Rapid Spanning Tree Migration

Mick Seaman

This note discusses options for plug and play migration from the original or 'legacy' spanning tree protocol to the enhanced or 'rapid' spanning tree

Introduction

The proposed Rapid Spanning Tree Protocol was explicitly designed to be compatible with the legacy protocol specified in IEEE Std. 802.1D-1998 and prior revisions. Computation of the spanning tree is identical. Protocol changes are in the following areas:

a) Inclusion of the Port Roles (Root Port, Designated Port, Backup Port) in the computation of Port State (Blocking, Listening, Learning, Forwarding[1]). In particular a new Root Port transitions to forwarding rapidly.

b) Signaling to neighboring bridges of a bridge Port's desire to be Designated and Forwarding, and explicit acknowledgment by the neighboring bridge on a point-to-point link. This allows the transition of the Port State to Forwarding to complete without waiting for a timer expiry.

c) Acceptance of messages from a prior Designated Bridge even if they conveyed "inferior' information. Additionally a minimum increment to the Message Age is specified so that messages propagating in this way cannot 'go round in circles' for long.

d) Improvements in the propagation of topology change information so that that information does not have to be propagated all the way to the Root Bridge and back before unwanted learnt source address information is purged from forwarding databases.

e) Origination of Configuration BPDUs on a port by port basis, instead of transmission on Designated Ports following reception of information from the Root.

In addition to state machines described in P802.1w/D2, the following are required to support these changes:

1. Respecification of timer values to accommodate changed behavior in the cases where neighboring bridges do not implement the rapid algorithm, and the Forward Delay timers do actually run to completion. The default timers are believed to work well, but some care may be necessary in environments where timers have been tuned to their minimum values.
2. Detection of point-to-point links to allow selection of the procedures to indicate 'Designated wanting to become Forwarding' (referred to as 'designated indication' or 'txmt_di' and 'rcvd_di' in the state machines) and 'Yes, go ahead' ('designated confirmation' or 'txmt_dc' and 'rcvd_dc').
3. Specification of message formats to include designated indications and confirmations.

This note addresses this last message format question specifically.

The Message Format Problem

The initial proposal was to include two additional flags in the flags field of Configuration BPDUs and to increment the Protocol Version Identifier field. The intent of the 802.1D specification was that additions could be made to the BPDU so long as each BPDU contained at minimum all the fields of prior versions, and implementations conformant to any given version of the protocol effectively ignored the version identifier field for subsequent versions, processing them only according to their knowledge of the protocol. That is:

- a version 0 bridge would completely ignore the version identifier field and process all BPDUs as if they were version 0 BPDUs, ignoring additional fields
- a version 1 bridge would process version 0 BPDUs as version 0 BPDUs, version 1 BPDUs as version 1 BPDUs, and version 2 and higher BPDUs as version 1 BPDUs, igoring additional fields.

Unfortunately it appears that a conformance test house has widely disseminated a tool that checks that BPDUs of unknown version and unknown flags are discarded – thus ensuring an installed base of bridges that can not simply be resident in upgraded networks as originally envisioned.

The problem then, is to choose suitable message formats and possibly accompanying procedures to select a BPDU format on a specific link that provides forward migration with the minimum of fuss.

[1] Disabled for ports not participating in the algorithm

Message Format Options

Our options in the face of the problem outlined above include the following:

1. Retain the original suggested approach of adding flags and upgrading the version identifier.
2. Revise the format as suggested, but expect the new messages to be dropped by previous revision bridges.
3. For each network declare a ‘flag day’, before that date all bridges use the old format BPDUs, after they all use the new format.
4. Revise the BPDU format, to a format that will definitely be ignored by all existing bridges, and for some indefinite transition period add procedures that verify that all bridges on a given LAN (the bridges at both ends of a link in the point-to-point link case) understand the new message formats, reverting to the old message formats if they do not.
5. Have new bridges send all messages twice, once in the old format and once in the new format.

(1) should get people to sit up and take notice of the problem with the installed base and validation tool, unfortunately it may cause significant delay in adopting the new algorithm and enjoying its benefits.

(2) is only part of some other solution, and a risky one at that.

(3) is very difficult to deploy, at least last time it was tried most networks had bridges from a single vendor, the problem is much more difficult today.

(4) has been extensively examined before, and after much work rejected as being too difficult, however we now have a very useful universal tool (a reserved or ‘link local’ MAC address at our disposal), and there are far more point-to-point links in networks than there were so in most cases the verification can be accomplished with little protocol chatter. A variant of this approach would use new format BPDUs only on point –to-point links, however there is a question as to whether MAC implementations truly provide a completely trustworthy indication of point-to-point.

(5) is unlikely to be deployed in practice in a plug and play fashion. Implementors are very conscious of the steady state load imposed on switches, and network administrators tend to resent ‘surplus’ traffic. Once an on-off switch is in place, any benefit of this approach has been negated.

This note proposes a version of approach (4).

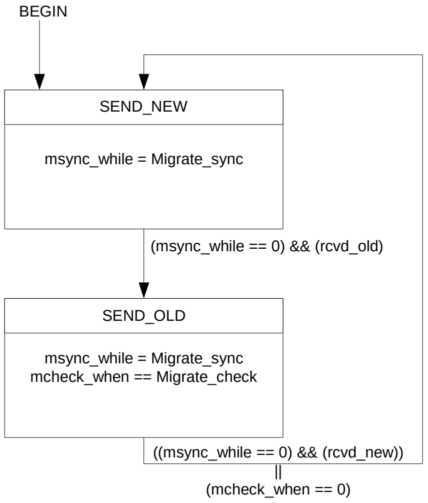
Message Format Selection

The proposed solution is as follows:

1. Retain the same multicast address for new style BPDUs.

   It would be entirely possible to move to a new multicast address, given the nature of the migration, however that would appear to be an egregious waste of a very limited resource.
2. Use a new and quite distinct protocol identifier for the new BPDUs.

   Since BPDUs are always confined to a single LAN that would allow us to revert to the Ethertype (Type in the Type/Length field) on 802.3 LANs. The natural encoding on other LANs would be to use the SNAP SAP.
3. Use the new format BPDUs on all media, both shared and point-to-point.
4. Allow ports other than Designated Ports to transmit Configuration BPDUs, without requiring some additional associated attached meaning, i.e. in the rapid protocol ‘designated confirmations’ can be transmitted by Root Ports and Alternate Ports , these ports should also be able to transmit BPDUs that are not to be interpreted as designated confirmations.
5. All new bridge ports should adopt the following algorithm when attached to a LAN (see following figure).


BEGIN
SEND_NEW
msync_while = Migrate_sync
(msync_while == 0) && (rcvd_old)
SEND_OLD
msync_while = Migrate_sync
mcheck_when == Migrate_check
((msync_while == 0) && (rcvd_new))
||
(mcheck_when == 0)

Suitable choices for the timer values Migrate_sync and Migrate_check are 3 seconds and 1 hour respectively.

This machine controls the choice of BPDU format on transmission. If a new bridge is added to a shared LAN it will start by sending a new style BPDU. It will receive and process BPDUs in any format for a 3 second period, but receipt of any old style BPDU will not cause it to change the format that it uses for transmission (transmission only occurs as required by the algorithm). If all the bridges attached to the LAN are new style bridges then they will see the new style BPDU and send new style BPDUs themselves (if they need to transmit). However if an old style bridge is present it will persist in sending old style BPDUs after 3 seconds has elapsed. After the initial three second period in the SEND_NEW state, any old style BPDU will cause a transition to the SEND_OLD state. In this state any transmissions required are sent as old style BPDUs, and the state is not changed for 3 seconds. If after 3 seconds a new style BPDU is received, then the machine reverts to the initial SEND_NEW state. It also reverts to the initial state every hour, just in case a last remaining old style bridge port has been removed from the LAN.

A likely scenario is that the remaining old style bridge port(s) are Root Ports or Alternate Ports. In this case when a new style Designated Port performs its hourly check to see if they have been removed from the LAN, they will be silent for a period until they time out the existing Root at which time they will attempt to become Designated. However this will drive the new bridge to send old style BPDUs once more, and the would be Designated port will be forced back to the Blocking port state well before it enters the Learning or Forwarding states.

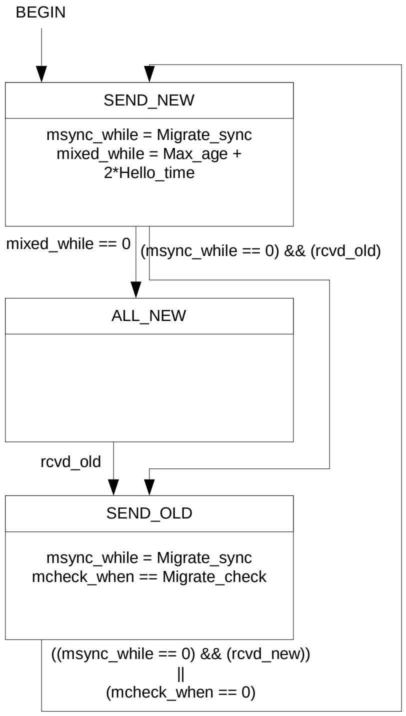
There is a possibility that the hourly checking performed by new bridges reverting to SEND_NEW from SEND_OLD will cause very temporary disruption and fast recovery in topologies where a new core is surrounded by old bridges that connect outward to new bridges themselves. If this is a concern then we could specify that the check to see if old style bridges are still present only occurs when a new bridge is added to the network or under explicit operator instruction.

Note: the above determination of BPDU format is made independently for each Bridge Port, any given Bridge may have some ports using new style BPDUs and some old style.

Quicker Shared Media

One question that is worth asking is ‘why use new style BPDUs on shared media?’. Apart from a concern that shared media may be incorrectly identified as point-to-point, and thus any solution to migration should work on both, there are advantages even on shared media in knowing that all bridges implement Rapid Reconfiguration.

The rapid transition of a Root Port to Forwarding can be achieved even if new and old bridges are mixed on a shared LAN. However a quicker transition of Designated Ports to Forwarding can be achieved even in the absence of an explict acknowledgment or ‘designated confirmed’. Clearly with multiple potential Designated, Root, or Alternate ports an acknowledgment from just one has no significance. But, using the single lost message assumption, knowing that all bridges attached to the LAN will retire prior root ports and synchronize with the topology within a second at most on request allows us to cut the transition from Listening through to Forwarding to twice the default Hello time. If an outright contradiction has not been received within that time, then a Designated port can be made forwarding even on a shared LAN. The above state machine does not actually tell us when we can be certain that no old bridges are present. The following modification remedies that.


BEGIN
SEND_NEW
msync_while = Migrate_sync
mixed_while = Max_age + 2*Hello_time
mixed_while == 0
(msync_while == 0) && (rcvd_old)
ALL_NEW
rcvd_old
SEND_OLD
msync_while = Migrate_sync
mcheck_when == Migrate_check
((msync_while == 0) && (rcvd_new))
||
(mcheck_when == 0)

In the all new state the initial value of fd_while assigned in the DBT and DLT states (see the state machines proposed in P802.1w/D2) can be cut to Hello_time.

Proposed RSTP BPDU Format

The following new BPDU format is proposed, assuming the migration scheme outlined here is acceptable.

Ethertype = ‘42’ (real value t.b.d)
Version = 1
BPDU Type = 1
Flags
Reserved
Root Identifier
Root Path Cost
Bridge Identifier
Port Identifier
Message Age
Max Age
Hello Time
Forward Delay

The flags field contains the following information:

Bit1 : Topology Change Flag

Bit2 : Topology Change Notification Flag

Bits 3 (less significant) and 4:

> Encode the following port roles:
>
> 0 Unknown
>
> 1 Alternate Port (or Backup)
>
> 2 Root Port
>
> 3 Designated Port

Bit 5 : Learning

Bit 6 : Forwarding

Bit 7 : In Sync

> (operational state matches administrative state, prior root ports retired or confirmed)

Bit 8 : Topology Change Acknowledge Flag.