</tr>
</tbody>
</table>

NOTE  STP only considers working interfaces (those in a connected state). Failed interfaces (for example, interfaces with no cable installed) or administratively shutdown interfaces are instead placed into an STP disabled state. So, this section uses the term working ports to refer to interfaces that could forward frames if STP placed the interface into a forwarding state.

The STP Bridge ID and Hello BPDU

The STA begins with an election of one switch to be the root switch. To better understand this election process, you need to understand the STP messages sent between switches as well as the concept and format of the identifier used to uniquely identify each switch.

The STP bridge ID (BID) is an 8-byte value unique to each switch. The bridge ID consists of a 2-byte priority field and a 6-byte system ID, with the system ID being based on a universal (burned-in) MAC address in each switch. Using a burned-in MAC address ensures that each switch’s bridge ID will be unique.

STP defines messages called bridge protocol data units (BPDU), which switches use to exchange information with each other. The most common BPDU, called a Hello BPDU, lists many details, including the sending switch’s BID. By listing its own unique BID, switches can tell which switch sent which Hello BPDU. Table 2-4 lists some of the key information in the Hello BPDU.
### Table 2-4  Fields in the STP Hello BPDU

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root bridge ID</td>
<td>The bridge ID of the switch the sender of this Hello currently believes to be the root switch</td>
</tr>
<tr>
<td>Sender's bridge ID</td>
<td>The bridge ID of the switch sending this Hello BPDU</td>
</tr>
<tr>
<td>Sender's root cost</td>
<td>The STP cost between this switch and the current root</td>
</tr>
<tr>
<td>Timer values on the root switch</td>
<td>Includes the Hello timer, MaxAge timer, and forward delay timer</td>
</tr>
</tbody>
</table>

For the time being, just keep the first three items from Table 2-4 in mind as the following sections work through the three steps in how STP chooses the interfaces to place into a forwarding state. Next, the text examines the three main steps in the STP process.

### Electing the Root Switch

Switches elect a root switch based on the BIDs in the BPDUs. The root switch is the switch with the lowest numeric value for the BID. Because the two-part BID starts with the priority value, essentially the switch with the lowest priority becomes the root. For example, if one switch has priority 4096, and another switch has priority 8192, the switch with priority 4096 wins, regardless of what MAC address was used to create the BID for each switch.

If a tie occurs based on the priority portion of the BID, the switch with the lowest MAC address portion of the BID is the root. No other tiebreaker should be needed because switches use one of their own universal (burned-in) MAC addresses as the second part of their BIDs. So if the priorities tie, and one switch uses a MAC address of 0200.0000.0000 as part of the BID and the other uses 0911.1111.1111, the first switch (MAC 0200.0000.0000) becomes the root switch.

STP elects a root switch in a manner not unlike a political election. The process begins with all switches claiming to be the root by sending Hello BPDUs listing their own BID as the root BID. If a switch hears a Hello that lists a better (lower) BID, that switch stops advertising itself as root and starts forwarding the superior Hello. The Hello sent by the better switch lists the better switch's BID as the root. It works like a political race in which a less-popular candidate gives up and leaves the race, throwing his support behind the more popular candidate. Eventually, everyone agrees which switch has the best (lowest) BID, and everyone supports the elected switch—which is where the political race analogy falls apart.

**NOTE**  A better Hello, meaning that the listed root’s BID is better (numerically lower), is called a *superior Hello*; a worse Hello, meaning that the listed root’s BID is not as good (numerically higher), is called an *inferior Hello*.

Figure 2-3 shows the beginning of the root election process. In this case, SW1 has advertised itself as root, as have SW2 and SW3. However, SW2 now believes that SW1 is a better root, so SW2 is now forwarding the Hello originating at SW1. So, at this point, the figure shows SW1 is saying Hello, claiming to be root; SW2 agrees, and is forwarding SW1’s Hello that lists SW1 as root; but, SW3 is still claiming to be best, sending its own Hello BPDUs, listing SW3’s BID as the root.
Two candidates still exist in Figure 2-3: SW1 and SW3. So who wins? Well, from the BID, the lower-priority switch wins; if a tie occurs, the lower MAC address wins. As shown in the figure, SW1 has a lower BID (32769:0200.0001.0001) than SW3 (32769:0200.0003.0003), so SW1 wins, and SW3 now also believes that SW1 is the better switch. Figure 2-4 shows the resulting Hello messages sent by the switches.

After the election is complete, only the root switch continues to originate STP Hello BPDU messages. The other switches receive the Hellos, update the sender’s BID field (and root cost field), and forward the Hellos out other interfaces. The figure reflects this fact, with SW1 sending Hellos at Step 1, and SW2 and SW3 independently forwarding the Hello out their other interfaces at Step 2.
Summarizing, the root election happens through each switch claiming to be root, with the best switch being elected based on the numerically lowest BID. Breaking down the BID into its components, the comparisons can be made as:

- The lowest priority
- If that ties, the lowest switch MAC address

**Choosing Each Switch’s Root Port**

The second part of the STP process occurs when each nonroot switch chooses its one and only root port. A switch’s RP is its interface through which it has the least STP cost to reach the root switch (least root cost).

The idea of a switch’s cost to reach the root switch can be easily seen for humans. Just look at a network diagram that shows the root switch, lists the STP cost associated with each switch port, and identifies the nonroot switch in question. Switches use a different process than looking at a network diagram, of course, but using a diagram can make it easier to learn the idea.

Figure 2-5 shows just such a figure, with the same three switches shown in the last several figures. SW1 has already won the election as root, and the figure considers the cost from SW3’s perspective.

**Figure 2-5  How a Human Might Calculate STP Cost from SW3 to the Root (SW1)**

SW3 has two possible physical paths to send frames to the root switch: the direct path to the left, and the indirect path to the right through switch SW2. The cost is the sum of the costs of all the switch ports the frame would exit if it flowed over that path. (The calculation ignores the inbound ports.) As you can see, the cost over the direct path out SW3’s G0/1 port has a total cost of 5, and the other path has a total cost of 8. SW3 picks its G0/1 port as root port because it is the port that is part of the least-cost path to send frames to the root switch.

Switches come to the same conclusion, but using a different process. Instead, they add their local interface STP cost to the root cost listed in each received Hello BPDU.
port cost is simply an integer value assigned to each interface, per VLAN, for the purpose of providing an objective measurement that allows STP to choose which interfaces to add to the STP topology. The switches also look at their neighbor’s root cost, as announced in Hello BPDUs received from each neighbor.

Figure 2-6 shows an example of how switches calculate their best root cost and then choose their root port, using the same topology and STP costs as shown in Figure 2-5. STP on SW3 calculates its cost to reach the root over the two possible paths by adding the advertised cost (in Hello messages) to the interface costs listed in the figure.

Focus on the process for a moment. The root switch sends Hellos, with a listed root cost of 0. The idea is that the root’s cost to reach itself is 0.

Next, look on the left of the figure. SW3 takes the received cost (0) from the Hello sent by SW1, and adds the interface cost (5) of the interface on which that Hello was received. SW3 calculates that the cost to reach the root switch, out that port (G0/1), is 5.

On the right side, SW2 has realized its best cost to reach the root is cost 4. So, when SW2 forwards the Hello toward SW3, SW2 lists a root cost 4. SW3’s STP port cost on port G0/2 is 4, so SW3 determines a total cost to reach root out its G0/2 port of 8.

As a result of the process depicted in Figure 2-6, SW3 chooses Gi0/1 as its RP, because the cost to reach the root switch through that port (5) is lower than the other alternative (Gi0/2, cost 8). Similarly, SW2 chooses Gi0/2 as its RP, with a cost of 4 (SW1’s advertised cost of 0 plus SW2’s Gi0/2 interface cost of 4). Each switch places its root port into a forwarding state.

Figure 2-6 How STP Actually Calculates the Cost from SW3 to the Root

In more complex topologies, the choice of root port will not be so obvious. Chapter 4, “LAN Troubleshooting,” discusses these more complex examples, including the tiebreakers to use if the root costs tie.
Choosing the Designated Port on Each LAN Segment

STP's final step to choose the STP topology is to choose the designated port on each LAN segment. The designated port (DP) on each LAN segment is the switch port that advertises the lowest-cost Hello onto a LAN segment. When a nonroot switch forwards a Hello, the nonroot switch sets the root cost field in the Hello to that switch's cost to reach the root. In effect, the switch with the lower cost to reach the root, among all switches connected to a segment, becomes the DP on that segment.

For example, earlier Figure 2-4 shows in bold text the parts of the Hello messages from both SW2 and SW3 that determine the choice of DP on that segment. Note that both SW2 and SW3 list their respective cost to reach the root switch (cost 4 on SW2 and cost 5 on SW3). SW2 lists the lower cost, so SW2's Gi0/1 port is the designated port on that LAN segment.

All DPs are placed into a forwarding state; so in this case, SW2's Gi0/1 interface will be in a forwarding state.

If the advertised costs tie, the switches break the tie by choosing the switch with the lower BID. In this case, SW2 would also have won, with a BID of 32769:0200.0002.0002 versus SW3's 32769:0200.0003.0003.

NOTE Two additional tiebreakers are needed in some cases, although these would be unlikely today. A single switch can connect two or more interfaces to the same collision domain by connecting to a hub. In that case, the one switch hears its own BPDUs. So, if a switch ties with itself, two additional tiebreakers are used: the lowest interface STP priority and, if that ties, the lowest internal interface number.

The only interface that does not have a reason to be in a forwarding state on the three switches in the examples shown in Figures 2-3 through 2-6 is SW3's Gi0/2 port. So, the STP process is now complete. Table 2-5 outlines the state of each port and shows why it is in that state.

<table>
<thead>
<tr>
<th>Switch Interface</th>
<th>State</th>
<th>Reason Why the Interface Is in Forwarding State</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1, Gi0/1</td>
<td>Forwarding</td>
<td>The interface is on the root switch, so it becomes the DP on that link.</td>
</tr>
<tr>
<td>SW1, Gi0/2</td>
<td>Forwarding</td>
<td>The interface is on the root switch, so it becomes the DP on that link.</td>
</tr>
<tr>
<td>SW2, Gi0/2</td>
<td>Forwarding</td>
<td>The root port of SW2.</td>
</tr>
<tr>
<td>SW2, Gi0/1</td>
<td>Forwarding</td>
<td>The designated port on the LAN segment to SW3.</td>
</tr>
<tr>
<td>SW3, Gi0/1</td>
<td>Forwarding</td>
<td>The root port of SW3.</td>
</tr>
<tr>
<td>SW3, Gi0/2</td>
<td>Blocking</td>
<td>Not the root port and not the designated port.</td>
</tr>
</tbody>
</table>

Influencing and Changing the STP Topology

Switches do not just use STP once and never again. The switches continually watch for changes. Those changes can be because a link or switch fails or it can be a new link that can now be used. The configuration can change in a way that changes the STP topology. This section briefly discusses the kinds of things that change the STP topology, either through configuration or through changes in the status of devices and links in the LAN.
Making Configuration Changes to Influence the STP Topology

The network engineers can choose to change the STP settings to then change the choices STP makes in a given LAN. Two main tools available to the engineer are to configure the bridge ID and to change STP port costs.

Switches have a way to create a default BID, by taking a default priority value, and adding a universal MAC address that comes with the switch hardware. However, engineers typically want to choose which switch becomes the root. Chapter 3, “Spanning Tree Protocol Implementation,” shows how to configure a Cisco switch to override its default BID setting to make a switch become root.

Port costs also have default values, per port, per VLAN. You can configure these port costs, or you can use the default values. Table 2-6 lists the default port costs suggested by IEEE. IOS on Cisco switches has long used the default settings as defined in the 1998 version of the 802.1D standard. The newer standard, useful when using links faster than 10 Gbps, can be used by adding a single configuration command to each switch (spanning-tree pathcost method long).

<table>
<thead>
<tr>
<th>Ethernet Speed</th>
<th>IEEE Cost: 1998 (and Before)</th>
<th>IEEE Cost: 2004 (and After)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Mbps</td>
<td>100</td>
<td>2,000,000</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>19</td>
<td>200,000</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>4</td>
<td>20,000</td>
</tr>
<tr>
<td>10 Gbps</td>
<td>2</td>
<td>2000</td>
</tr>
<tr>
<td>100 Gbps</td>
<td>N/A</td>
<td>200</td>
</tr>
<tr>
<td>1 Tbps</td>
<td>N/A</td>
<td>20</td>
</tr>
</tbody>
</table>

With STP enabled, all working switch interfaces will settle into an STP forwarding or blocking state, even access ports. For switch interfaces connected to hosts or routers, which do not use STP, the switch still forwards Hellos on to those interfaces. By virtue of being the only device sending a Hello onto that LAN segment, the switch is sending the least-cost Hello on to that LAN segment, making the switch become the designated port on that LAN segment. So, STP puts working access interfaces into a forwarding state as a result of the designated port part of the STP process.

Reacting to State Changes That Affect the STP Topology

Once the engineer has finished all STP configuration, the STP topology should settle into a stable state and not change, at least until the network topology changes. This section examines the ongoing operation of STP while the network is stable, and then it covers how STP converges to a new topology when something changes.

The root switch sends a new Hello BPDU every 2 seconds by default. Each nonroot switch forwards the Hello on all DPs, but only after changing items listed in the Hello. The switch sets the root cost to that local switch’s calculated root cost. The switch also sets the “sender’s bridge ID” field to its own bridge ID. (The root’s bridge ID field is not changed.)
By forwarding the received (and changed) Hellos out all DPs, all switches continue to receive Hellos every 2 seconds. The following steps summarize the steady-state operation when nothing is currently changing in the STP topology:

**Step 1.** The root creates and sends a Hello BPDU, with a root cost of 0, out all its working interfaces (those in a forwarding state).

**Step 2.** The nonroot switches receive the Hello on their root ports. After changing the Hello to list their own BID as the sender’s BID, and listing that switch’s root cost, the switch forwards the Hello out all designated ports.

**Step 3.** Steps 1 and 2 repeat until something changes.

Each switch relies on these periodically received Hellos from the root as a way to know that its path to the root is still working. When a switch ceases to receive the Hellos, or receives a Hello that lists different details, something has failed, so the switch reacts and starts the process of changing the spanning-tree topology.

### How Switches React to Changes with STP

For various reasons, the convergence process requires the use of three timers. Note that all switches use the timers as dictated by the root switch, which the root lists in its periodic Hello BPDU messages. Table 2-7 describes the timers.

<table>
<thead>
<tr>
<th>Timer</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hello</td>
<td>2 seconds</td>
<td>The time period between Hellos created by the root.</td>
</tr>
<tr>
<td>MaxAge</td>
<td>10 times Hello</td>
<td>How long any switch should wait, after ceasing to hear Hellos, before trying to change the STP topology.</td>
</tr>
<tr>
<td>Forward delay</td>
<td>15 seconds</td>
<td>Delay that affects the process that occurs when an interface changes from blocking state to forwarding state. A port stays in an interim listening state, and then an interim learning state, for the number of seconds defined by the forward delay timer.</td>
</tr>
</tbody>
</table>

If a switch does not get an expected Hello BPDU within the Hello time, the switch continues as normal. However, if the Hellos do not show up again within MaxAge time, the switch reacts by taking steps to change the STP topology. With default settings, MaxAge is 20 seconds (10 times the default Hello timer of 2 seconds). So, a switch would go 20 seconds without hearing a Hello before reacting.

After MaxAge expires, the switch essentially makes all its STP choices again, based on any Hellos it receives from other switches. It reevaluates which switch should be the root switch. If the local switch is not the root, it chooses its RP. And it determines whether it is DP on each of its other links. The best way to describe STP convergence is to show an example using the same familiar topology. Figure 2-7 shows the same familiar figure, with SW3’s Gi0/2 in a blocking state, but SW1’s Gi0/2 interface has just failed.

SW3 reacts to the change because SW3 fails to receive its expected Hellos on its Gi0/1 interface. However, SW2 does not need to react because SW2 continues to receive its periodic Hellos in its Gi0/2 interface. In this case, SW3 reacts either when MaxAge time passes without hearing the Hellos, or as soon as SW3 notices that interface Gi0/1 has failed. (If the interface fails, the switch can assume that the Hellos will not be arriving in that interface anymore.)
Now that SW3 can act, it begins by reevaluating the choice of root switch. SW3 still receives the Hellos from SW2, as forwarded from the root (SW1). SW1 still has a lower BID than SW3; otherwise, SW1 would not have already been the root. So, SW3 decides that SW1 is still the best switch and that SW3 is not the root.

Next, SW3 reevaluates its choice of RP. At this point, SW3 is receiving Hellos on only one interface: Gi0/2. Whatever the calculated root cost, Gi0/2 becomes SW3’s new RP. (The cost would be 8, assuming the STP costs had no changes since Figures 2-5 and 2-6.)

SW3 then reevaluates its role as DP on any other interfaces. In this example, no real work needs to be done. SW3 was already DP on interface Fa0/13, and it continues to be the DP because no other switches connect to that port.

**Changing Interface States with STP**

STP uses the idea of roles and states. *Roles*, like root port and designated port, relate to how STP analyzes the LAN topology. *States*, like forwarding and blocking, tell a switch whether to send or receive frames. When STP converges, a switch chooses new port roles, and the port roles determine the state (forwarding or blocking).

Switches can simply move immediately from forwarding to blocking state, but they must take extra time to transition from blocking state to forwarding state. For instance, when a switch formerly used port G0/1 as its RP (a role), that port was in a forwarding state. After convergence, G0/1 might be neither an RP nor DP; the switch can immediately move that port to a blocking state.
When a port that formerly blocked needs to transition to forwarding, the switch first puts the port through two intermediate interface states. These temporary states help prevent temporary loops:

- **Listening**: Like the blocking state, the interface does not forward frames. The switch removes old stale (unused) MAC table entries for which no frames are received from each MAC address during this period. These stale MAC table entries could be the cause of the temporary loops.

- **Learning**: Interfaces in this state still do not forward frames, but the switch begins to learn the MAC addresses of frames received on the interface.

STP moves an interface from blocking to listening, then to learning, and then to forwarding state. STP leaves the interface in each interim state for a time equal to the forward delay timer, which defaults to 15 seconds. As a result, a convergence event that causes an interface to change from blocking to forwarding requires 30 seconds to transition from blocking to forwarding. In addition, a switch might have to wait MaxAge seconds before even choosing to move an interface from blocking to forwarding state.

For example, follow what happens with an initial STP topology as shown in Figures 2-3 through 2-6, with the SW1-to-SW3 link failing as shown in Figure 2-7. If SW1 simply quit sending Hello messages to SW3, but the link between the two did not fail, SW3 would wait MaxAge seconds before reacting (20 seconds is the default). SW3 would actually quickly choose its ports' STP roles, but then wait 15 seconds each in listening and learning states on interface Gi0/2, resulting in a 50-second convergence delay.

Table 2-8 summarizes spanning tree’s various interface states for easier review.

<table>
<thead>
<tr>
<th>State</th>
<th>Forwards Data Frames?</th>
<th>Learns MACs Based on Received Frames?</th>
<th>Transitory or Stable State?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocking</td>
<td>No</td>
<td>No</td>
<td>Stable</td>
</tr>
<tr>
<td>Listening</td>
<td>No</td>
<td>No</td>
<td>Transitory</td>
</tr>
<tr>
<td>Learning</td>
<td>No</td>
<td>Yes</td>
<td>Transitory</td>
</tr>
<tr>
<td>Forwarding</td>
<td>Yes</td>
<td>Yes</td>
<td>Stable</td>
</tr>
<tr>
<td>Disabled</td>
<td>No</td>
<td>No</td>
<td>Stable</td>
</tr>
</tbody>
</table>

**Rapid STP (IEEE 802.1w) Concepts**

The original STP worked well given the assumptions about networks and networking devices in that era. However, as with any computing or networking standard, as time passes, hardware and software capabilities improve, so new protocols emerge to take advantage of those new capabilities. For STP, one of the most significant improvements over time has been the introduction of Rapid Spanning Tree Protocol (RSTP), introduced as standard IEEE 802.1w.

Before getting into the details of RSTP, it helps to make sense of the standards numbers a bit. 802.1w was actually an amendment to the 802.1D standard. 802.1D was published anew in 1998 (and a few times before that). After the 1998 version of 802.1D, the IEEE published the 802.1w amendment in 2001. Later, when the IEEE 802.1 committee updated the 802.1D standard in 2004, the IEEE pulled the 802.1w amendment details into the 802.1D-2004 standard.
So, why do we care? Sometimes people use the term STP to refer to the original pre-RSTP rules for STP. Some use STP to mean anything in the 802.1D standard, which now includes RSTP. So for real life, make sure you know what people mean when they use STP: do they mean STP to include RSTP concepts, or not? For this book, throughout the book, if the distinction between STP and RSTP matters, the book will use STP for the original STP rules and RSTP for the new ones introduced by 802.1w.

NOTE The IEEE sells its standards, but through the “Get IEEE 802” program, you can get free PDFs of the current 802 standards. To read about RSTP 802.1w, you will need to download the 802.1D standard, and then look for the sections about RSTP.

Now on to the details about RSTP in this chapter. There are similarities between RSTP and STP, so this section next compares and contrasts the two. Following that, the rest of this section discusses the concepts unique to RSTP that are not found in STP: alternate root ports, different port states, backup ports, and the port roles used by RSTP.

Comparing STP and RSTP

RSTP (802.1w) works just like STP (the original 802.1D) in several ways:

- It elects the root switch using the same parameters and tiebreakers.
- It elects the root port on nonroot switches with the same rules.
- It elects designated ports on each LAN segment with the same rules.
- It places each port in either forwarding or blocking state, although RSTP calls the blocking state the discarding state.

In fact, RSTP works so much like STP that they can both be used in the same network. RSTP and STP switches can be deployed in the same network, with RSTP features working in switches that support it, and traditional 802.1D STP features working in the switches that support only STP.

With all these similarities, you might be wondering why the IEEE bothered to create RSTP in the first place. The overriding reason is convergence. STP takes a relatively long time to converge (50 seconds with the default settings when all the wait times must be followed). RSTP improves network convergence when topology changes occur, usually converging within a few seconds (or in slow conditions, in about 10 seconds).

IEEE 802.1w RSTP changes and adds to IEEE 802.1D STP in ways that avoid waiting on STP timers, resulting in quick transitions from forwarding to blocking state and vice versa. Specifically, RSTP, compared to STP, defines more cases in which the switch can avoid waiting for a timer to expire, such as the following:

- Adds a new mechanism to replace the root port, without any waiting to reach a forwarding state (in some conditions)
- Adds a new mechanism to replace a designated port, without any waiting to reach a forwarding state (in some conditions)
- Lowers waiting times for cases in which RSTP must wait

For instance, when a link remains up, but Hello BPDUs simply stop arriving regularly on a port, STP requires a switch to wait for MaxAge seconds. STP defines the MaxAge timers
based on ten times the Hello timer, or 20 seconds, by default. RSTP shortens this timer, defining MaxAge as three times the Hello timer.

The best way to get a sense for these mechanisms is to see how the RSTP alternate port and the backup port both work. RSTP uses the term alternate port to refer to a switch’s other ports that could be used as root port if the root port ever fails. The backup port concept provides a backup port on the local switch for a designated port, but only applies to some topologies that frankly do not happen often with a modern network design. However, both are instructive about how RSTP works. Table 2-9 lists these RSTP port roles.

<table>
<thead>
<tr>
<th>Function</th>
<th>Port Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonroot switch’s best path to the root</td>
<td>Root port</td>
</tr>
<tr>
<td>Replaces the root port when the root port fails</td>
<td>Alternate port</td>
</tr>
<tr>
<td>Switch port designated to forward onto a collision domain</td>
<td>Designated port</td>
</tr>
<tr>
<td>Replaces a designated port when a designated port fails</td>
<td>Backup port</td>
</tr>
<tr>
<td>Port that is administratively disabled</td>
<td>Disabled port</td>
</tr>
</tbody>
</table>

RSTP and the Alternate (Root) Port Role

With STP, each nonroot switch places one port in the STP root port (RP) role. RSTP follows that same convention, with the same exact rules for choosing the RP. RSTP then takes another step, naming other possible RPs, identifying them as alternate ports.

To be an alternate port, both the RP and the alternate port must receive Hellos that identify the same root switch. For instance, in Figure 2-8, SW1 is the root. SW3 will receive Hello BPDUs on two ports: G0/1 and G0/2. Both Hellos list SW1’s bridge ID (BID) as the root switch, so whichever port is not the root port meets the criteria to be an alternate port. SW3 picks G0/1 as its root port in this case, and then makes G0/2 an alternate port.

![Figure 2-8 Example of SW3 Making G0/2 Become an Alternate Port](image)

An alternate port basically works like the second-best option for root port. The alternate port can take over for the former root port, often very rapidly, without requiring a wait in
other interim RSTP states. For instance, when the root port fails, or when Hellos stop arriving on the original root port, the switch changes the former root port’s role and state: (a) the role from root port to a disabled port, and (b) the state from forwarding to discarding (the equivalent of STP’s blocking state). Then, without waiting on any timers, the switch changes roles and state for the alternate port: its role changes to be the root port, with a forwarding state.

Notably, the new root port also does not need to spend time in other states, such as learning state, instead moving immediately to forwarding state.

Figure 2-9 shows an example of RSTP convergence. SW3’s root port before the failure shown in this figure is SW3’s G0/1, the link connected directly to SW1 (the root switch). Then SW3’s link to SW1 fails as shown in Step 1 of the figure.

```
Figure 2-9  Convergence Events with SW3 G0/1 Failure
```

Following the steps in Figure 2-9:

**Step 1.** The link between SW1 and SW3 fails, so that SW3’s current root port (Gi0/1) fails.

**Step 2.** SW3 and SW2 exchange RSTP messages to confirm that SW3 will now transition its former alternate port (Gi0/2) to be the root port. This action causes SW2 to flush the required MAC table entries.

**Step 3.** SW3 transitions G0/1 to the disabled role and G0/2 to the root port role.

**Step 4.** SW3 transitions G0/2 to a forwarding state immediately, without using learning state, because this is one case in which RSTP knows the transition will not create a loop.

As soon as SW3 realizes its G0/1 interface has failed, the process shown in the figure takes very little time. None of the processes rely on timers, so as soon as the work can be done, the convergence completes. (This particular convergence example takes about 1 second in a lab.)
RSTP States and Processes

The depth of the previous example does not point out all details of RSTP, of course; however, the example does show enough details to discuss RSTP states and internal processes.

Both STP and RSTP use *port states*, but with some differences. First, RSTP keeps both the learning and forwarding states as compared with STP, for the same purposes. However, RSTP does not even define a listening state, finding it unnecessary. Finally, RSTP renames the blocking state to the discarding state, and redefines its use slightly.

RSTP uses the discarding state for what 802.1D defines as two states: disabled state and blocking state. Blocking should be somewhat obvious by now: The interface can work physically, but STP/RSTP chooses to not forward traffic to avoid loops. STP’s disabled state simply meant that the interface was administratively disabled. RSTP just combines those into a single discarding state. Table 2-10 shows the list of STP and RSTP states for comparison purposes.

<table>
<thead>
<tr>
<th>Function</th>
<th>802.1D State</th>
<th>802.1w State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port is administratively disabled</td>
<td>Disabled</td>
<td>Discarding</td>
</tr>
<tr>
<td>Stable state that ignores incoming data frames and is not used to forward data frames</td>
<td>Blocking</td>
<td>Discarding</td>
</tr>
<tr>
<td>Interim state without MAC learning and without forwarding</td>
<td>Listening</td>
<td>Not used</td>
</tr>
<tr>
<td>Interim state with MAC learning and without forwarding</td>
<td>Learning</td>
<td>Learning</td>
</tr>
<tr>
<td>Stable state that allows MAC learning and forwarding of data frames</td>
<td>Forwarding</td>
<td>Forwarding</td>
</tr>
</tbody>
</table>

RSTP also changes some processes and message content (compared to STP) to speed convergence. For example, STP waits for a time (forward delay) in both listening and learning states. The reason for this delay in STP is that, at the same time, the switches have all been told to time out their MAC table entries. When the topology changes, the existing MAC table entries may actually cause a loop. With STP, the switches all tell each other (with BPDU messages) that the topology has changed, and to time out any MAC table entries using the forward delay timer. This removes the entries, which is good, but it causes the need to wait in both listening and learning state for forward delay time (default 15 seconds each).

RSTP, to converge more quickly, avoids relying on timers. RSTP switches tell each other (using messages) that the topology has changed. Those messages also direct neighboring switches to flush the contents of their MAC tables in a way that removes all the potentially loop-causing entries, without a wait. As a result, RSTP creates more scenarios in which a formerly discarding port can immediately transition to a forwarding state, without waiting, and without using the learning state, as shown in the example in Figure 2-9.

RSTP and the Backup (Designated) Port Role

The RSTP backup port role acts as yet another new RSTP port role as compared to STP. As a reminder, the RSTP alternate port role creates a way for RSTP to quickly replace a switch’s root port. Similarly, the RSTP backup port role creates a way for RSTP to quickly replace a switch’s designated port on some LAN.
The need for a backup port can be a bit confusing at first, because the need for the backup port role only happens in designs that are a little unlikely today. The reason is that a design must use hubs, which then allows the possibility that one switch connects more than one port to the same collision domain.

Figure 2-10 shows an example. SW3 and SW4 both connect to the same hub. SW4’s port F0/1 happens to win the election as designated port (DP). The other port on SW4 that connects to the same collision domain, F0/2, acts as a backup port.

With a backup port, if the current designated port fails, SW4 can start using the backup port with rapid convergence. For instance, if SW4’s F0/1 interface were to fail, SW4 could transition F0/2 to the designated port role, without any delay in moving from discarding state to a forwarding state.

**Figure 2-10  RSTP Backup Port Example**

**RSTP Port Types**

The final RSTP concept included here relates to some terms RSTP uses to refer to different types of ports and the links that connect to those ports.

To begin, consider the basic figure of Figure 2-11. It shows several links between two switches. RSTP considers these links to be point-to-point links and the ports connected to them to be point-to-point ports, because the link connects exactly two devices (points).

RSTP further classifies point-to-point ports into two categories. Point-to-point ports that connect two switches are not at the edge of the network and are simply called **point-to-point ports**. Ports that instead connect to a single endpoint device at the edge of the network, like a PC or server, are called **point-to-point edge ports**, or simply **edge ports**. In Figure 2-11, SW3’s switch port connected to a PC is an edge port.

Finally, RSTP defines the term **shared** to describe ports connected to a hub. The term **shared** comes from the fact that hubs create a shared Ethernet; hubs also force the attached switch port to use half-duplex logic. RSTP assumes that all half-duplex ports may be connected to hubs, treating ports that use half duplex as shared ports. RSTP converges more slowly on shared ports as compared to all point-to-point ports.
Optional STP Features

At this point, you have learned plenty of details that will be useful to next configure and verify STP operations, as discussed in Chapter 3. However, before moving to that chapter, the final section of the chapter briefly introduces a few related topics that make STP work even better or be more secure: EtherChannel, PortFast, and BPDU Guard.

EtherChannel

One of the best ways to lower STP’s convergence time is to avoid convergence altogether. EtherChannel provides a way to prevent STP convergence from being needed when only a single port or cable failure occurs.

EtherChannel combines multiple parallel segments of equal speed (up to eight) between the same pair of switches, bundled into an EtherChannel. The switches treat the EtherChannel as a single interface with regard to STP. As a result, if one of the links fails, but at least one of the links is up, STP convergence does not have to occur. For example, Figure 2-12 shows the familiar three-switch network, but now with two Gigabit Ethernet connections between each pair of switches.
With each pair of Ethernet links configured as an EtherChannel, STP treats each EtherChannel as a single link. In other words, both links to the same switch must fail for a switch to need to cause STP convergence. Without EtherChannel, if you have multiple parallel links between two switches, STP blocks all the links except one. With EtherChannel, all the parallel links can be up and working at the same time, while reducing the number of times STP must converge, which in turn makes the network more available.

When a switch makes a forwarding decision to send a frame out an EtherChannel, the switch then has to take an extra step in logic: Out which physical interface does it send the frame? The switch has load-balancing logic that lets it pick an interface for each frame, with a goal of spreading the traffic load across all active links in the channel. As a result, a LAN design that uses EtherChannels makes much better use of the available bandwidth between switches, while also reducing the number of times that STP must converge.

Note that EtherChannels may be Layer 2 EtherChannels (as described here) or Layer 3 EtherChannels (as discussed in Chapter 19, “IPv4 Routing in the LAN”). Layer 2 EtherChannels combine links that switches use as switch ports, with the switches using Layer 2 switching logic to forward and receive Ethernet frames over the EtherChannels. Layer 3 EtherChannels also combine links, but the switches use Layer 3 routing logic to forward packets over the EtherChannels. All references to EtherChannel in Part I of this book refer to Layer 2 EtherChannels unless otherwise noted.

**PortFast**

PortFast allows a switch to immediately transition from blocking to forwarding, bypassing listening and learning states. However, the only ports on which you can safely enable PortFast are ports on which you know that no bridges, switches, or other STP-speaking devices are connected. Otherwise, using PortFast risks creating loops, the very thing that the listening and learning states are intended to avoid.

PortFast is most appropriate for connections to end-user devices. If you turn on PortFast on ports connected to end-user devices, when an end-user PC boots, the switch port can move to an STP forwarding state and forward traffic as soon as the PC NIC is active. Without PortFast, each port must wait while the switch confirms that the port is a DP, and then wait while the interface sits in the temporary listening and learning states before settling into the forwarding state.

PortFast is a popular feature for edge ports; in fact, RSTP incorporates PortFast concepts. You may recall the mention of RSTP port types, particularly point-to-point edge port types, around Figure 2-11. RSTP, by design of the protocol, converges quickly on these point-to-point edge type ports by bypassing the learning state, which is the same idea Cisco originally introduced with PortFast. In practice, Cisco switches enable RSTP point-to-point edge ports by enabling PortFast on the port.

**BPDU Guard**

STP opens up the LAN to several different types of possible security exposures. For example:

- An attacker could connect a switch to one of these ports, one with a low STP priority value, and become the root switch. The new STP topology could have worse performance than the desired topology.
The attacker could plug into multiple ports, into multiple switches, become root, and actually forward much of the traffic in the LAN. Without the networking staff realizing it, the attacker could use a LAN analyzer to copy large numbers of data frames sent through the LAN.

Users could innocently harm the LAN when they buy and connect an inexpensive consumer LAN switch (one that does not use STP). Such a switch, without any STP function, would not choose to block any ports and could cause a loop.

The Cisco BPDU Guard feature helps defeat these kinds of problems by disabling a port if any BPDUs are received on the port. So, this feature is particularly useful on ports that should be used only as an access port and never connected to another switch.

In addition, the BPDU Guard feature helps prevent problems with PortFast. PortFast should be enabled only on access ports that connect to user devices, not to other LAN switches. Using BPDU Guard on these same ports makes sense because if another switch connects to such a port, the local switch can disable the port before a loop is created.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter’s material using either the tools in the book, DVD, or interactive tools for the same material found on the book’s companion website. Refer to the “Your Study Plan” element for more details. Table 2-11 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 2-11  Chapter Review Tracking

<table>
<thead>
<tr>
<th>Review Element</th>
<th>Review Date(s)</th>
<th>Resource Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review key topics</td>
<td></td>
<td>Book, DVD/website</td>
</tr>
<tr>
<td>Review key terms</td>
<td></td>
<td>Book, DVD/website</td>
</tr>
<tr>
<td>Answer DIKTA questions</td>
<td></td>
<td>Book, PCPT</td>
</tr>
<tr>
<td>Review memory tables</td>
<td></td>
<td>Book, App</td>
</tr>
</tbody>
</table>

Review All the Key Topics

Table 2-12  Key Topics for Chapter 2

<table>
<thead>
<tr>
<th>Key Topic Element</th>
<th>Description</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 2-2</td>
<td>Lists the three main problems that occur when not using STP in a LAN with redundant links</td>
<td>47</td>
</tr>
<tr>
<td>Table 2-3</td>
<td>Lists the reasons why a switch chooses to place an interface into forwarding or blocking state</td>
<td>49</td>
</tr>
<tr>
<td>Table 2-4</td>
<td>Lists the most important fields in Hello BPDU messages</td>
<td>50</td>
</tr>
<tr>
<td>List</td>
<td>Logic for the root switch election</td>
<td>52</td>
</tr>
<tr>
<td>Figure 2-6</td>
<td>Shows how switches calculate their root cost</td>
<td>53</td>
</tr>
</tbody>
</table>
Table 2-6  Lists the original and current default STP port costs for various interface speeds  55
Step list  A summary description of steady-state STP operations  56
Table 2-7  STP timers  56
List  Definitions of what occurs in the listening and learning states  58
Table 2-8  Summary of 802.1D states  58
List  Key similarities between 802.1D STP and 802.1w RSTP  59
Table 2-9  List of 802.1w port roles  60
Table 2-10  Comparisons of port states with 802.1D and 802.1w  62

Key Terms You Should Know
blocking state, BPDU Guard, bridge ID, bridge protocol data unit (BPDU), designated port, EtherChannel, forward delay, forwarding state, Hello BPDU, IEEE 802.1D, learning state, listening state, MaxAge, PortFast, root port, root switch, root cost, Spanning Tree Protocol (STP), rapid STP (RSTP), IEEE 802.1w, alternate port, backup port, disabled port, discarding state

This is Chapter 2: Spanning Tree Protocol Concepts from the CCNA Routing and Switching ICND2 200-105 Official Cert Guide by Wendell Odom.

This free chapter is brought to you by Cisco Press and the Cisco Learning Network, Your Lifelong Learning IT Community.

For more information and to purchase this book, visit: www.ciscopress.com/ccna. For more information on the CCNA Routing and Switching Certification, including Cisco’s Exam Topics visit: www.cisco.com/go/ccna.