

# Loop Cutting in the Original and Rapid Spanning Tree Algorithms

Mick Seaman

This note compares the different strategies for preventing temporary loops in the original (STP) and rapid reconfiguration variant (RSTP) of the spanning tree algorithm and protocol.

## Introduction

The Spanning Tree Algorithm and Protocol first standardized in ANSI/IEEE Std. 802.1D-1990 and unchanged in 802.1D-1998 [2], was based<sup>1</sup> on the work of Radia Perlman<sup>2</sup> at Digital Eqpt. Corporation [1].

The STP algorithm is essentially a Distance Vector Routing Protocol that computes the shortest path from each LAN in a Bridged Local Area Network to a distinguished Bridge, the Root Bridge, selected for having the lowest numeric value of a globally unique identifier, the Bridge Identifier.

Bridges connect to LANs through their Ports, and when the spanning tree computation is complete, Bridge Ports that form the shortest path from each LAN to the Root are in a 'Forwarding' state, while all others are 'Blocking'. The resulting active topology is 'spanning' (connects all LANs) and is a 'tree' (has no loops).

The proposed rapid reconfiguration enhancements to the algorithm (RSTP) [3], [5], [7], uses exactly the same algorithm to determine the final active topology of the network. Where the algorithms differ is in their approach to temporary loop prevention as a network reconfigures.

## Loop Cutting in STP

Once STP has determined that a Bridge Port should participate in the active topology, it waits for that spanning tree information to propagate throughout the network, or for different information to arrive. This delay is necessarily a worst case estimate of the propagation delay of spanning tree information across the entire network. This worst case estimate includes process scheduling, computation, and

transmission delays for each bridge, and an upper bound for the number of bridges from one edge of the network to the other ('the bridge diameter'). Allowing for differences in the time at which bridges are prepared to accept new information, for the subsequent propagation of that information, and for frames to stop flowing on a previous active topology<sup>4</sup>, the total delay is approximately three times the worst case delay across the network<sup>5</sup>.

## Loop Cutting in RSTP

Unlike STP, RSTP actually makes use of the bridge port roles (Root Port, Designated Port, Alternate Port, and Backup Port) to decide when bridge ports can be made Forwarding. RSTP is founded on the observation that a Root Port can be made forwarding immediately once all prior Root Ports have been made Blocking[3]<sup>6</sup>. The definition of 'prior' here is 'within the time for which information is not certain to have been propagated across the entire network', i.e. within a worst case delay.

RSTP's almost immediate transition of a Root Port to Forwarding leaves the task of transitioning a Designated Port to Forwarding rapidly. For ports attached to point-to-point links this can be done by telling the partner port (the Bridge Port attached to the other end of the link) that the local port is a Designated Port not in the Forwarding state. Then as soon as the partner port returns a message indicating that it is an Alternate Port in the Blocking state, or that it is a Root Port and all prior Root Ports are no longer Forwarding, the Designated Port can proceed to Forwarding. It is important that the returning message carry the best spanning tree information that the partner port could send were it the designated port for the link, so that the local port is sure that the answer 'go ahead' is relevant to its current spanning tree information, i.e. the partner port is definitely an Alternate Port or a Root Port.

---

<sup>1</sup> There are a number of minor differences. None of these need concern us here.

<sup>2</sup> I should also mention Tony Lauck who made the crucial step of recognizing that the 'loop detection' problem in bridges was a routing problem and assigned the task of finding a solution to Radia; George Varghese who developed the first procedural description; Floyd Backes and Paul Langille who constructed the first product implementation; and of course the team who built the LANBridge 100, thus giving the algorithm a home.

<sup>3</sup> There are other differences that need not concern us further here. They include propagating 'bad news' more quickly following a network failure [4], and quicker flushing of bad database entries [6].

---

<sup>4</sup> Quite separately it can be shown that the inclusion of this component is being significantly over cautious but the difference between two and three cross network delays is not material to the discussion here, involving as it does orders of magnitude improvement.

<sup>5</sup> Annex B of [2] provides a more detailed and accurate, though not significantly lower, estimate

<sup>6</sup> Or at least returned to the Listening state, which is the same thing in loop cutting terms.

## Loop Cutting in the Original and Rapid Spanning Tree Algorithms

To reply in this way, the Bridge for the partner port may have to return some Designated Ports that are Forwarding to the Listening state. To transition these ports back to Forwarding it asks their partner ports in turn.

This explicit acknowledgment procedure works well on point-to-point links but cannot be simply applied to shared LANs where there may be three or more bridges attached. However, if it is known that all bridges attached implement RSTP and will block prior root ports on requested, the hop-by-hop approach to loop cutting can be retained and the transition of the Designated Port to Forwarding shortened to twice the Hello Time<sup>7</sup>.

### Scenarios

This note includes a number of network scenarios, each comprising an initial physical and active topology together with the addition or removal of links and bridges to or from the network. For each of these scenarios the 'network-wide worst case delay' approach to prevention of temporary loops used by STP is contrasted with the 'hop-by-hop' approach used by RSTP. This latter approach can involve creating 'cuts' next to 'joins' in the active topology, and then propagating those cuts to their final locations as determined by spanning tree information.

The scenarios are:

1. Simple loops involving three bridges.
2. A new Root Bridge is added to a network, connecting two branches of the existing tree. In the final tree a cut will be introduced close to the old Root.
3. In the inverse of the prior example, the Root Bridge is removed from the network. The bridge that takes over as Root is 'on the other side' of the network.

### Graphical Notation

The following notation is used in the reconfiguration scenarios.

#### LEGEND

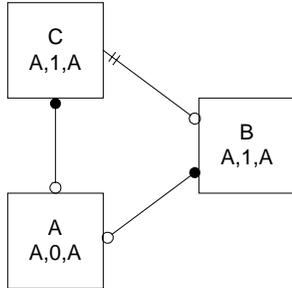
	Bridge X with Spanning Tree Parameters Root Bridge = Y Root Path Cost = 2 Designated Bridge = Z
	Various example point to point links, with Port Roles and Administrative and Operational Forwarding States at each end of the link as follows (reference numbers are for drawing package convenience):
	
	
	
	Root Port - Admin and Oper OFF (36)
	Root Port - Admin OFF, Oper ON (35)
	Root Port - Admin and Oper ON (42)
	Designated Port - Admin and Oper OFF(32)
	Designated Port - Admin OFF, Oper ON (31)
	Designated Port - Admin and Oper ON (41)
	Alternate or Backup Port - Admin and Oper OFF (25)
	Alternate or Backup Port - Admin OFF, Oper ON (24)
	Port is 'recent root'
	Port is 'recent backup'
	Transmitted messages:
	Config BPDU (13)

<sup>7</sup> To protect against one lost message, and assuming that it takes no more than a second to block prior root ports. Of course if better performance guarantees were available the transition could be made substantially quicker.

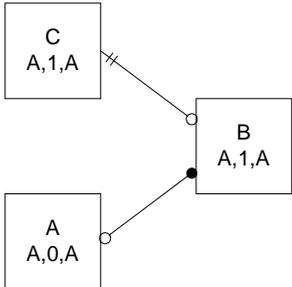
# Loop Cutting in the Original and Rapid Spanning Tree Algorithms

## Simple Loop Scenarios

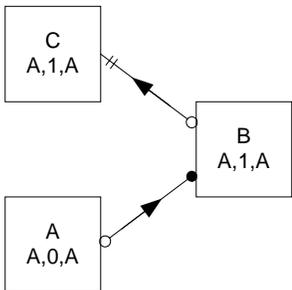
The next figure shows a very simple redundantly connected network.



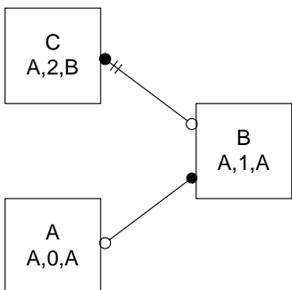
Assume that the A-C link is lost:



STP will respond as follows. For a period C will effectively ignore further configuration messages from A relayed via B, as C ages out its memory of A.

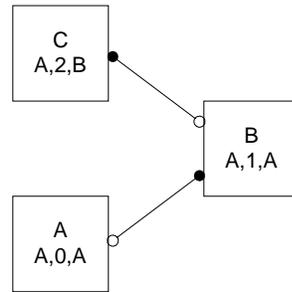


Finally C will age out information from A that it had received on C-A, and move its Root Port to C-B.

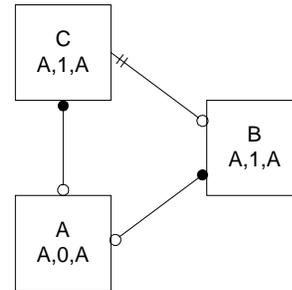


Then C will wait. It may have additional ports not shown in the figure, and in any case neither STP or RSTP treat a single port as a special case. The waiting period ensures that all other bridges have forgotten about any prior path to A, and then that they all have received new information from A. This waiting period necessarily accomodates worst case delays, lost messages, and a maximally sized network. Finally after a period during which additional configuration

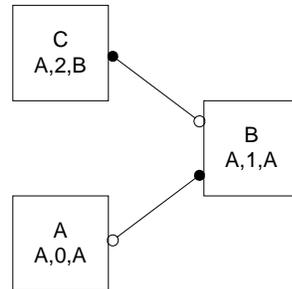
messages referesh the information held at C-B, the final active topology is arrived at.



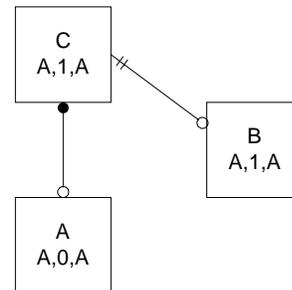
Starting from the same initial network configuration:



If C implements RSTP it will respond to the loss of A-C, by immediatly electing C-B as its Root Port and make it forwarding, thus arriving at the final configuration with minimal delay.

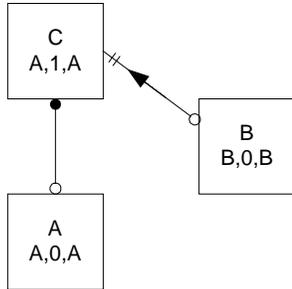


If, in the initial configuration, the link A-B is lost:

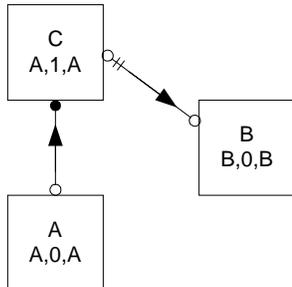


STP will respond as follows. First C has to age out the information previously received on C-B, while B also ages out its memory of A. On the assumption that the latter occurs first, B will send a message to C claiming to be Root.

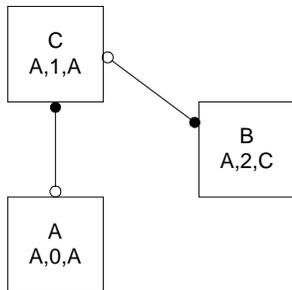
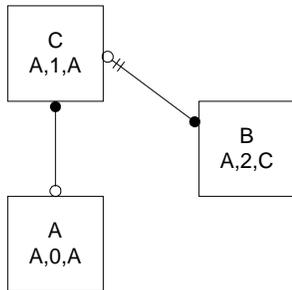
## Loop Cutting in the Original and Rapid Spanning Tree Algorithms



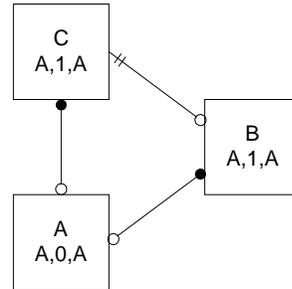
But C will discard this message as containing inferior information on C-B, until it has aged out that prior information. Then C will decide that it should be designated for C-B and the next BPDU received from A will stimulate transmission on C-B.



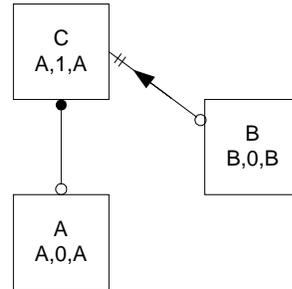
Now B knows the final topology, but C has to wait to ensure that this information pervades the network before finally making C-B forwarding, restoring service to any systems connected to B.



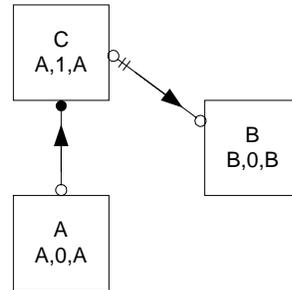
From the same initial configuration, with B and C using RSTP:



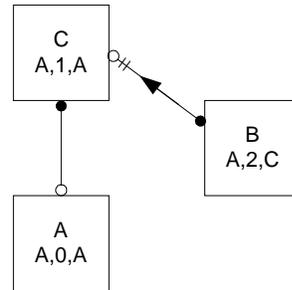
B responds to the loss of B-A by becoming the root itself and sending this new information to C.



C will process this information, rather than simply discarding it as inferior since it is from the designated bridge for C-B. Having processed the information, C updates the information on C-B<sup>8</sup> and transmits in its turn.



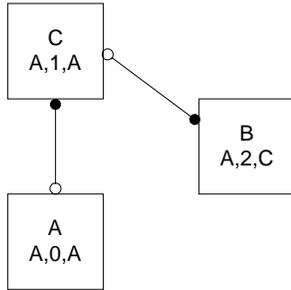
B receives this information, which includes a request for role confirmation (since C-B is Designated but not Forwarding), and responds – having updated its Root Port.



<sup>8</sup> The source of the information that C is using is not C-B itself, this is key because it prevents two bridges facing each other from "counting to infinity".

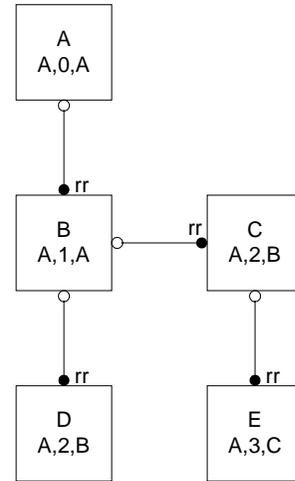
## Loop Cutting in the Original and Rapid Spanning Tree Algorithms

When C-B receives this confirmation, it can become forwarding immediately. Effectively B has absorbed the 'cut' in the active topology.

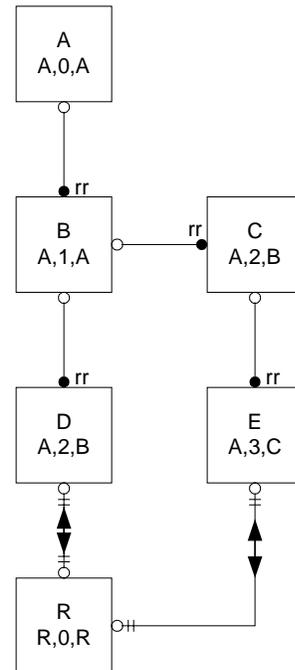


## New Root Bridge Scenario

The next figure shows a possible initial network configuration



When a new higher priority root, R, is attached to both D and E<sup>9</sup>, all these bridges will send BPDUs on their newly operational ports. All of these are believed to be Designated, and are Listening.

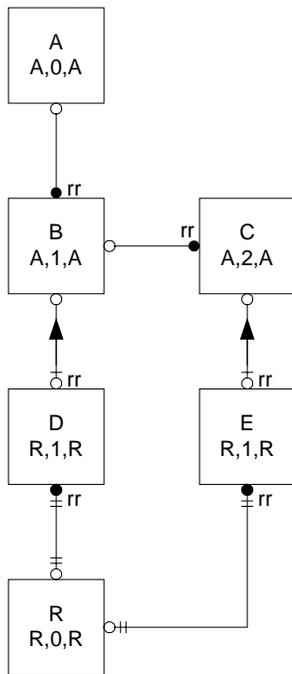


<sup>9</sup> Assuming it is possible to do this simultaneously.

## Loop Cutting in the Original and Rapid Spanning Tree Algorithms

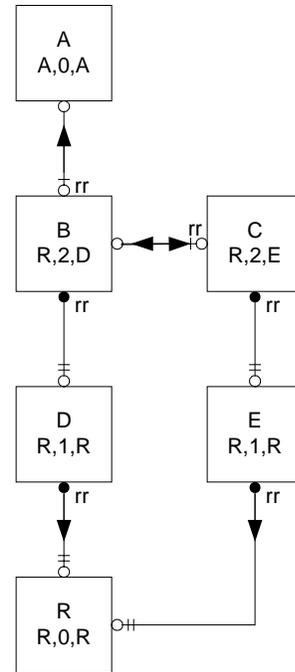
Following the receipt of these BPDUs, D and E will have selected R as the new Root, and will generate BPDUs to B and C respectively. Following STP's state transition rules, R, D, and E will have to wait until the new information propagates throughout the network, at an unknown rate and through an unknown number of bridge hops. Then the ports attached to links R-D and R-E can transition to Forwarding.

With RSTP, the next step is to transition the ports D-B and E-C back to the Listening state. This is begun as part of processing the BPDU from R, but does not have to complete before BPDUs are propagated to B and C.



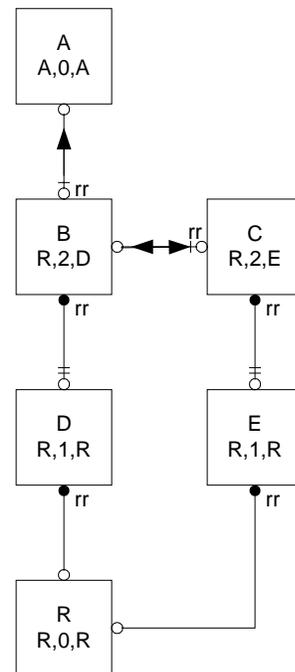
Once this has been done the new root ports can be made forwarding, and a BPDU sent to R containing the port role confirmation<sup>10</sup>.

<sup>10</sup> R's inclusion of its Designated Role and Listening State in the initial BPDU is sufficient indication of its need for confirmation, and causes D and E to block their prior root port and respond as described.



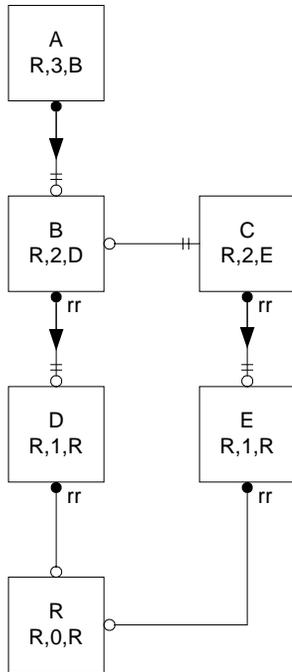
From D and E's point of view the 'cuts' in the network that prevent any temporary loop have been moved from their new root ports to their designated ports. At about the same time B and C will be sending each other the new topology information. Both will have started to block their prior root ports.

Following the receipt of the role confirmation BPDUs, R can transition both its ports directly to Forwarding.

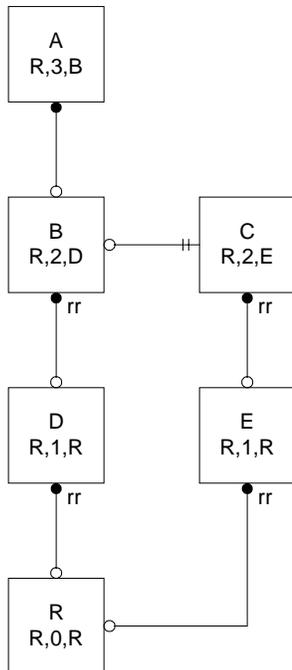


As B and C block their prior root ports to A and B respectively, they too can send role confirmation BPDUs; as can A immediately on receipt of the BPDU on its new root port, since it has no prior root port to block.

## Loop Cutting in the Original and Rapid Spanning Tree Algorithms



On receipt of these BPDUs, D, E, and B can transition their designated ports to Forwarding. The cut in the potential loop has been moved from D-B and E-C to C-B, its final position in the topology as determined by spanning tree parameters.

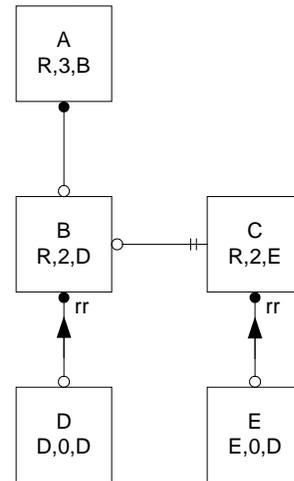


## Root Bridge Removal Scenario

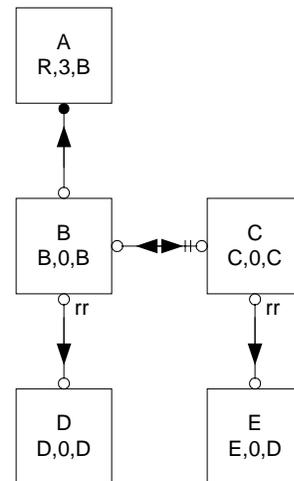
If we begin with this final topology and remove the Root Bridge R, the two algorithms respond as follows.

With STP there is an initial period while A, B, C, D, and E time out the last messages originated by R. One by one they will each decide that they are the Root, and C will decide that C-B should be Forwarding. Then each bridge will start to receive messages originating from A. These do not contradict C's decision to make C-B forwarding, so C continues to wait to ensure (a) that all other bridges have timed out R and that no newer Root will emerge to maintain C-B in the blocking state, and (b) knowledge of the new topology as established by A has propagated throughout the network. Once a suitable worst case time for these two events has elapsed, C-B is made Forwarding.

With RSTP, both D and E react to the loss of their links to R and send new BPDUs claiming that they are the new Root on their links to B and C respectively.

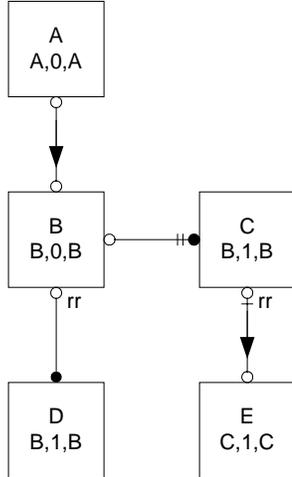


B and C will both accept the new information since it comes from bridges that were designated for their respective links, but will conclude that they are better new roots, and will send this information on all their ports.

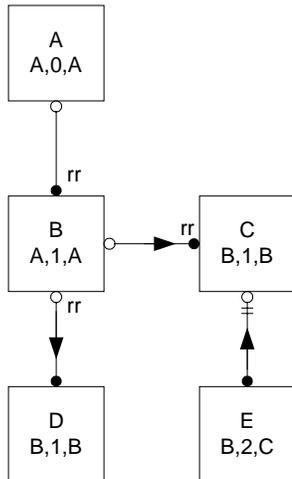


## Loop Cutting in the Original and Rapid Spanning Tree Algorithms

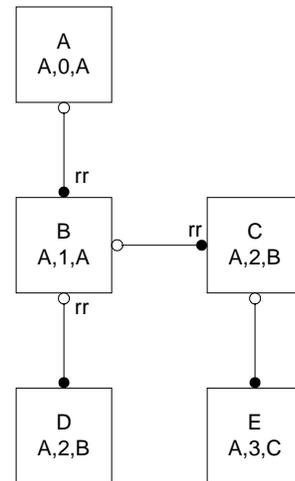
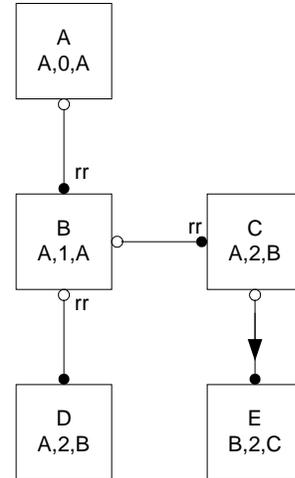
Once this has been received, A in its turn will have concluded it is the new Root, and will transmit. Meanwhile C will begin to block its port C-E, to allow its new root port to go Forwarding, and will have sent a BPDU requesting role confirmation on this port so that it can make it Forwarding once more.



B receives the BPDU from A, adjusts its spanning tree parameters accordingly and forwards the new information, but does not have to block its previous root port since the new root port is already forwarding and A did not need to request role confirmation. As C-E becomes blocking, C can now make its root port Forwarding, thus moving the cut in the topology to its designated port. E responds to the request for role confirmation.



When E's response is received C can transition its designated port to Forwarding. Effectively the cut in the topology has been absorbed by E. One last message will be transmitted from C to E to update E's spanning tree parameters, but the final topology has already been arrived at.



### References

- [1] Perlman, R., "A Protocol for Distributed Computation of a Spanning Tree in an Extended LAN", Ninth Data Communications Symposium, 1985.
- [2] ANSI/IEEE Std 802.1D-1998.
- [3] High Availability Spanning Tree. Mick Seaman. Rev 1.1 10/26/98
- [4] Speedy Tree Protocol. Mick Seaman.
- [5] Truncating Tree Timers. Mick Seaman. Rev 1.0 January 11<sup>th</sup> 1999
- [6] Faster flushing with fewer addresses. Vipin Jain, Mick Seaman.
- [7] P802.1w/D2. 'Rapid Reconfiguration' Ed. Tony Jeffree.