802.1AS Fast Master Clock Selection

Moving 802.1AS closer to RSTP

Version 1

Norman Finn

Cisco Systems

Introduction

Introduction

- IEEE 1588 and the current draft of P802.1AS both have Announce Messages that:
  - Elect the clock that will drive the network's timing;
  - Propagate over an underlying data transport network, a spanning tree in the case of Layer 2; and
  - Introduce timeouts to let the master election settle.
- The result is that the convergence of the master clock election and clock distribution tree formation:
  - Depends on a pre-existing data forwarding topology;
  - Is considerably slower than the alternative presented in this slide deck; and
  - Is more complex than the alternative presented, here.

Rapid Spanning Tree Protocol Basics

Rapid Spanning Tree Protocol (RSTP) Basics

- Networks consist of Bridges that have Ports attaching them to LANs.
  - A LAN can be attached to two Ports (point-to-point medium) or more than two Ports (shared medium).
- Each Bridge can transmit Bridge Protocol Data Units (BPDUs) on Ports. A BPDU says, “This is the state of RSTP on this Port.”
  - That state is different on each Port.
- Every Bridge considers itself either the Root Bridge or not the Root Bridge.

Applying RSTP to 802.1AS

- The most tricky bits of RSTP are concerned with ensuring that the data plane, which can operate independently of the control plane, never forwards frames in a closed loop, barring malfunctions that cripple the algorithm.

  The tricky bits also make RSTP “rapid” compared to the old STP.

  Those tricky bits are what cause RSTP to fail catastrophically when algorithm malfunctions do occur.

- But, 802.1AS has no such independent data plane, so those tricky bits are not needed for clock distribution.

- Therefore, these slides present only the (simpler) bits that are needed by 802.1AS.

RSTP Port Roles

- Every Port on a Bridge takes one of four roles:

  The **Root Port**;

  An **Alternate Port**;

  A **Designated Port**; or

  A **Backup Port**.

Port Roles: Root Port

- The one Port closest to the Root Bridge. This Bridge:

  Expects to receive a regular stream of BPDUs (Sync/Followup/Announce) on this Port from the Bridge closer to the Root Bridge.

  Expects to receive a regular stream of Pdelay_Reqs from the Designated Bridge, and to respond to them.

  Will modify and propagate the information received in these BPDUs (Sync/Followup/Announce) messages to the rest of the network through this Bridge’s Designated Ports.

  The Root Bridge has no Root Port. Each Non-Root Bridge has exactly one Root Port.

Port Roles: Alternate Port

- Any port that is connected to a Bridge that is closer to the Root than this Bridge, but is not the Root Port. This Bridge:

  Expects to receive a regular stream of BPDUs (Sync/Followup/Announce) on this Alternate Port from the Designated Bridge.

  Expects to receive a regular stream of Pdelay_Reqs from the Designated Bridge, and to respond to them.

  Will **not** propagate the information received in the BPDUs (Sync/Followup/Announce).

  Can instantly transform the best of the Alternate Ports into the Root Port, if the Root Port fails.

Port Roles: Designated Port

- A Designated Port: A port that has the best claim, via BPDUs (Sync/Followup/Announce), to being the closest Port to the Root Bridge among all of the Ports connected to the same LAN as the Designated Port.
  - A Port that is not connected to another Bridge is always a Designated Port.
  - All of the other Bridges' Ports connected to that same LAN are either Root Ports or Alternate Ports.
  - Every Port on the Root Bridge is either a Designated Port or a Backup Port (see next slide).
- On a Designated Port, this Bridge will:
  - Transmit a regular stream of BPDUs (Sync/Followup/Announce) that propagate timing information to the rest of the network.
  - Transmit a regular stream of Pdelay_Reqs and expect responses.

Port Roles: Backup Port

- A Port on a Bridge that is connected to a LAN that is connected to a Designated Port on the same Bridge. This Bridge:

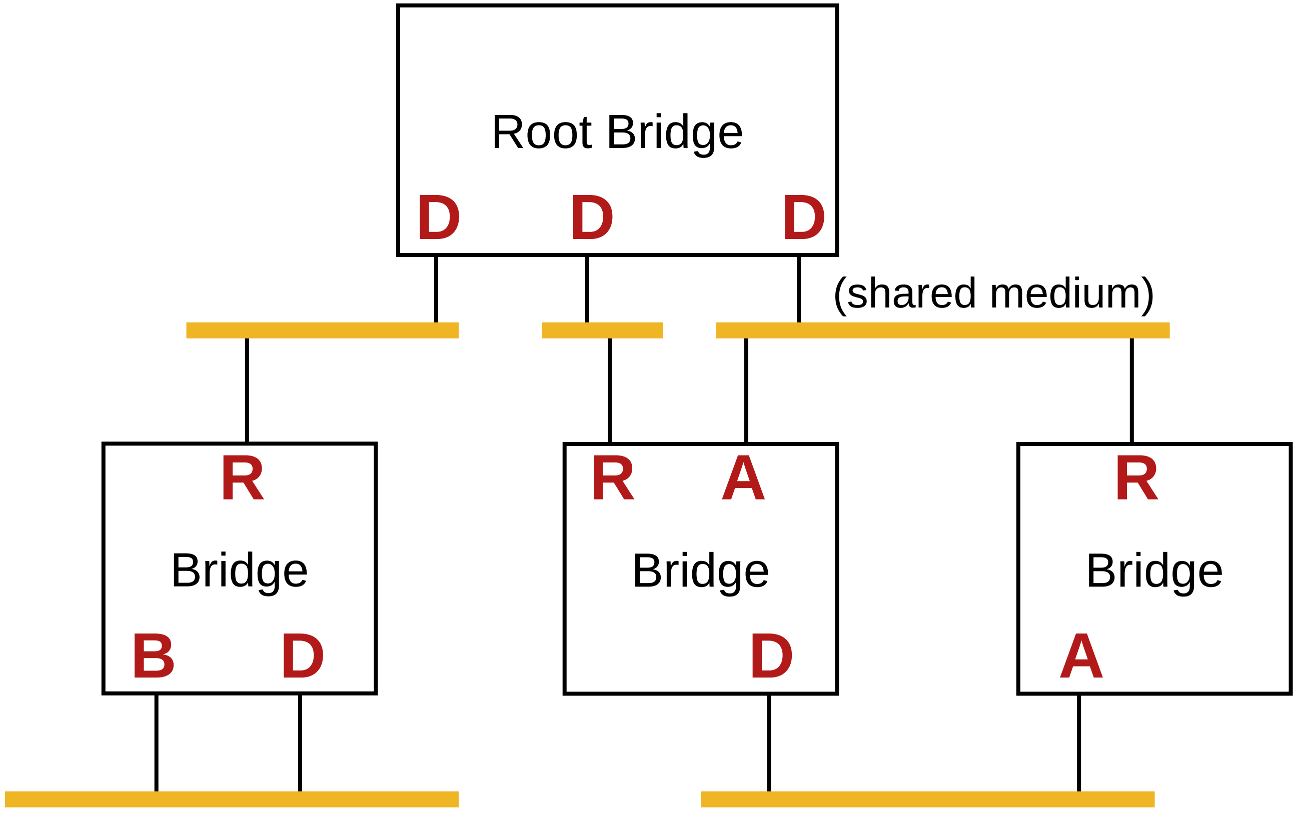
  Expects to receive a regular stream of BPDUs (Sync/Followup/Announce) on this Port from the Designated Bridge. (Could the Syncs be deleted? They're not needed.)

  Expects to receive a regular stream of Pdelay_Reqs from the Designated Bridge, and to respond to them. (These aren't needed, either.)

  Will **not** propagate the information received in the BPDUs (Sync/Followup/Announce).

  Every Port on the Root Bridge is either a Designated Port or a Backup Port (see next slide).

Port Roles



- **R**oot Port, **D**esignated Port, **A**lternate Port, **B**ackup Port

RSTP BPDU: Comparing BPDUs

- **On each LAN, there is (eventually) exactly one Designated Port.**

  The Bridge to which that Port belongs is the Designated Bridge for this LAN.

  Only the LAN’s Designated Bridge transmits BPDUs (Sync/Followup/Announce); the other Bridge(s) just listen.

  When a LAN comes up, every Bridge attached to it assumes that it is the Designated Bridge.

  The first Bridge to transmit a BPDU (Sync/Followup/Announce) either really is the Designated Bridge, or the other Bridge(s) will respond with their BPDUs.

  Very quickly, all agree on which is the Designated Bridge.

  (The algorithm works even if agreement is delayed.)

RSTP BPDUs: What does "best" mean?

- To compare claims, the fields in a BPDU (or variables in memory) are concatenated into a "priority vector" that is treated as a very long binary number.

  The smallest numerical value wins.

  Since all priority vectors are the same length and are unsigned, there is no difference between lexical and numerical comparisons.

- Different combinations of fields, and thus different vectors, are used for different computations.

RSTP BPDUs: The four main fields

- **Root ID:** The globally unique ID of the Bridge that this Bridge thinks is the Root Bridge.
- **Root Path Cost:** The total cost from this Bridge to the Root Bridge, where the cost of each hop is inversely proportional to the link speed.
- **Bridge ID:** The globally unique ID of this Bridge. (Same as Root ID in a BPDU sent by the Root Bridge.)
- **Port ID:** Uniquely identifies the Port on which the BPDU was sent among all Ports with the same Bridge ID.

RSTP: When I receive a BPDU ...

- **Step 1: New Root Bridge?**

Compare received:	To my:
Root ID	Root ID from Root Port (my Bridge ID if I think I’m Root)

- If received Root ID is better, replace my information with his.

  This is now the Root Port.

  Update all of the Ports’ information.

RSTP: When I receive a BPDU ...

- **Step 2: Who is the Designated Bridge on this LAN?**

Signifi-cance	Compare received:	To my:
most	Root ID	Root ID from Root Port (my Bridge ID if I think I’m Root)
...	Root Path Cost	Root Port’s Root Path Cost (0 if I think I’m Root)
...	Bridge ID	My Bridge ID
least	Port ID	Port ID of this Port

- If I’m Designated, ignore his information.

RSTP: When I receive a BPDU ...

- **Designated Bridge determination:**

Root ID	Whose information is derived from the best Root (best clock)? If equal ...
Root Path Cost	Who is closest to that Root? If equal ...
Bridge ID	Who is configured best (high bytes of ID) or has the lowest address? If equal ...
Port ID	(I'm listening to myself.) Which port has the lowest priority or lowest address?

- If I win, this is a Designated Port.

RSTP: When I receive a BPDU ...

- **Step 3: Which is the Root Port? Compare all non-Designated Ports':**

BPDU Root ID	Which Port's information is derived from the best Root (best clock)? If equal ...
BPDU Root Path Cost + this Port's Cost	Which is closest to that Root, including the receiving Costs? If equal ...
BPDU Bridge ID	Which Designated Bridge is configured best (high bytes of ID) or has the lowest address? If equal ...
BPDU Port ID	(Two Ports are listening to the same other Bridge.) Which port has the lowest priority or lowest address?

No data plane. What RSTP fields matter?

Protocol Identifier (2)	Bridge Identifier (8)
Protocol Version Identifier (1)	Port Identifier (2)
BPDU Type (1)	Message Age (2)
Flags (1)	Max Age (2)
Root Identifier (8)	Hello Time (2)
Root Path Cost (4)	Forward Delay (2)

- Ignore the dull bits and data plane interlock bits.
- We’ve seen the green fields.
- What are these fields?

RSTP BPDU: What fields are left?

- **Message Age:** How many hops has this information made since leaving the Root Bridge?
- **Max Age:** After how many hops should this information be discarded?
- **Hello Time:** How often does the Designated Port send BPDUs?

RSTP BPDU: Message Age and Max Age

- The Root Bridge’s configured Max Age is spread throughout the network via the BPDUs, and determines how many hops the information can travel. Message Age is incremented at each hop.
- The **good news** is that this is a simple technique that ensures the algorithm will converge. Unless ...
- The **bad news** is that, if the network is larger than Max Age hops, the network will not converge.
- When this happens, the outlying areas may not be connected to each other, and each island will use the best Root (Master Clock) it can find.

RSTP BPDU: Hello Time

- Each Designated Bridge announces the rate at which it intends to transmit BPDUs.
- If the receiving Bridge misses three BPDUs, it figures that the Designated Bridge is dead, and proceeds as if it never received a BPDU on that Port.
  - If not the Root Port, that's easy.
  - If it's the Root Port, then another Root Port must be selected from among the Alternate Ports. If there are no Alternate Ports, then this Bridge becomes the Root Bridge (until corrected).

RSTP+802.1AS: An alternative to hop count

- One could also record a chain of Bridge IDs from the Root Bridge, with each Bridge adding its own Bridge ID to the list, and discarding any BPDU containing its own Bridge ID.
- The **good news** is that this is a simple technique that ensures the algorithm will converge. Unless ...
- The **bad news** is that the frame grows and grows, and if it fills up, the algorithm will not fail.
- When this happens, the outlying areas may not be connected to each other, and each will use the best Root (clock) it can find.

RSTP+802.1AS: Getting Started

- A Bridge simply considers itself to be the Root Bridge, and starts running.
- If it’s wrong, its neighbors will quickly inform it.
- Every Bridge has a clock and can be Root Bridge!

  It might be a lousy clock – an integer count of times through the scheduler – but, it’s got a clock.
- There is no such thing as a Station!

  There is only a Bridge that happens to have just one Port.

  That one Port is either a Designated Port (I’m the Master Clock) or it’s the Root Port (I’m not the Master Clock).

RSTP+802.1AS: Time until convergence?

- **This scheme reacts as fast as the information can propagate. There are no timers!**
- Every link has a Designated Bridge, and that Bridge runs the Pdelay measurements on that link, whether the Master Clock (BPDU) information is propagated (the other end of the link is a Root Port) or not (the other end is an Alternate Port).
- We do not expect that the measured link delay will be significantly different, depending on which end of the link is driving the Pdelay measurements, so topology changes that reverse the Designated – Root/Alternate roles of a link will have a head start on propagating the time properly.

RSTP+802.1AS: Time until convergence?

- **This scheme reacts as fast as the information can propagate. There are no timers!**
- Having said that, we may introduce a safety timer that limits Announce messages, just in case a bad implementation prevents convergence.

Merging 1588 and RSTP
The NewAnnounce Message

NewAnnounce fields

domainNumber	Same as 1588 – ignore any NewAnnounce not matching.
Master Clock Identifier	From 1588, but works like Root Identifier
Master Clock Path Cost	New. Makes better comparisons of different paths than hop count.
Clock Identifier	From 1588, but works like Bridge Identifier
Port Identifier	1588 portIdentity is a subset of Port Identifier
Message Age, Max Age	From RSTP
Hello Time	Discuss. Hello Time, logMean-MessageInterval, or both?
Clock Information	As defined for 1588

NewAnnounce Fields

- **domainNumber:** Identifies the reach of this protocol. Not in RSTP.

  Bridge ignores NewAnnounce messages from a different domain.

  This could be expanded to a name + version number, like MSTP.

- **Master Clock Identifier:** Same function as RSTP Root Identifier. Consists of:

  1 byte (or less) of priority1: Just like the high bytes of an RSTP Bridge ID. This supports absolutely forcing the Master Clock selection.

  1 bytes (or less) of clock quality: In the absence of priority1 configuration, this causes the best clock to be selected as Master Clock.

  1 byte (or less) of priority2: This supports forcing a Master Clock selection among equal quality clocks.

  2 bytes (or more) of virtual entity number: Just like the high bytes of an RSTP Bridge ID. This provides 4k or more IDs for virtual devices from a single physical address.

  8 bytes of MAC address: 1588 supports EUI-64 addresses, so this protocol should, also.

NewAnnounce Fields

- **Master Clock Path Cost:** Same function as RSTP Root Path Cost.

  Allows comparison of two paths to the Master Clock, so that the path that introduces the least inaccuracies can be chosen.

  In RSTP, each Port has its own cost. The Port Cost is normally inversely proportional to the speed of the link, but it can be overridden to any value by the management.

  In RSTP, the configured Port Cost is added to the Root/Alternate Port's Root Path Cost, and that value is distributed to all other Bridges. Thus, only the BPDU receiver adds Port Cost.

  For 802.1AS, each end of the link contributes some inaccuracy, perhaps due to implementation choices. Therefore, the Designated Port adds the Root-to-Designated-across-the-Bridge Cost and the Transmit Cost, and the Root/Alternate Port adds the Receive Cost.

  If we believe that the probable error after n hops is the sum of the n hops' errors, that's fine. We might believe instead that the probable error is the root-mean-square of the hop errors, so the Cost parameter would be a sum of squares of the individual Costs. (There is no need to take the square root to compare relative Costs.)

NewAnnounce Fields

- **Clock Identifier:** Same function as RSTP Bridge Identifier.

  Same format as Master Clock Identifier.

  If I’m the Master Clock, this becomes the Master Clock Identifier.

- **Port Identifier:** 1588 portIdentity is a subset of RSTP Port Identifier, works like RSTP Port Identifier

  1588 provides a 2-byte Port ID.

  RSTP provides a 4-bit configurable priority field, so that the Backup vs. Designated choice can be configured, followed by a 12-bit Port ID.

  The RSTP format is probably more useful, but this should be discussed.

NewAnnounce Fields

- **Message Age:** Same function as RSTP Message Age.

  It is now a simple hop count.

- **Max Age:** Same function as RSTP Max Age.

- **Clock ID List:** Possible improvement.

  The Message Age and Max Age could be augmented by including a list of Clock IDs traversed along the path taken by this Master Clock information.

  The whole Clock ID does not need to be stored; only the virtual entity number and the MAC address are required.

- Including all three of these fields allows absolute minimum convergence time, while bounding the frame size.

Combining message types.

Keep the Announce Message separate?

- We could keep the NewAnnounce Message separate from the other messages, as they exist in the current 802.1AS draft.

  Master Clock selection and propagation would still be more responsive than 1588.

  One could more easily argue that 802.1AS is derived properly from 1588, than if the NewAnnounce is combined with another message type, i.e., Pdelay_Req.

  Announce Messages can be dropped, so a Bridge must be ready to ignore other messages that are not expected, and we must ensure that the fields are in the messages to allow us to do that.

Combine the Announce Message with sync?

- We could combine the NewAnnounce Message with any or all Sync Messages from a Designated Port.

  This is one step further removed from 1588.

  This results in the receiver learning about the best Master Clock at the same moment it gets a Sync based on that clock.

  But, we know that the Syncs will often be handled by hardware. We don’t want to take a chance that someone cannot handle the additional information in a Sync.

Combine the Announce Message with Pdelay?

- We could combine the NewAnnounce Message with the Pdelay_Req and/or other Pdelay Messages.

  (Not Pdelay_Resp – the NewAnnounce isn't sent in that direction.)

  This is, again, one step further removed from 1588 than a separate NewAnnounce message.

  But, it results in the earliest notification of the software that can be done without using Sync, and eliminates the separate Announce message.

  At this moment, this is this writer's proposal. Further discussion is required, of course.

Do Pdelays happen too often?

- If necessary, we can reduce the amount of computation required for the spanning tree even further if we:

  Attach an “InfoRevision” field to each NewAnnounce message.

  InfoRevision is incremented each time the information sent in a given Port’s NewAnnounce message changes.

  The “spanning tree” process is awakened only when a received InfoRevision doesn’t match the last-receive value.

  InfoRevision is initialized to a random value each time a device reboots. (Not the same value each time!)

  InfoRevision is incremented, whether the information changed or not, every one or a few seconds, just in case the receiver rebooted, or in case the random value collided with the value in use before the reboot of the sender.

Summary

Summary

- Protocol simplification: Every node that participates in the protocol is an n-Port Clock/Bridge. n=1 for a station.
- Making the Announce message work like RSTP BPDUs makes both Master Clock selection and recovery from topology changes faster, because there are no more timeouts, and there is no underlying data transport topology that must converge, first.
- Tying a Master Clock Path Cost algorithm to clock propagation accuracy improves the clocks' accuracy.
- Combining the new Announce message with another message, probably Pdelay_Req, further simplifies and speeds up the protocol.
- Hopefully, this still fits under the 1588 umbrella.