Understanding Rapid Spanning Tree Protocol (802.1w)

Contents

Introduction

The current 802.1d Spanning Tree Protocol (STP) standard was designed at a time where recovering connectivity after an outage within a minute or so was considered adequate performance. With the advent of Layer 3 (L3) switching in LAN environments, bridging now competes with routed solutions where protocols such as Open Shortest Path First (OSPF) and Enhanced Interior Gateway Routing Protocol (EIGRP) are able to provide an alternate path in less time.

Cisco enhanced the original 802.1d specification with features such as Uplink Fast, Backbone Fast, and Port Fast to speed up the convergence time of a bridged network. The drawback is that these mechanisms are proprietary and need additional configuration.

Rapid Spanning Tree Protocol (RSTP; IEEE 802.1w) can be seen as an evolution of the 802.1d standard more than a revolution. The 802.1d terminology remains primarily the same, and most parameters have been left unchanged so users familiar with 802.1d can rapidly configure the new protocol comfortably. In most cases, RSTP performs better than Cisco's proprietary extensions without any additional configuration. 802.1w is also capable of reverting back to 802.1d in order to interoperate with legacy bridges (thus dropping the benefits it introduces) on a per-port basis. This document briefly explains the enhancements added by RSTP to the previous 802.1d standard.

Support of RSTP in Catalyst Switches

The table below shows the support of RSTP in Catalyst switches, and the minimum software required for that support.

Catalyst Platform	MST w/ RSTP	RPVST+ (also known as PVRST+)
Catalyst 2900 XL / 3500 XL	Not available.	Not available.
Catalyst 2940	Not available.	Not available.
Catalyst 2950/2955/3550	12.1(9)EA1	12.1(13)EA1
Catalyst 2970/3750	12.1(14)EA1	12.1(14)EA1
Catalyst 3560	12.1(19)EA1	12.1(19)EA1

Catalyst 3750 Metro	12.1(14)AX	12.1(14)AX
Catalyst 2948G-L3/4908G-L3	Not available.	Not available.
Catalyst 4000/2948G/2980G (CatOS)	7.1	7.5
Catalyst 4000/4500 (IOS)	12.1(12c)EW	12.1(19)EW
Catalyst 5000/5500	Not available.	Not available.
Catalyst 6000/6500	7.1	7.5
Catalyst 6000/6500 (IOS)	12.1(11b) EX, 12.1(13) E, 12.2(14) SX	12.1(13)E
Catalyst 8500	Not available.	Not available.

New Port States and Port Roles

The 802.1d is defined in four different port states: `listening`, `learning`, `blocking`, and `forwarding`. Refer to the table below for more information. The state of the port is mixed (whether it blocks or forwards traffic), and the role it plays in the active topology (root port, designated port, and so on). For example, from an operational point of view, there is no difference between a port in `blocking` state and a port in `listening` state; they both discard frames and do not learn MAC addresses. The real difference lies in the role the spanning tree assigns to the port. It can safely be assumed that a listening port is either designated or root and is on its way to the `forwarding` state. Unfortunately, once in `forwarding` state, there is no way to infer from the port state whether the port is root or designated, which contributes to demonstrating the failure of this state-based terminology. RSTP addresses this by decoupling the role and the state of a port.

Port States

There are only three port states left in RSTP, corresponding to the three possible operational states. The 802.1d states `disabled`, `blocking`, and `listening` have been merged into a unique 802.1w discarding state.

STP (802.1D) Port State	RSTP (802.1w) Port State	Is Port Included in Active Topology?	Is Port Learning MAC Addresses?
Disabled	Discarding	No	No
Blocking	Discarding	No	No
Listening	Discarding	Yes	No
Learning	Learning	Yes	Yes
Forwarding	Forwarding	Yes	Yes

Port Roles

The role is now a variable assigned to a given port. The root port and designated port roles remain, while the blocking port role is now split into the backup and alternate port roles. The Spanning Tree Algorithm (STA) determines the role of a port based on Bridge Protocol Data Units (BPDUs). To keep things simple, the thing to remember about a BPDU is that there is always a way of comparing any two of them and deciding whether one is more useful than the other. This is based on the value stored in the BPDU and occasionally on the port on which they are received. This considered, the following paragraphs explain very practical approaches to port roles.

Root Port Roles

- The port receiving the best BPDU on a bridge is the root port: This is the port that is the closest to the root bridge in terms of path cost. The STA elects a single root bridge in the whole bridged network (per-VLAN). The root bridge sends BPDUs that are more useful than the ones that any other bridge can send. The root bridge is the only bridge in the network that does not have a root port. All other bridges receive BPDUs on at least one port.

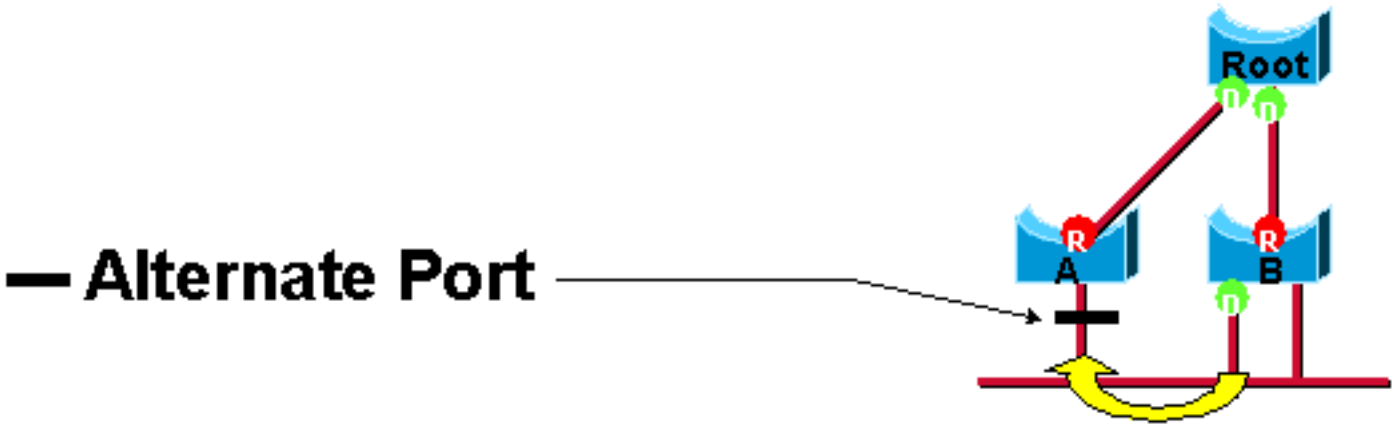
Designated Port Role

- A port is designated if it can send the best BPDU on the segment to which it is connected. 802.1d bridges create a bridged domain by linking together different segments (Ethernet segments for example). On a given segment, there can be only one path toward the root bridge (if there were two, there would be a bridging loop in the network). All bridges connected to a given segment listen to each other's BPDUs and agree on the bridge sending the best BPDU as the designated bridge for the segment. The corresponding port on that bridge is designated.

Alternate and Backup Port Roles

- These two port roles correspond to the `blocking` state of 802.1d. A blocked port is defined as not being the designated or root port. A blocked port receives a more useful BPDU than the one it would send out on its segment. Remember that a port absolutely needs to receive BPDUs in order to stay blocked. RSTP introduces these two roles for this purpose.

- An alternate port is a port blocked by receiving more useful BPDUs from another bridge, as shown in the following diagram:



- A backup port is a port blocked by receiving more useful BPDUs from the same bridge it is on, as shown in the following diagram:

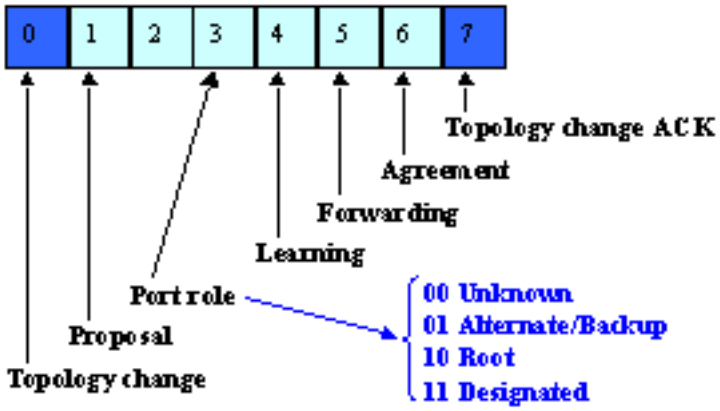
This distinction was already made internally within 802.1d. This is essentially how Cisco's UplinkFast functions. The rationale behind this is that an alternate port provides an alternate path to the root bridge and could therefore replace the root port should it fail. Of course, a backup port provides redundant connectivity to the same segment and cannot guarantee an alternate connectivity to the root bridge. It was therefore excluded from the uplink group.

As a result, RSTP calculates the final topology for the spanning tree using exactly the same criteria as 802.1d. There is absolutely no change in the way the different bridge and port priorities are used. The name `blocking` is used for the discarding state in Cisco implementation. CatOS release 7.1 and later still display the `listening` and `learning` states, giving even more information about a port than the IEEE standard requires. However, the new feature is that there now is a difference between the role the protocol has determined for a port and its current state. For example, it is now perfectly valid for a port to be designated and `blocking` at the same time. While this will typically happen for very short periods of time, it simply means that this port is in a transitory state towards designated `forwarding`.

New BPDU Format

few changes have been introduced by RSTP to the BPDU format. Only two flags, Topology Change (TC) and TC Acknowledgment (TCA), were defined in 802.1d, however RSTP now uses all six remaining bits of the flag byte in order to do the following:

- Encode the role and state of the port originating the BPDU
- Handle the proposal/agreement mechanism



Another important change is that the RSTP BPDU is now of type 2, version 2. The implication of this is that legacy bridges must drop this new BPDU. This property makes it easy for a 802.1w bridge to detect legacy bridges connected to it.

New BPDU Handling

BPDU are Sent Every hello-time

BPDU are sent every hello-time, and not simply relayed anymore. With 802.1d, a non-root bridge would only generate BPDUs when it received one on its root port. In fact, a bridge was rather relaying BPDUs more than actually generating them. This is not the case anymore with 802.1w. A bridge now sends a BPDU with its current information every <hello-time> seconds (2 by default), even if it does not receive any from the root bridge.

Faster Aging of Information

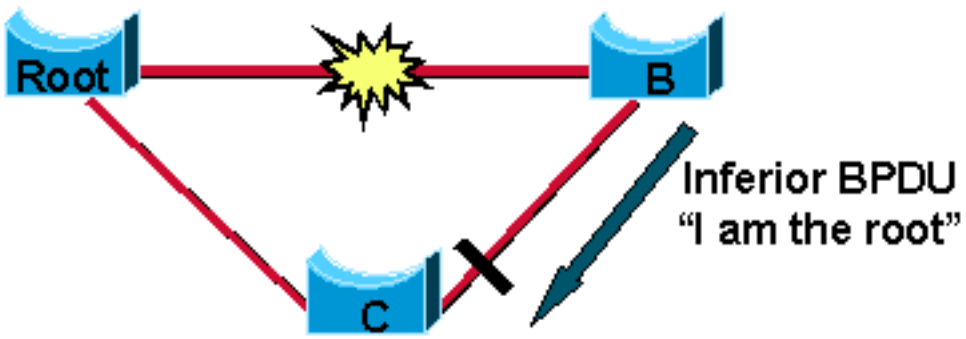
On a given port, if hellos are not received for three consecutive times, protocol information can be immediately aged out (or if max_age expires). Because of the previously mentioned protocol modification, BPDUs are now used as a keep-alive mechanism between bridges. A bridge considers that it has lost connectivity to its direct neighboring root or designated bridge if it misses three BPDUs in a row. This fast aging of the information allows quick failure detection. If a bridge fails to receive BPDUs

from a neighbor, it is certain that the connection to that neighbor has been lost, as opposed to 802.1d where the problem could have been anywhere on the path to the root.

Note: Failures are detected even much faster in case of physical link failures.

Accepting Inferior BPDUs

This concept is what makes up the core of the BackboneFast engine. The IEEE 802.1w committee decided to incorporate a similar mechanism into RSTP. When a bridge receives inferior information from its designated or root bridge, it immediately accepts it and replaces the one previously stored.



Because Bridge C still knows the root is alive and well, it immediately sends a BPDU to Bridge B containing information about the root bridge. As a result, Bridge B stops sending its own BPDUs and accepts the port leading to Bridge C as its new root port.

Rapid Transition to Forwarding State

Rapid transition is the most important feature introduced by 802.1w. The legacy STA was passively waiting for the network to converge before turning a port into the `forwarding` state. Achieving faster convergence was a matter of tuning the conservative default parameters (forward delay and max_age timers), often putting the stability of the network at stake. The new rapid STP is able to actively confirm that a port can safely transition to `forwarding` without relying on any timer configuration. There is now a real feedback mechanism that takes place between RSTP-compliant bridges. In order to achieve fast convergence on a port, the protocol relies upon two new variables: edge ports and link type.

Edge Ports

The edge port concept is already well known to Cisco's spanning tree users as it basically corresponds to the PortFast feature. All ports directly connected to end stations cannot create bridging loops in the network and can thus directly transition to `forwarding`, skipping the `listening` and `learning` stages. Neither edge ports nor PortFast enabled ports generate topology changes when the link toggles.

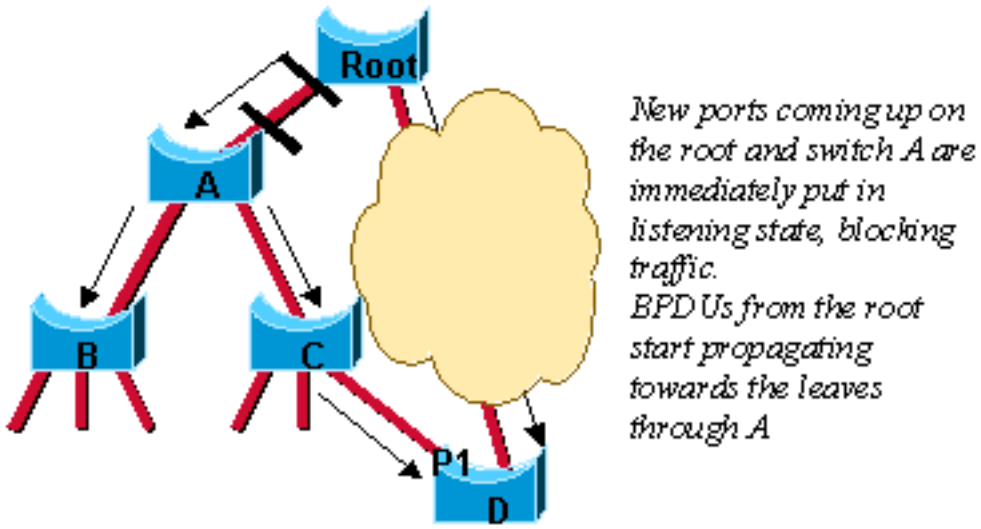
Unlike PortFast, an edge port that receives a BPDU immediately loses its edge port status and becomes a normal spanning tree port. At this point, there is a user-configured value and an operational value for the edge port state. Cisco's implementation maintains the *PortFast* keyword be used for edge port configuration, thus making the transition to RSTP simpler.

Link Type

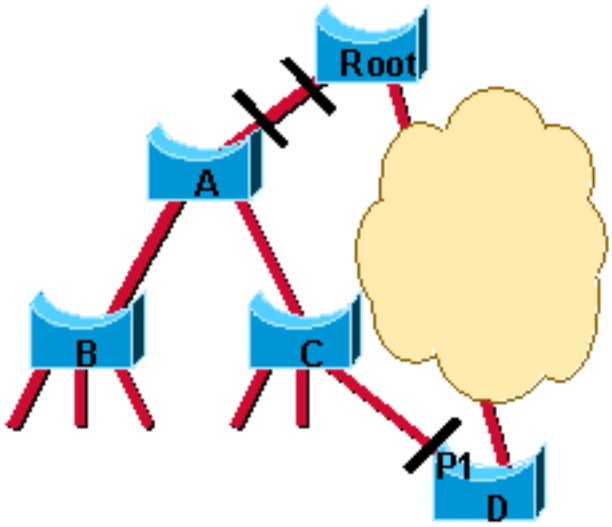
RSTP can only achieve rapid transition to `forwarding` on edge ports and on point-to-point links. The link type is automatically derived from the duplex mode of a port. A port operating in full-duplex will be assumed to be point-to-point, while a half-duplex port will be considered as a shared port by default. This automatic link type setting can be overridden by explicit configuration. In today's switched networks, most links are operating in full-duplex mode and are therefore treated as point-to-point links by RSTP. This makes them candidates for rapid transition to `forwarding`.

Convergence with 802.1d

The following diagram illustrates the way 802.1d deals with a new link being added to a bridged network:



In this scenario, a link between the root bridge and Bridge A has just been added. Suppose there was already an indirect connection between Bridge A and the root bridge (via C - D in the diagram). The STA will disable the bridging loop by blocking a port. First, as they are just coming up, both ports on the link between the root and A are put in `listening` state. Bridge A is now able to hear the root directly, and it immediately propagates its BPDUs on its designated ports, toward the leaves of the tree. As soon as B and C receive this new superior information from A, they immediately relay it toward the leaves. In a few seconds, Bridge D has received a BPDU from the root and instantly blocks its port P1.



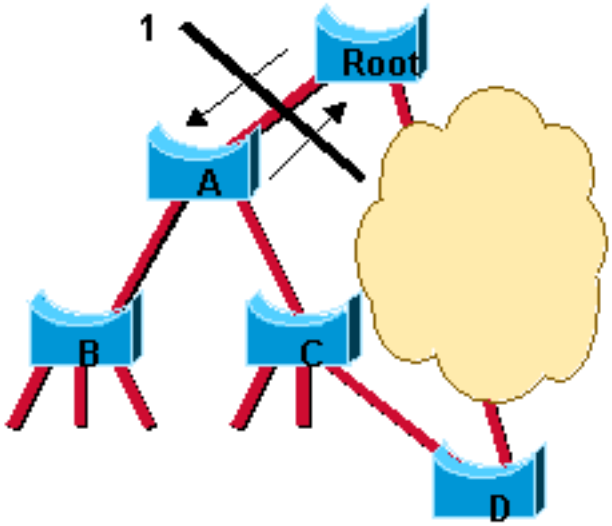
Very quickly, the BPDUs from the root reach D that immediately blocks its port P1. The topology has now converged, though, the network is disrupted for twice forward_delay.

Spanning tree has been very efficient in calculating the new topology of the network. The only problem now is that twice the forward delay has to elapse before the link between the root and A eventually ends up in the `forwarding` state. This means 30 seconds of disruption of traffic (the entire A, B, and C part of the network is isolated) because the 8021.D algorithm lacks a feedback mechanism clearly advertising that the network has converged in a matter of seconds.

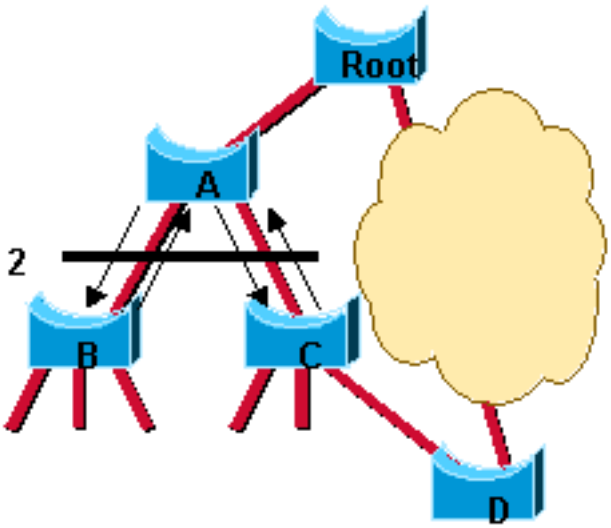
Convergence with 802.1w

Now, we see how RSTP deals with a similar situation. Remember that the final topology is exactly the same as the one calculated by 802.1d (that is, one blocked port at the same place as before), only the steps used to reach this topology have changed.

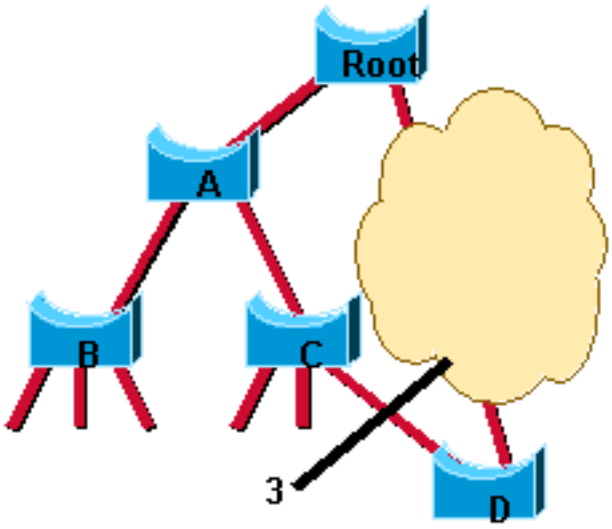
Both ports on the link between A and the root are put in designated blocking as soon as they come up. So far, everything behaves as in a pure 802.1d environment. However, at this stage, a negotiation takes place between Switch A and the root. As soon as A receives the root's BPDU, it blocks its non-edge designated ports. This operation is called sync. Once this is done, Bridge A explicitly authorizes the root bridge to put its port in `forwarding`. The diagram below illustrates the result of this process on the network. The link between switch A and the root bridge is blocked, and both bridges exchange BPDUs.



Once Switch A has blocked its non-edge designated ports, the link between Switch A and the root is put in `forwarding` state and we reach the situation:



There still can not be a loop because instead of blocking *above* Switch A, the network is now blocking below Switch A. The potential bridging loop is cut, however at a different location. This cut is traveling down the tree along with the new BPDUs originated by the root through Switch A. At that stage, the newly blocked ports on Switch A will also negotiate a quick transition to `forwarding` with their neighboring ports on Switch B and Switch C that both initiate a sync operation. Switch B only has edge designated ports (other than its root port towards A), so it has no port to block in order to authorize A to go to `forwarding`. Similarly, Switch C only has to block its designated port to D. The state shown in the diagram below has now been reached.

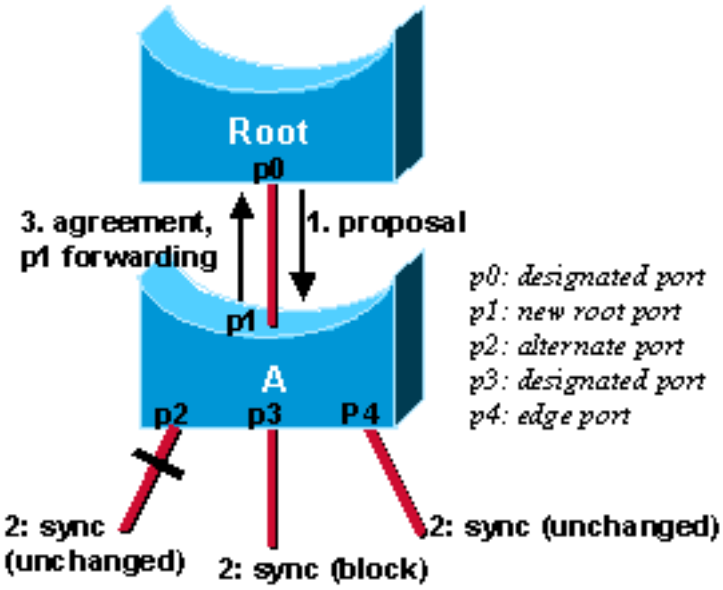


Remember that the final topology is exactly the same as for the 802.1d example, which means that port P1 on D ends up blocking. This means that we have reached the final network topology, just in the time necessary for the new BPDUs to travel down the tree. No timer has been involved in this quick convergence. The only new mechanism introduced by RSTP is the acknowledgment that a switch can send on its new root port in order to authorize immediate transition to `forwarding`, bypassing the twice-the-forward-delay long `listening` and `learning` stages. To benefit from fast convergence, the administrator only needs to remember the following:

- This negotiation between bridges is only possible when bridges are connected by point-to-point links (that is, full-duplex links unless explicit port configuration).
- Edge ports now play an even more important role that PortFast is enabled on ports in 802.1d. If the network administrator fails to properly configure the edge ports on B for instance, their connectivity would have been impacted by the link between A and the root coming up.

Proposal/Agreement Sequence

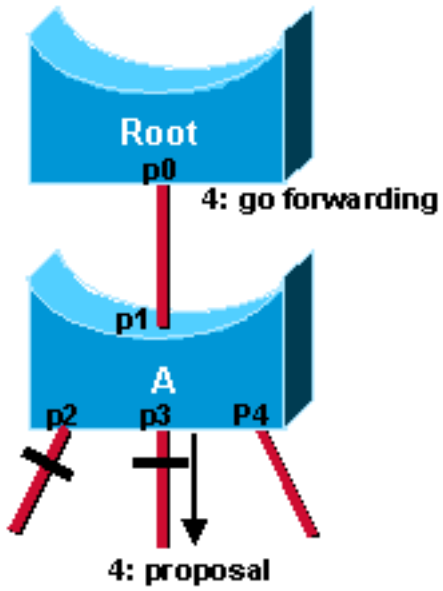
When a port has been selected by the STA to become a designated port, 802.1d still waits twice <forward delay> seconds (2x15 by default) before transitioning it to the `forwarding` state. In RSTP, this condition corresponds to a port with a designated role but a blocking state. The diagrams below illustrate how fast transition is achieved step-by-step. Suppose a new link is created between the root and Switch A. Both ports on this link are put in a designated `blocking` state until they receive a BPDU from their counterpart.



When a designated port is in a `discarding` or `learning` state (and only in this case), it sets the proposal bit on the BPDUs it sends out. This is what happens for port p0 of the root bridge, as shown in Step 1 of the diagram above. Because Switch A receives superior information, it immediately knows that p1 is going to be its new root port. Switch A then starts a sync to ensure that all of its ports are in-sync with this new information. A port is in-sync if it meets either of the following criteria:

- The port is in `blocking` state (which means discarding, in a stable topology).
- The port is an edge port.

In order to illustrate the effect of the sync mechanism on different kind of ports, suppose there exists an alternate port p2, a designated forwarding port p3, and an edge port p4 on Switch A. Notice that p2 and p4 already meet one of the criteria listed above. In order to be in sync (Step 2 of the diagram above), Switch A just needs to block port p3, assigning it the discarding state. Now that all of its ports are in sync, Switch A can now unblock its newly selected root port p1 and reply to the root by sending an agreement message (Step 3). This message is a copy of the proposal BPDU, with the agreement bit set instead of the proposal bit. This ensures that port p0 knows exactly to which proposal the agreement it receives corresponds to.

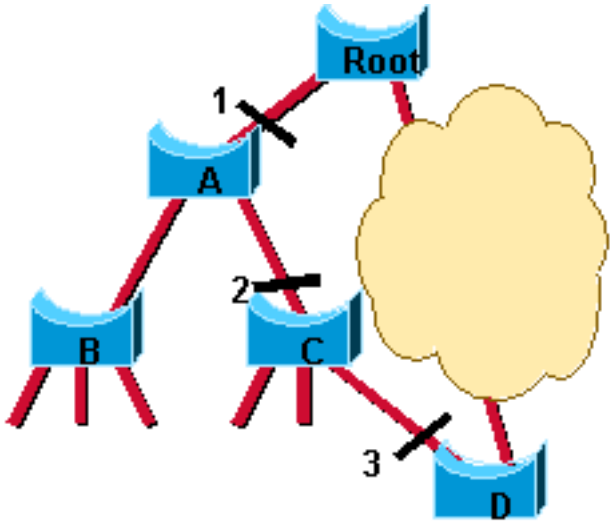


Once p0 receives that agreement, it can immediately transition to `forwarding`. This is Step 4 of the figure above. Notice that port p3 was left in a designated discarding state after the sync. In Step 4, that port is in the exact same situation as was port p0 during Step 1. It then starts proposing to its neighbor, attempting to quickly transition to `forwarding`.

- The proposal agreement mechanism is very fast, as it does not rely on any timers. This wave of handshakes propagates quickly towards the edge of the network, and quickly restores connectivity after a change in the topology.

- If a designated discarding port does not receive an agreement after having sent a proposal, it slowly transitions to the `forwarding` state, falling back to the traditional 802.1d `listening-learning` sequence. This could happen for instance if the remote bridge doesn't understand RSTP BPDUs, or if the remote bridge's port is blocking.

- Cisco introduced an enhancement to the sync mechanism that allows a bridge to put only its former root port in the discarding state when syncing. Detailing the way this mechanism works is beyond the scope of this document. However, one can safely assume that it will be invoked in most common reconvergence cases. The scenario described in the Convergence with 802.1w section of this document now becomes extremely efficient, as only the ports on the path to the final blocked port are temporarily confused.



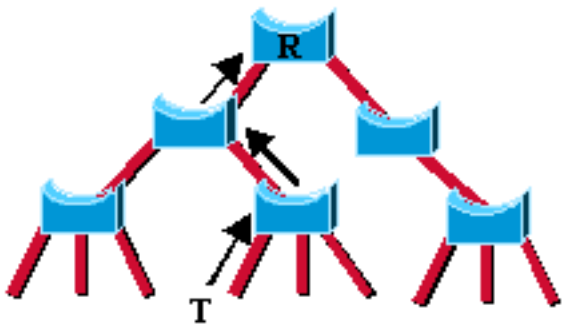
UplinkFast

Another form of immediate transition to the `forwarding` state included in RSTP is exactly similar to Cisco's UplinkFast proprietary spanning tree extension. Basically, when a bridge loses its root port, it is able to put its best alternate port directly into `forwarding` mode (the appearance of a new root port is also handled by RSTP). The selection of an alternate port as the new root port generates a topology change. The 802.1w topology change mechanism clears the appropriate entries in the upstream bridge's Content Addressable Memory (CAM) tables, removing the need for the dummy multicast generation process of UplinkFast.

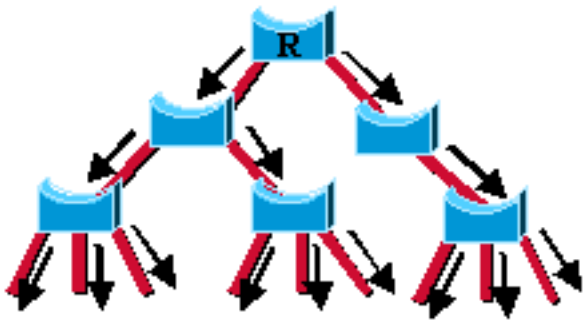
UplinkFast does not need to be configured further, as the mechanism is natively included and automatically enabled in RSTP.

New Topology Change Mechanisms

When an 802.1d bridge detects a topology change, it first notifies the root bridge, using a reliable mechanism, as shown in the diagram below:



A topology change is generated on point T.
1st step: A TCN is going up to the root.



2nd step: the root advertises the TC for max-age+ forward delay.

Once the root bridge is aware of a change in the topology of the network, it sets the TC flag on the BPDUs it sends out, which are then relayed to all the bridges in the network. When a bridge receives a BPDU with the TC flag bit set, it reduces its bridging-table aging time to forward delay seconds, ensuring a relatively quick flushing of stale information. Refer to Understanding Spanning-Tree Protocol Topology Changes for more information on this process. This topology change mechanism has been deeply remodeled in RSTP. Both the detection of a topology change and its propagation through the network have evolved.

Topology Change Detection

In RSTP, only non-edge ports moving to the `forwarding` state cause a topology change. This means that a loss of connectivity is not considered as a topology change any more, contrarily to 802.1d (that is, a port moving to blocking does no longer generates a TC). When a RSTP bridge detects a topology change, the following happens:

- It starts the TC While timer with a value equal to twice the hello time for all its non-edge designated ports and its root port if necessary.
- It flushes the MAC addresses associated with all these ports.

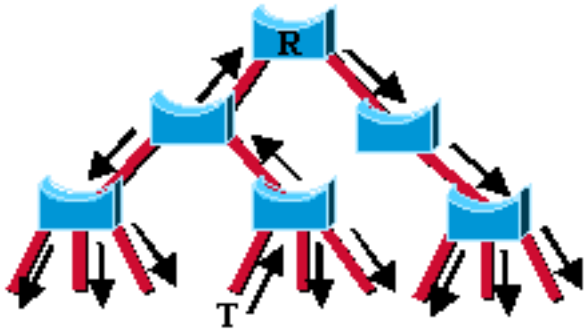
Note: As long as the TC While timer is running on a port, the BPDUs sent out of that port have the TC bit set. BPDUs are also sent on the root port while the timer is active.

Topology Change Propagation

When a bridge receives a BPDU with the TC bit set from a neighbor, the following happens:

- It clears the MAC addresses learnt on all its ports except the one that received the topology change.
- It starts the TC While timer and sends BPDUs with TC set on all its designated ports and root port (RSTP no longer uses the specific TCN BPDU, unless a legacy bridge needs to be notified).

This way, the TCN is flooded very quickly across the whole network. The TC propagation is now a one step process. In fact, the initiator of the topology change is flooding this information throughout the network (as opposed to 802.1d where only the root could do so). This mechanism is much faster than the 802.1d equivalent. There is no need to wait for the root bridge to be notified and then maintain the topology change state for the whole network for <max age plus forward delay> seconds.



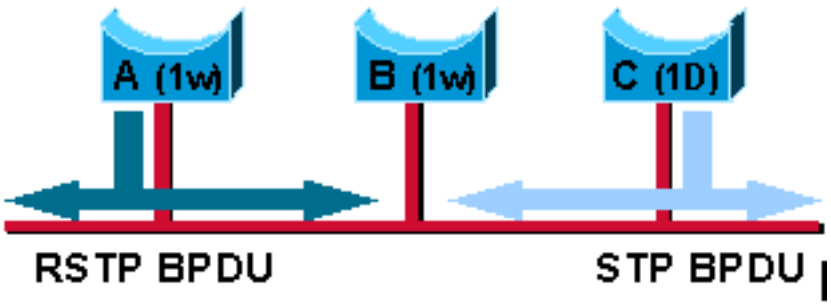
The originator of the TC directly floods this information through the network

In just a few seconds (a small multiple of hello times), most of the entries in the CAM tables of the entire network (VLAN) are flushed. This approach results in potentially more temporary flooding, but on the other hand it clears potential stale information that prevents rapid connectivity restitution.

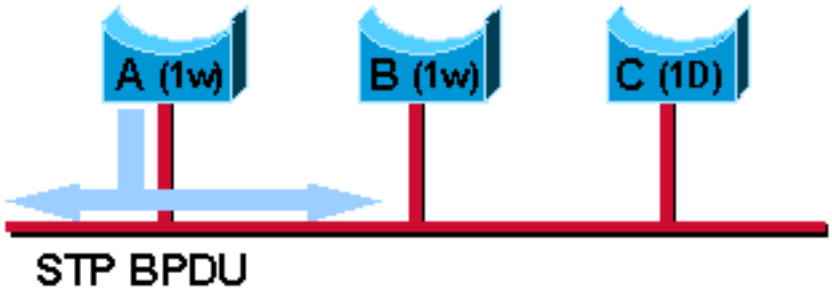
Compatibility with 802.1d

RSTP is able to interoperate with legacy STP protocols. However, it is important to note that 802.1w's inherent fast convergence benefits are lost when interacting with legacy bridges.

Each port maintains a variable defining the protocol to run on the corresponding segment. A migration delay timer of three seconds is also started when the port comes up. When this timer is running, the current (STP or RSTP) mode associated to the port is locked. As soon as the migration delay has expired, the port will adapt to the mode corresponding to the next BPDU it receives. If the port changes its operating mode as a result of receiving a BPDU, the migration delay is restarted, limiting the possible mode change frequency.



For instance, suppose Bridges A and B in the figure above are both running RSTP, with Switch A being designated for the segment. A legacy STP Bridge C is introduced on this link. As 802.1d bridges ignore RSTP BPDUs and drop them, C believes there are no other bridges on the segment and starts sending its inferior 802.1d-format BPDUs. Switch A receives these BPDUs and, after twice hello time seconds maximum, changes its mode to 802.1d on that port only. As a result, C can now understand A's BPDUs and accepts A as the designated bridge for that segment.



Notice that in this particular case, if Bridge C was removed, Bridge A would keep running in STP mode on that port even though it is able to work more efficiently in RSTP mode with its unique neighbor B. That is because A has no way of knowing that Bridge C got removed from the segment. For this particular (rare) case, user intervention is required in order to restart the port's protocol detection manually.

When a port is in 802.1d compatibility mode, it is also able to handle topology change notification (TCN) BPDUs, and BPDUs with TC or TCA bit set. Conclusion

Conclusion

RSTP (IEEE 802.1w) natively includes most of Cisco's proprietary enhancements to the 802.1d spanning tree such as BackboneFast, UplinkFast, and PortFast. RSTP can achieve much faster convergence in a properly configured network, sometimes in the order of a few hundred milliseconds. Classic 802.1d timers such as forward delay and max_age are nearly only used as a backup and should not be necessary if point-to-point links and edge ports are properly identified set by the administrator, and if there is no interaction with legacy bridges.

Related Information

- **Understanding and Configuring the Cisco Uplink Fast Feature**
- **Tools and Utilities**
- **Technical Support - Cisco Systems**

Statement.